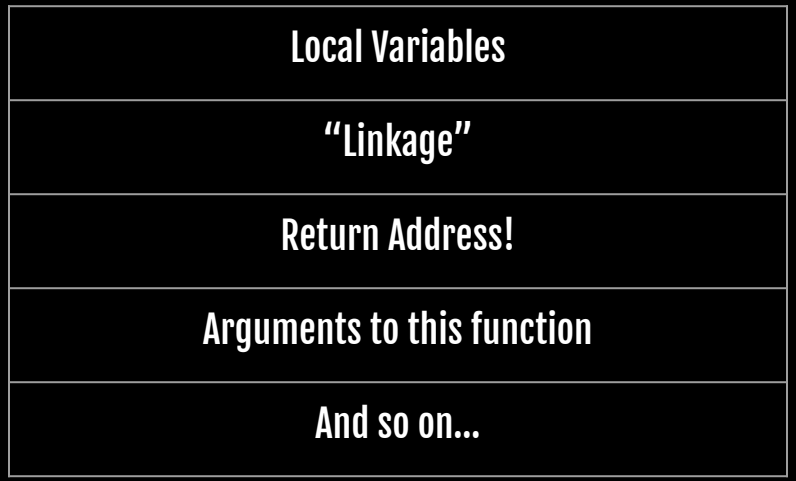
Binary Exploitation

SIGPWNY

Where we last left our heroes...

The Stack

“It’s where things go™”

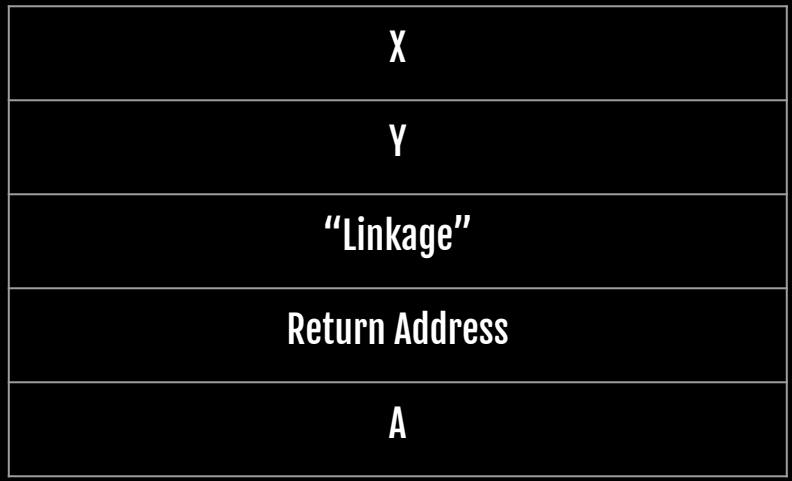


Example

```
int demo (int A) {
    int Y;
    int X;

    X = A + 2;
    Y = A - X;

    return A + X + Y;
}
```

