Memory Mismanagement

2 min

Most modern programming languages do not require the programmer to manage memory themselves. Still, some older languages like C and C++ do, and some modern languages allow the programmer to manage memory themselves.

From a security perspective, manual memory management (MMM for short) is a problem. There are many ways that it can go wrong, with various consequences. This isn't to say that programs that use MMM can't be secure, but it is another place for human error to become an issue.

Here are just some of the fun ways that improperly managed memory can cause problems:

- Memory Leak: If you want to store data in, say, a variable, you need to allocate space for that data in memory. When that variable is no longer used, you need to *free* that space so the memory can be used for something else. If the space isn't freed, a program can use more and more memory the longer it runs, eventually causing performance issues or crashes. If this happens in kernel memory (memory used by the operating system's core), a memory leak can be terrible.

- Use After Free: Once the space used to store a piece of data has been freed, attempting to access that data can cause unpredictable behavior and potentially allow exploitation.

- Double Free: Once memory has been freed, trying to free it again can cause issues, too. The data structures used to manage memory can become corrupted, causing unpredictable behavior and potentially allowing exploitation.

- Buffer Overflow: This is a classic memory issue responsible for everything from glitched apps to famous pieces of

Preview: Docs Loading link description

malware

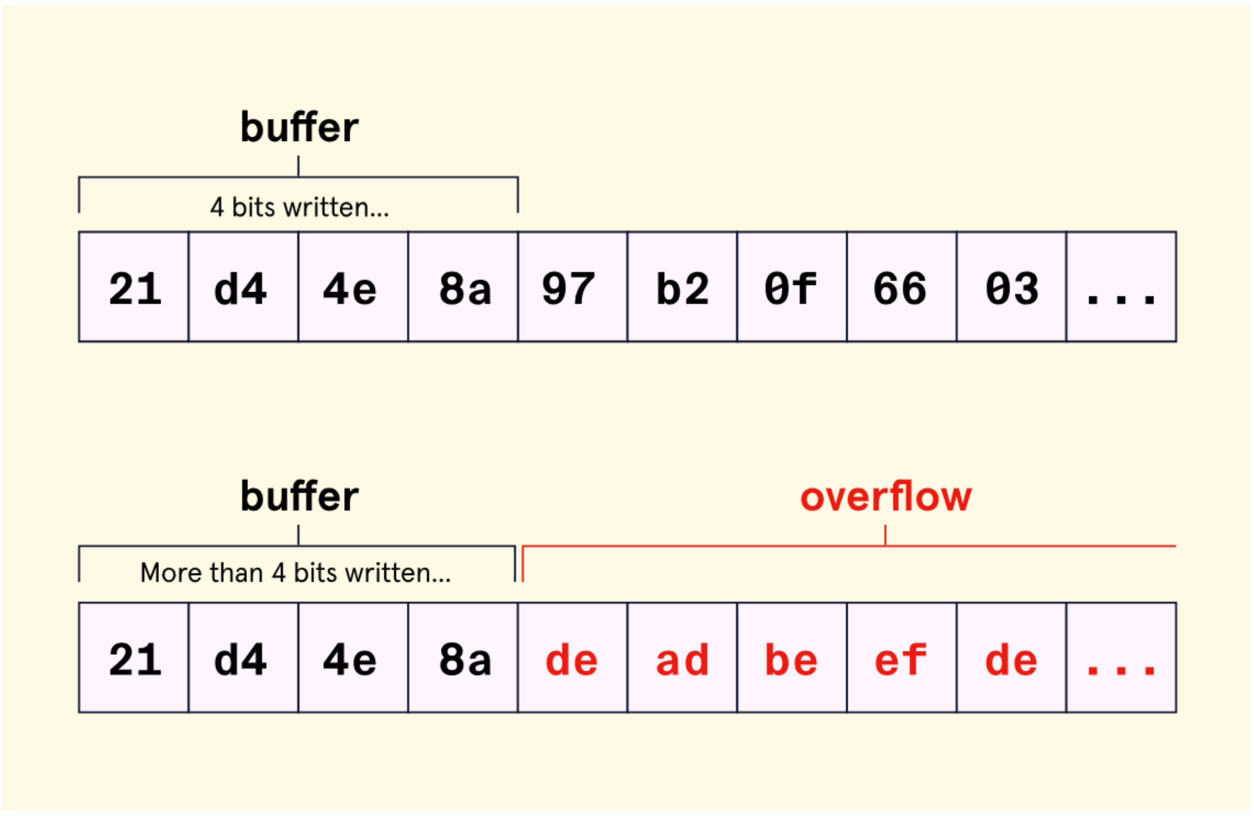
. Buffers are a memory structure used to store multiple pieces of data. Buffers are created with a fixed size, so they have a limit on how much data they can hold. Buffer Overflows occur when too much data is written to a buffer. In the worst case, this can cause the memory to be overwritten immediately after the buffer is overwritten. Buffer overflows can allow for arbitrary code execution or, at best, cause unpredictable behavior.

Buffer Overread: Buffer overreads (sometimes erroneously referred to as "buffer underflows") occur when the program tries to read more data from a buffer than the buffer contains. This can cause data to be read after the buffer, potentially exposing sensitive data. The famous Heartbleed

Preview: Docs Loading link description

vulnerability

was caused by a buffer overread.


buffer
4 bits written...
21 d4 4e 8a 97 b2 0f 66 03 ...
buffer
overflow
More than 4 bits written...
21 d4 4e 8a de ad be ef de ...