Hacking and the Art of the Unexpected: What DEFCON's CTF can teach you about computer science

Aleatha Parker-Wood
Conservatoire National des Arts et Métiers, Paris
University of California, Santa Cruz

5/19/2013

The talk

- What is hacking and why should I hack?
- Why are CTF and similar contexts a good place to hack?
- Some examples of hacking
- What are the downsides of hacking?

Hacking is not

- DDOSing
- Destroying and using brute force
- Drinking RockStar in your mom’s basement (unless you want to…)
- Illegal (unless you want to. Please don’t.)

Hacking is

- Puzzle solving
- Learning a system deeply enough to use it in non-obvious ways
- Finding weaknesses and exploiting them

Hacking is applied security

- Formal security proofs are as good as their predicates
- To know what matters in a system, find ways to subvert it
- You will learn to build better systems and have fun doing it
- Not just for hackers

Hacking is a microcosm of CS

- Assembly
- Operating systems
- Cryptography
- File formats, databases, protocols, hardware, system tools....
- And your most important skills:
    - Thinking outside the box
    - Learning to learn

The Capture The Flag qualifiers

- Team based
- Jeopardy style questions (first to answer picks the next puzzle)
- Find a secret string to answer a question
- Focus on puzzle solving, not out of the box exploits
- 72 hours of intense hacking

The puzzles

- 5 categories (varies year to year)
  - Forensics
  - Pwnage
  - Reverse Engineering
  - “Real World” (Web, SQL injection, etc.)
  - Trivia

Simple problems...

- Find the key in the metadata of a JPG
- FTP into a server with guest credentials, and find the file with the key
- Figure out what a piece of assembly from an obscure architecture does

Hard problems

- Reconstruct a zip file with a missing table of contents, unzip the OGG file inside, find the PNG embedded in the OGG, turn anomalies in the PNG into ASCII text, look at the resulting ASCII art to read the key.
- Write your first buffer overflow!

