SecTalks 0x14 ASM Reference

Basic X86: MOV

mov dst,src

- mov dst,src
    - copies src to dst
    - src can be register, [reference], value
    - dst can be register, [reference]
    - de-[referencing] invalid memory causes a segmentation fault / access violation

Basic X86: LEA

lea dst,src

- lea dst,src
  - loads the effective address of src into dst
  - src can be register, [reference]
  - dst can be register, [reference]
  - lea [reference] will put the value of reference into dst
  - lea dst, value is impossible. why?

Basic X86: PUSH / POP

push src / pop dst

- push puts something “ONTO” the stack
  - val can be register, value
- pop removes something “OFF” the stack
  - val can be register, stores the item removed from the stack
  - “pop” removes the last item “pushed” on the stack. i.e. the first item “push”’ed gets “pop”’ed last.

call dst / jmp dst / ret

- call pushes the next instruction’s address onto the stack
    - then transfers execution to dst
    - dst can be register, [reference], value
- jmp simply transfers execution to dst
- ret pops an address off the stack, and transfers execution to it.
    - this does not require a “target” register

Basic X86: JXX (Conditional Jump)

```
j** dst
```

- jmp transfers control to dst if the condition is true:
  - je / jz = jmp if equals flag is set. why?
  - jge = “jmp if greater than or equal”
  - etc...
  - (reference) http://unixwiz.net/techtips/x86-jumps.html

Basic X86: CMP

cmp dst,src

- cmp “fakes” a sub operation, and sets flags according to the result of said operation.
  - neither src nor dst is modified
  - src/dst can be register, [reference], value
- cmp dst,dst is often used as a test for zero

Basic X86: ADD / SUB / XOR

add dst,src

- _math_instruction_ dst,src
    - performs maths instruction dst _math_instruction_ src
    - src can be register, [reference], value
    - dst can be register, [reference]
    - de-[referencing] invalid memory causes a segmentation fault / access violation