March 25th: course day #5

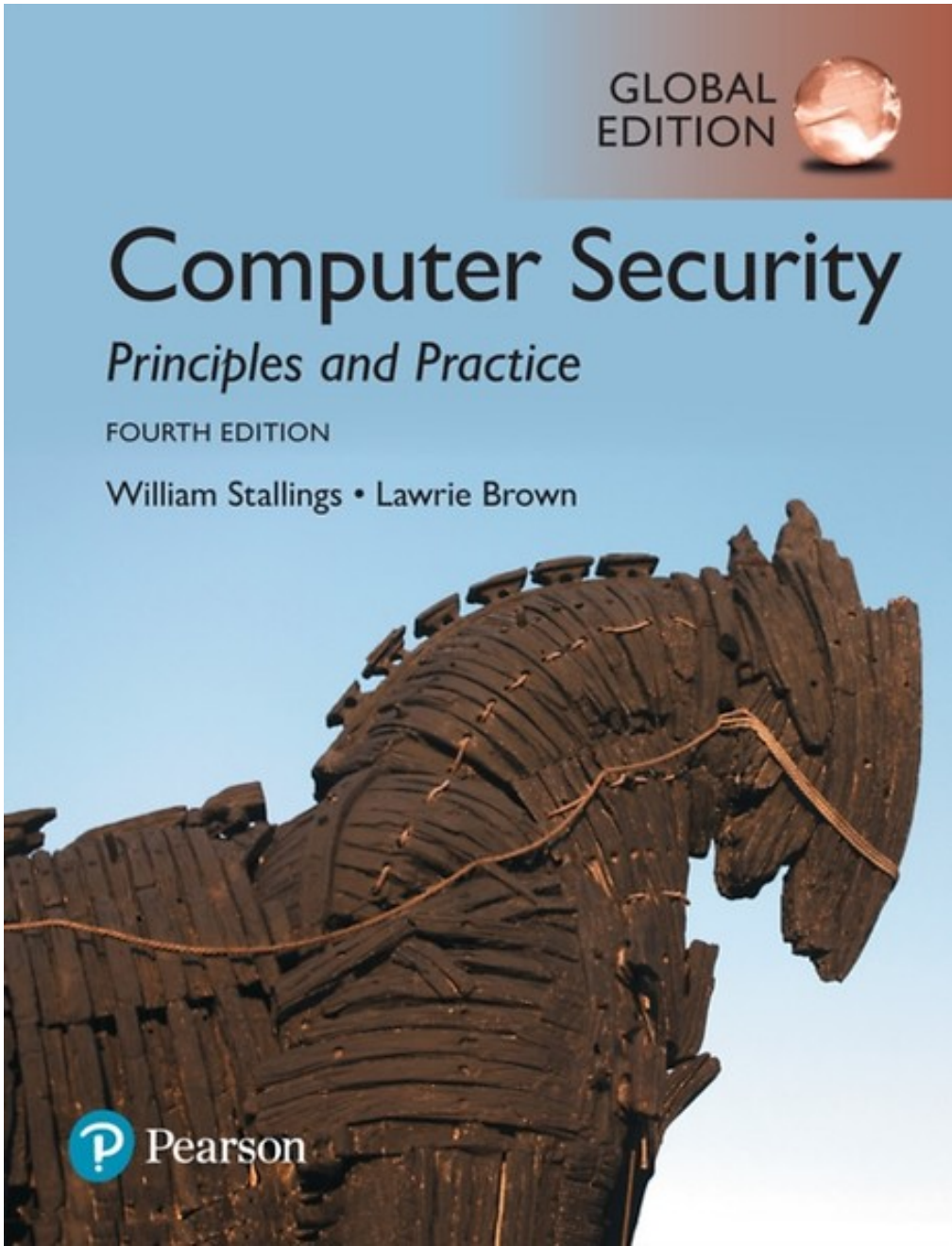
Theme B: Software and system security

Case: The Code Red buffer overflow attack.

Stallings & Brown
Chapter 10:
- buffer overflows: 341-347, 364-368

Chapter 13:
- cloud computing: 445- 451, 457-464
- IoT: 466-473

Additional mandatory literature:
- *"Code red"*, Communications of the ACM, December 2001.
- *"Webcam hack shows vulnerability of connected devices",* Engineering & Technology December 2016.



Student presentations
March 11th

Present *"Code red"*, Communications of the ACM, December 2001.
(5 pages)

Present *"Webcam hack shows vulnerability of connected devices",* Engineering & Technology December 2016.
(1 page, small print)

Buffer overflows in C
Stallings & Brown, Figure 10.1, p 344

```
int main(int argc, char *argv[]) {
    int valid = FALSE;
    char str1[8];
    char str2[8];

    next_tag(str1);
    gets(str2);
    if (strncmp(str1, str2, 8) == 0)
        valid = TRUE;
    printf("buffer1: str1(%s), str2(%s), valid(%d)\n", str1, str2, valid);
}
```

str1 and str2 are character arrays (buffers).
str2 may overflow into str1

- for example, an overflow may trick the comparison str1/str2 to succeed

See my programs on moodle:

- buffer1.c is a selv-contained C-version (with a definition of next_tag())
- Buffer1.java is a Java version (protects against buffer overflows)