Lecture 9:
Pointers and Hashing

CSE 29: Systems Programming and Software Tools

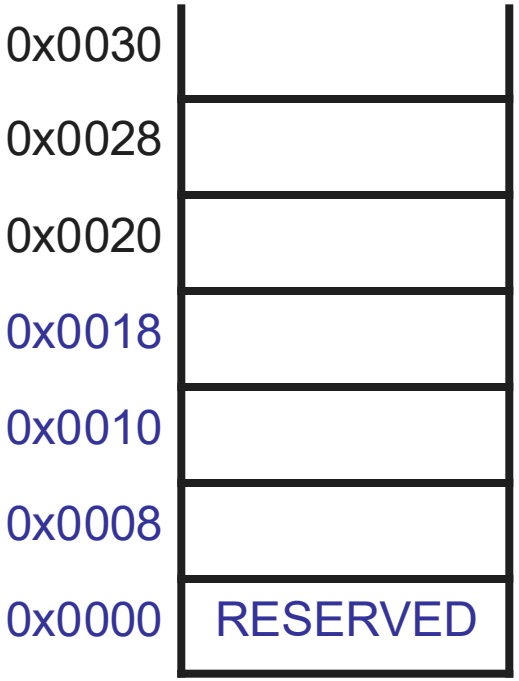
Aaron Schulman (Shalev)

How do you point to “nothing”

int *p = **NULL**;

// C always has this
#define NULL 0

*p = 5;
SEGMENTATION FAULT!!!!



Special C syntax for pointers

& - Address of variable (get addr of var)

```
char* pnum = &num;
```

* - *de*reference address (get var at addr)

```
char num2 = *pnum;
```

We need a way to fingerprint arbitrary bytes

Everything stored in the memory of a computer is just arrays of bytes

- » Strings, numbers, images, videos, etc.

If there was a way to produce an integer as a fingerprint any array of bytes we could:

- **Create a unique digital fingerprint for any unique file/data:**
  - » A (relatively) small integer can uniquely represent any data
  - » We can validate that a file/data has not changed by checking the fingerprint
  - » Often used to check if a downloaded file is correct
- **Application: Store passwords securely**
  - » Applications can store fingerprints of passwords, instead of passwords
  - » These fingerprints can not easily be reversed if the password fingerprint file is compromised

Hash: An integer as a fingerprint for bytes

- Hash: A linear mathematical function that maps bytes to an integer
  - …a really really big integer: 2^{256} (256-bits)
- Hashes are fingerprints that represent those bytes
  - » However, they are one-way functions
  - » The integer that makes up the fingerprint can not be reversed to find the bytes that made it
  - » This makes them particularly useful for use in password security
  - » They must *uniquely* represent those bytes or they are insecure
- There are many hash functions that you can come up with:
  - MD5 [Rivest] (1992) – 128-bits (broken in 2008)
  - SHA-1 [NSA] (1993) – 160-bits (broken in 2017)
  - SHA-256 [NSA] (2001) – 256-bits (still secure as far as we know)