# Software Security

**Memory Corruption Bugs:**

- Programming errors it is a property of specific types of languages called memory-unsafe languages
- It can lead to arbitrary reading, writing and hijacking

Pointers:

When pointers point to somewhere they shouldn't is where a memory bug can be introduced.

**Memory Safety:**

- Spatial Safety - When object boundary access is violated via pointers creates a spatial safety violation.
- Temporal Safety - When a pointer is used to attempt to access or deallocate memory that has already been deallocated. This is temporal because the memory was previously allocated to the program.
- Going beyond the end of an array can lead to arbitrary execution
- Using memory after it's deallocated leads to an arbitrary write

If the buffer overflow overwrites the return pointer, some other code can be executed as the functions return to the wrong place. Assuming a seg fault is not triggered.

Modern CPUs don't allow you to write to regions of memory you can execute or execute from regions of memory you can write to. Stack canaries are used to prevent this

**Format String Errors:**

- Formatted output functions consist of a format string and a variable number of arguments.
- The results are undefined if there are not enough arguments for all the conversion specifications.

Conversion Specification %n will store the number of characters before the %n in the variable specified in the arguments. Therefore a memory write is performed by the printf function.

Conversion Specification %p will print memory from the stack if no arguments are provided.

String format errors can lead to data corruption of local variables, and arbitrary code execution if the return pointer is overwritten.