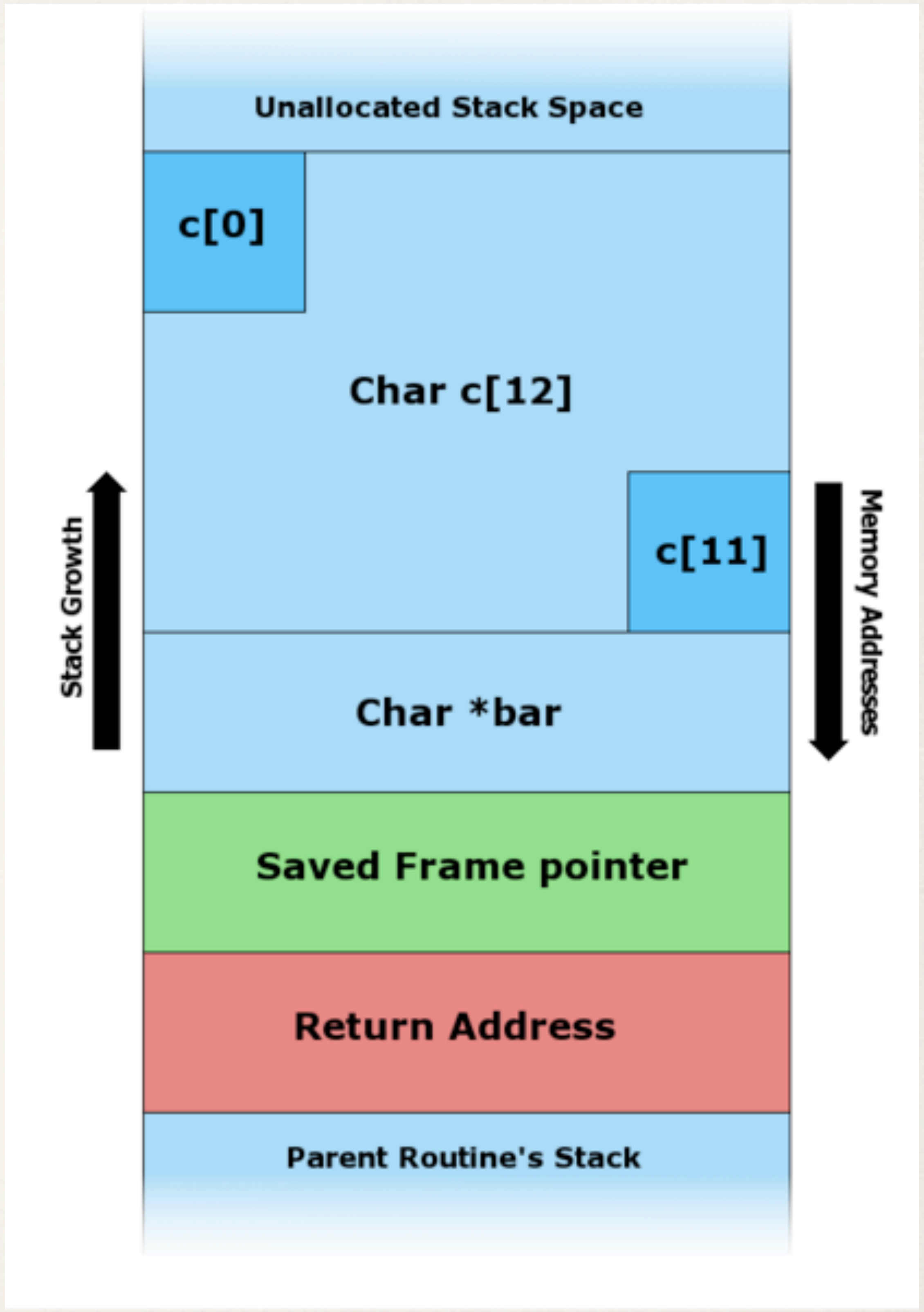
Buffer overflows

Will really test your knowledge of:

- Assembly
- stack versus heap
- C calling conventions
- Standard library functions

Buffer Overflows 101

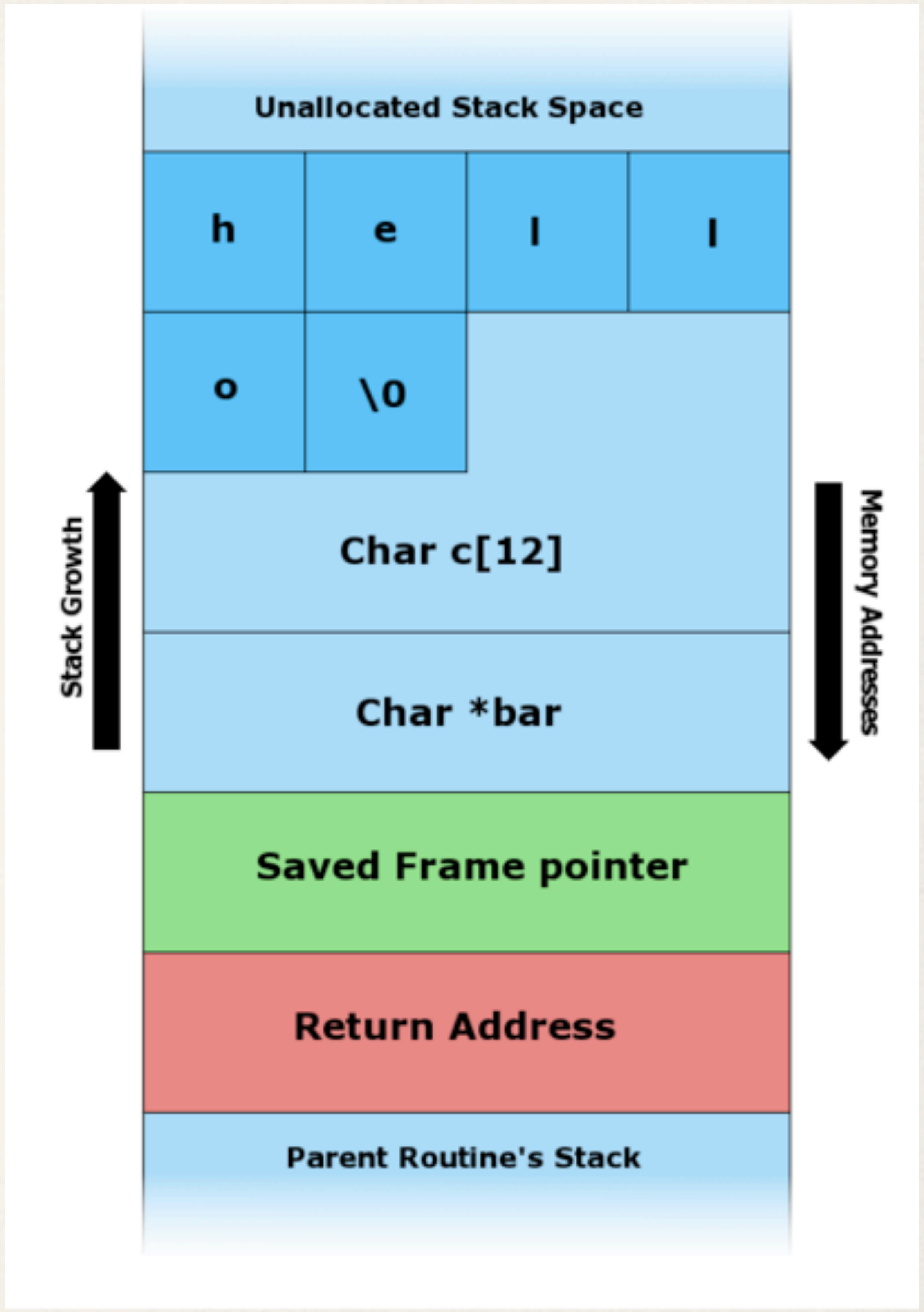
- Stack contains variables, frame pointer, and return address
- Program copies data from user input into c[], without bounds checking.



