

What if the buffer is overflowed?

➔ `strcpy` does not check whether the string at `*str` contains fewer than 126 characters

● If a string longer than 126 bytes is copied into buffer, it will overwrite adjacent stack locations

02 E E E E E E E E E E

0x0000000000000000

OXFORD







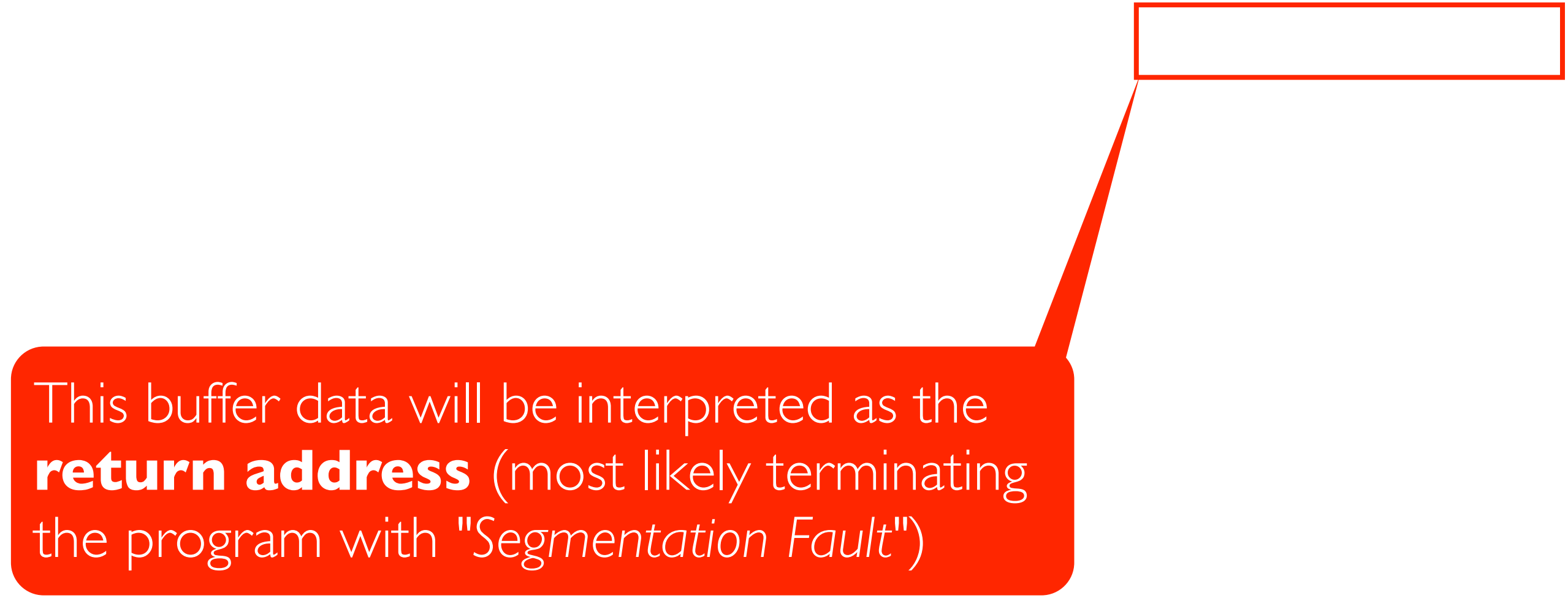
Args

Return Address

Base Pointer

buf

Overflow



The diagram illustrates a buffer overflow. A large blue rectangle represents a memory buffer. A red arrow points from the top right corner of this buffer to a smaller, empty red-outlined rectangle located above and to the right. A red callout box with rounded corners points to the red arrow. Inside the callout box, text explains that the data in the buffer is being interpreted as a return address, which will likely cause a segmentation fault.

