

# Outline

---

- ▶ Format String Vulnerabilities
- ▶ **Integer Overflow Vulnerabilities**
- ▶ Scripting Vulnerabilities

# Integer Representation

---

In mathematics integers form an infinite set.

In a computer system, integers are represented in binary.

- ▶ The representation of an integer is a binary string of fixed length (precision), so there is only a finite number of “integers”.
- ▶ Signed integers can be represented as two’s complement: the Most Significant Bit (MSB) indicates the sign of the integer:
  - MSB is 0: positive integer
  - MSB is 1: negative integer.

# Integer Overflow

---

An operation causes its integer operand to increase beyond its maximal value, or decrease below its minimal value. The results are no longer correct.

- ▶ Unsigned overflow: the binary cannot represent an integer value.
- ▶ Signed overflow: a value is carried over to the sign bit

Possible operations that lead to integer overflow.

- ▶ Arithmetic operation
- ▶ Type conversion.

Integer overflow is difficult to spot, and can lead to other types of bugs, frequently buffer overflow.

# Arithmetic Overflow

In mathematics:  $a+b>a$  and  $a-b<a$  for  $b>0$

- ▶ Such obvious facts are no longer true for binary represented integers

```
#include <stdio.h>
#include <string.h>

int main(int argc, char* argv[]) {

    unsigned int u1 = UINT_MAX;
    u1 ++;
    printf("u1 = %u\n", u1);

    unsigned int u2 = 0;
    u2 --;
    printf("u2 = %u\n", u2);

    signed int s1 = INT_MAX;
    s1 ++;
    printf("s1 = %d\n", s1);

    signed int s2 = INT_MIN;
    s2 --;
    printf("s2 = %d\n", s2);

}
```

➡ 4,294,967,295

➡ 0

➡ 4,294,967,295

➡ 2,147,483,647

➡ -2,147,483,648

➡ -2,147,483,648

➡ 2,147,483,647

# Example 1: Bypass Length Checking

Incorrect length checking could lead to integer overflows, and then buffer overflow.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <limits.h>

char buf[128];
void combine(char *s1, unsigned int len1, char *s2, unsigned int len2) {
    if (len1 + len2 + 1 <= sizeof(buf)) {
        strncpy(buf, s1, len1);
        strncat(buf, s2, len2);
    }
}

int main(int argc, char* argv[]) {
    unsigned int len1 = 10;
    unsigned int len2 = UINT_MAX;
    char *s1 = (char *)malloc(len1 * sizeof(char));
    char *s2 = (char *)malloc(len2 * sizeof(char));
    combine(s1, len1, s2, len2);
}
```

Buffer Overflow!

len1 + len2 + 1 = 10 < 128  
strncpy and strncat will be executed.