Images courtesy of Wikimedia

Buffer Overflows 101

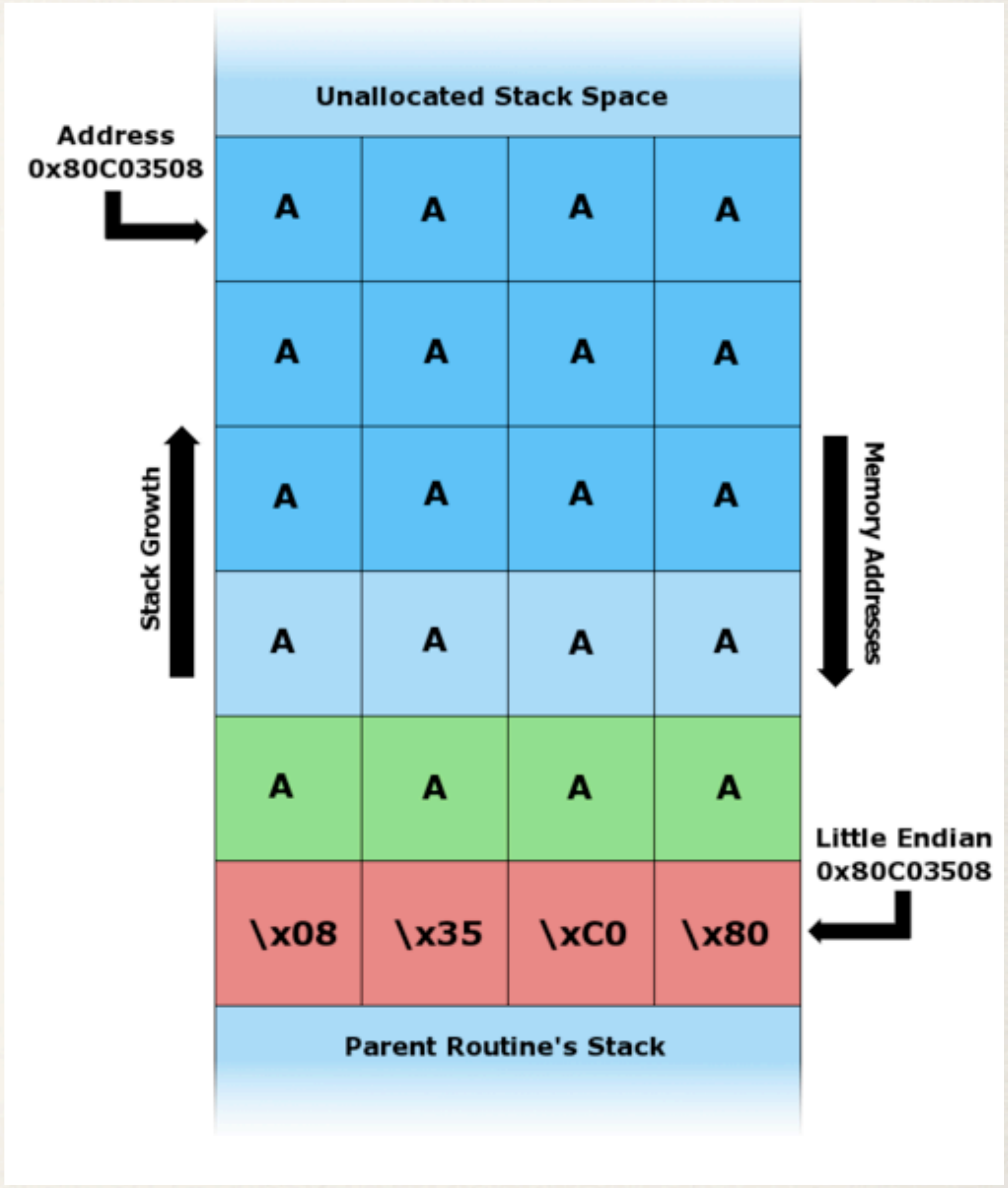
- A friendly user types 'hello'..