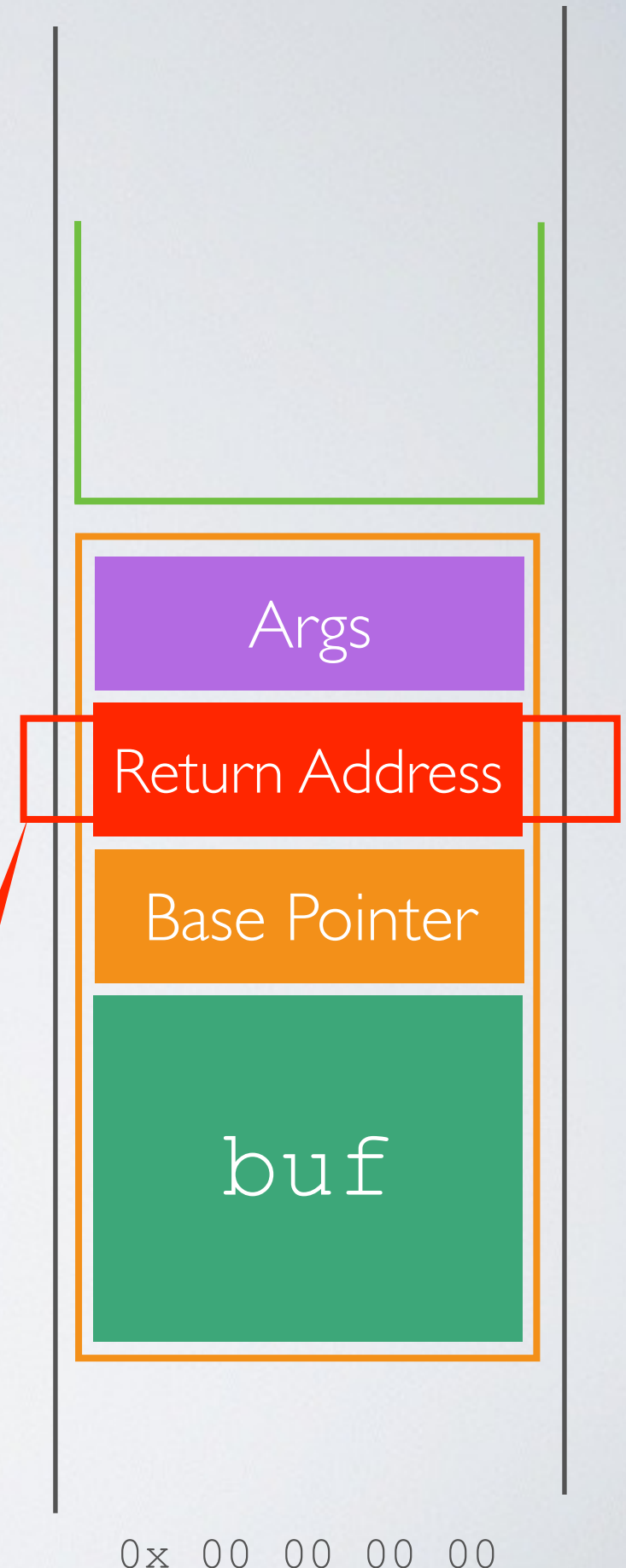
This buffer data will be interpreted as the **return address** (most likely terminating the program with "*Segmentation Fault*")

0x FF FF FF FF

What if the buffer is overstuffed?

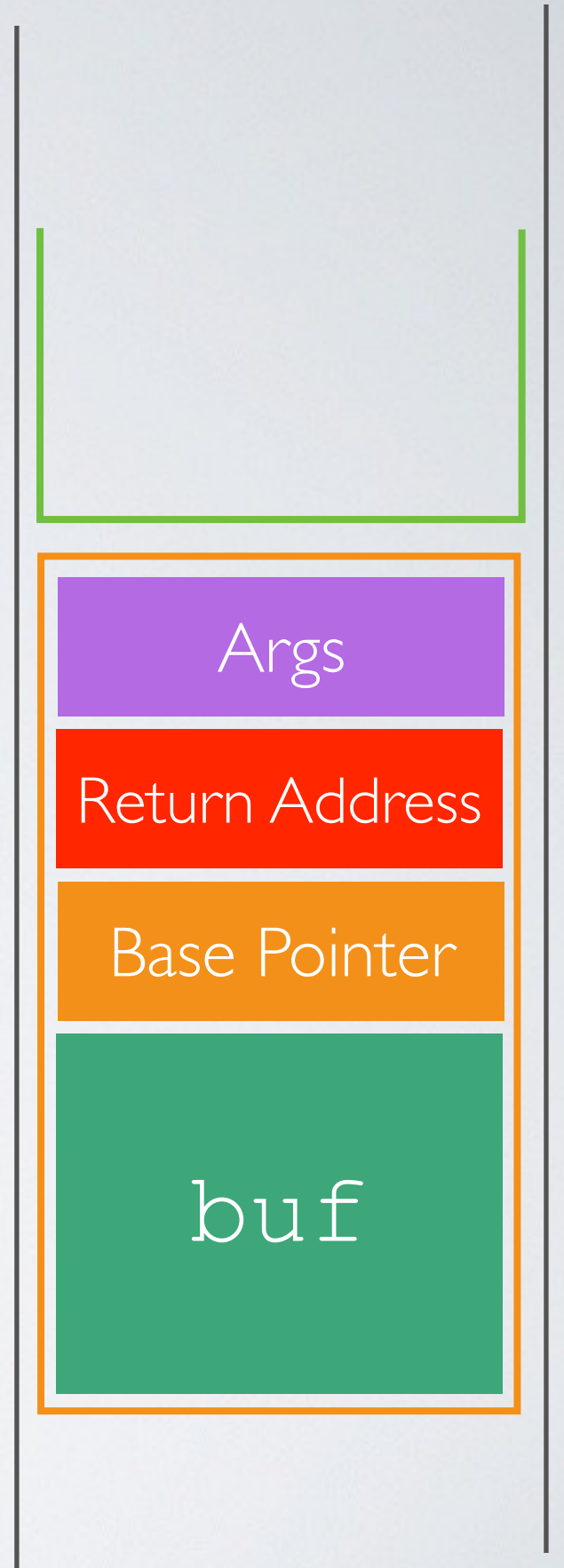
- ➔ `strcpy` does not check whether the string at `*str` contains fewer than 126 characters
- ⦿ If a string longer than 126 bytes is copied into buffer, it will overwrite adjacent stack locations

This buffer data will be interpreted as the **return address** (most likely terminating the program with "Segmentation Fault")



Injecting Code

0x FF FF FF FF



0x 00 00 00 00