

x86 Assembly and Call Stack

CS 161 Spring 2025 - Lecture 2

Last Time: Security Principles

Computer Science 161

- **Know your threat model:** Understand your attacker and their resources and motivation
- **Consider human factors:** If your system is unusable, it will be unused
- **Security is economics:** Balance the expected cost of security with the expected benefit
- **Detect if you can't prevent:** Security requires not just preventing attacks but detecting and responding to them
- **Defense in depth:** Layer multiple types of defenses
- **Least privilege:** Only grant privileges that are needed for correct functioning, and no more
- **Separation of responsibility:** Consider requiring multiple parties to work together to exercise a privilege
- **Ensure complete mediation:** All access must be monitored and protected, unbypassable
- **Shannon's maxim:** The enemy knows the system
- **Use fail-safe defaults:** Construct systems that fail in a safe state, balancing security and usability.
- **Design in security from the start:** Consider all of these security principles when designing a new system, rather than patching it afterwards

Today

Computer Science 161

- Half CS 61C review
 - How do computers represent numbers as bits and bytes?
 - How do computers interpret and run the programs we write?
 - How do computers organize segments of memory?
- Half new content
 - How does x86 assembly work?
 - How do you call a function in x86?

Number Representation

Textbook Chapter 2.1

Units of Measurement

- In computers, all data is represented as bits
 - **Bit**: a binary digit, 0 or 1
 - 1 **Byte** = a group of 8 bits
- **0b1000100010001000**: 16 bits, or 2 bytes

Hexadecimal

- 4 bits can be represented as 1 hexadecimal digit (base 16)

Binary	Hexadecimal
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7

Binary	Hexadecimal
1000	8
1001	9
1010	A
1011	B
1100	C
1101	D
1110	E
1111	F

Hexadecimal

- The byte `0b11000110` can be written as `0xC6` in hex
- For clarity, we add `0b` in front of bits and `0x` in front of hex

Binary	Hexadecimal
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7

Binary	Hexadecimal
1000	8
1001	9
1010	A
1011	B
1100	C
1101	D
1110	E
1111	F