Images courtesy of Wikimedia

Buffer Overflows 101

- A less friendly user types ‘A A A A A A A A A A A A A A A A A A A A \x08 \x35 \xC0 \x80’
- Computer jumps to 0x80C03508 and executes A A A A A A A A A A A A A A A A



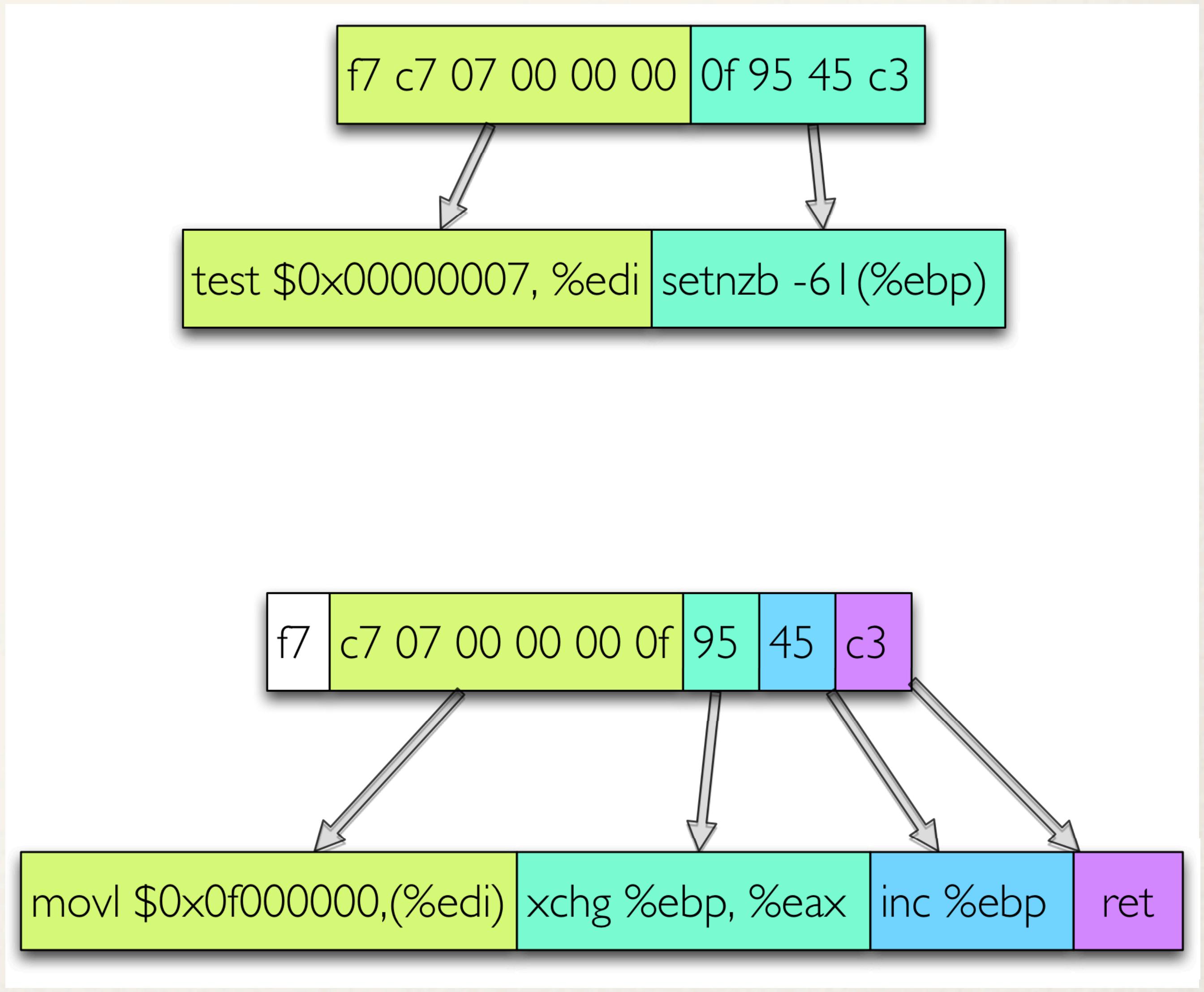
Images courtesy of Wikimedia

More sophisticated....

- Overflow a heap buffer as well, jump into heap and exec there
- Jump into libc and run `system()` with your own arguments
- Set up a series of jumps into shared libraries and use small pieces of code from each

Really sophisticated

- Set up a series of jumps into shared libraries on improper byte alignments
- Use a totally new set of instructions (Shacham '07)



The tip of the iceberg

- SQL injection
- Cross site scripting
- Man in the Middle attacks
- Social engineering
- Anyone can hack; anyone can learn from hacking

The dark side of hacking

- Microcosm of CS means...
    - People in their mom's basement drinking RockStar
    - Crude, rude, and sexist
    - Obsessed with minutia, especially flaws
- A lot of breaking and less building

Exploring Further

- http://www.hackthissite.org
- The Shellcoder's Handbook
- https://legitbs.net/ (This year's CTF quals)
- Questions?