Call to memory access function may overflow buffer

Name: Call to memory access function may overflow buffer

Description: Incorrect use of a function that accesses a memory buffer may read or write data past the end of that buffer.

ID: cpp/overflow-buffer

Kind: problem

Severity: recommendation

The software uses a function to access a memory buffer in a way that may read or write data past the end of that buffer. This may result in software instability, improper access to or corruption of sensitive information, or code execution by an attacker.

Recommendation

When accessing buffers with functions such as `memcpy`, `memset` or `strncpy`, ensure that the size value for the operation is no greater than the amount of space available in the destination buffer. Failure to do this may permit a buffer overwrite to occur. Also ensure that the size value is no greater than the amount of data in the source buffer, to prevent a buffer overread from occurring.

Example

In the following example, `memcpy` is used to fill a buffer with data from a string.

```
1 const char *message = "Hello";
2 char password[32];
3 char buffer[256];
4
5 memcpy(buffer, message, 256);
```

Although the size of the operation matches the destination buffer, the source is only 6 bytes long so an overread will occur. This could copy sensitive data from nearby areas of memory (such as the local variable `password` in this example) into the buffer as well, potentially making it visible to an attacker.

To fix this issue, reduce the size of the `memcpy` to the smaller of the source and destination buffers, `min(256, strlen(message) + 1)`. Alternatively in this case it would be more appropriate to use the `strncpy` function rather than `memcpy`.

References

- Common Weakness Enumeration: CWE-119.
- Common Weakness Enumeration: CWE-121.
- Common Weakness Enumeration: CWE-122.
- Common Weakness Enumeration: CWE-126.