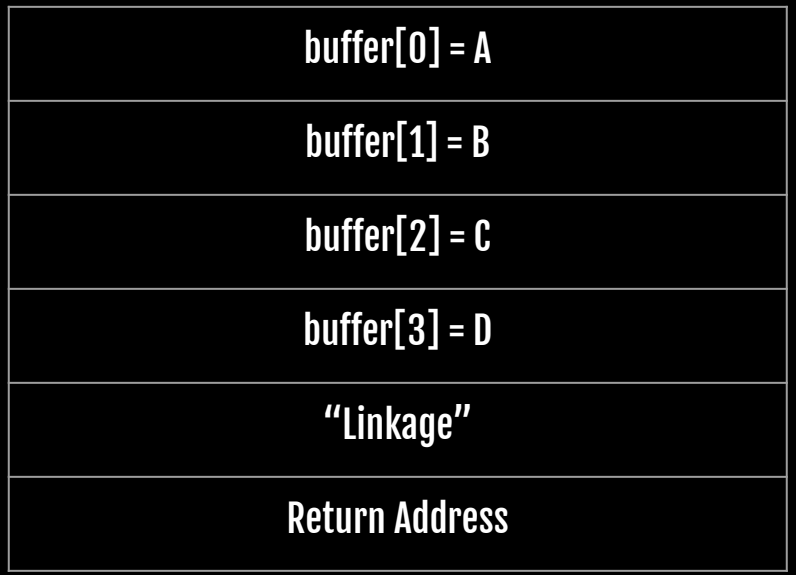


Another Example

```
int demo () {
    char buffer[4];
    buffer[0] = ‘A’;
    buffer[1] = ‘B’;
    buffer[2] = ‘C’;
    buffer[3] = ‘D’;
    return 0;
}
```

Another Example

```
int demo () {
    char buffer[4];
    buffer[0] = ‘A’;
    buffer[1] = ‘B’;
    buffer[2] = ‘C’;
    buffer[3] = ‘D’;
    return 0;
}
```



What about user input?

```
int read_input () {
    char buffer[4];
    gets(buffer);
    printf(“You said: %s\n”, buffer);
    return 0;
}
```

What about user input?

```
int read_input () {
    char buffer[4];
    printf(“Say something!\n”);
    gets(buffer);
    printf(“You said: %s\n”, buffer);
    return 0;
}
```

What about user input?

```
int read_input () {
    char buffer[4];
    printf(“Say something!\n”);
    gets(buffer);
    printf(“You said: %s\n”, buffer);
    return 0;
}

./program
Say something!
1234
You said: 1234
```

What about user input?

```
int read_input () {
    char buffer[4];
    printf(“Say something!\n”);
    gets(buffer);
    printf(“You said: %s\n”, buffer);
    return 0;
}

./program
Say something!
1234
You said: 1234
```

What about user input?

```
int read_input () {
    char buffer[4];
    printf(“Say something!\n”);
    gets(buffer);
    printf(“You said: %s\n”, buffer);
    return 0;
}

./program
Say something!
1234
You said: 1234
```

What about user input?

```
int read_input () {
    char buffer[4];
    printf(“Say something!\n”);
    gets(buffer);
    printf(“You said: %s\n”, buffer);
    return 0;
}

./program
Say something!
1234
You said: 1234
```

What about user input?

```
int read_input () {
    char buffer[4];
    printf(“Say something!\n”);
    gets(buffer);
    printf(“You said: %s\n”, buffer);
    return 0;
}

./program
Say something!
1234
You said: 1234
```

What about a lot of user input?

```
int read_input () {
    char buffer[4];
    printf("Say something!\n");
    gets(buffer);
    printf("You said: %s\n", buffer);
    return 0;
}

./program
Say something!
1234AAAABBBB
You said: 1234AAAABBBB
Segmentation fault
```

What about a lot of user input?

```
int read_input () {
    char buffer[4];
    printf("Say something!\n");
    gets(buffer);
    printf("You said: %s\n", buffer);
    return 0;
}

./program
Say something!
1234AAAABBBB
You said: 1234AAAABBBB
Segmentation fault
```

1
2
3
4
AAAA
BBBB

How to play

1. Program asks for user input, stores it in a variable
2. You give extra data to “overflow” the variable
3. Carefully choose extra data to change program execution flow

Your First Challenge

```
int main()
{
      printf("This is SIGPwny thing, go\n");
      vulnerable();
}
```

Your First Challenge

```
void print_flag()
{
      printf("GOOD JOB! Run this on server for flag");
}
```

Your First Challenge

```
void vulnerable()
{
      int changethis = 0x12345678;
      char buf[4];
      gets(buf);
      if(changethis != 0x12345678) {
            print_flag();
      }
}
```

Your First Challenge

```
void print_flag()
{
	printf("GOOD JOB! Run this on server for flag");
}

void vulnerable()
{
	int changethis = 0x12345678;
	char buf[4];
	gets(buf);
	if(changethis != 0x12345678) {
		print_flag();
	}
}

int main()
{
	printf("This is SIGPwny thing, go\n");
	vulnerable();
}
```