VotD: Integer Overflow

Description. See CWE-190 (http://cwe.mitre.org/data/definitions/190.html), and CWE-680 (http://cwe.mitre.org/data/definitions/680.html)

Examples: integer-overflow.zip (https://uncw.instructure.com/courses/16302/files/351736/download?wrap=1). Also for an interesting example from Firefox, check out the CVE-2010-2753 (http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2010-2753), and corresponding bugzilla (https://bugzilla.mozilla.org/show_bug.cgi?id=571106), especially the patch to fix (https://bug571106.bugzilla.mozilla.org/attachment.cgi?id=451552&action=diff&collapsed=&context=patch&format=raw&headers=1).

- Download the code to your Ubuntu VM and unzip it to a location you will remember. It's probably a good idea to create a 'VotD' folder somewhere and have subfolders in it for each vulnerability we will look at.
- This example is in Java. Use `sudo apt install default-jdk` to install the Java JDK on Ubuntu.
- Run the `make` command to execute the code.

Mitigations:

- Check the size of your integers, considering what would happen if it wrapped around
- Watch the casting - don't just ignore those compiler warnings!
- Libraries such as SafeInt or BigInteger might be more suitable if the problem is very complex

Notes

- A wraparound combined with a `malloc` operation can result in a zero-sized buffer being allocated - leading to a zero-byte buffer, which will always be overflowed.
- In practice, most integer wraparounds come from improper casting, not as much from math operations.
- It's impractical to always check every integer for wraparound after every operation. But, keep this as a consideration in sensitive situations.
- Trivia: Psy Youtube (http://www.bbc.com/news/world-asia-30288542); Deep Impact (https://en.wikipedia.org/wiki/Deep_Impact_(spacecraft))