Hash Functions

Hash

- Very useful for cybersecurity
- They allow for more efficient data structures
- A hash function takes an arbitrary length string x and computes a fixed-length string $y = h(x)$
- This output is known by many names: hash, message digest, fingerprint or tag
- For a given out (hash) there will be many (an infinite amount) of inputs that give this hash
- Cryptographic hash function should have three properties
  - **Pre-Image resistance**Given y it is very hard to find the pre-image x of this hash
  - **Second pre-image resistance** given x it is hard to find x' such that $h(x) = h(x')$
  - **Collision resistance** It is hard to find two distinct values x, x' such that $h(x) = h'(x)$
- Collision is easiest to break and pre-image is the hardest
- If you can guarantee collision resistance, this is the strongest as if you break collision resistance you can break the rest
- The perfect hash function
  - A random oracle is a black box that has a pure source of randomness and infinite memory
  - We can instantiate a perfect hash function using a random oracle
  - If we send a new query (input) it generates a random output of the hash length
  - If we send a query we have done in the past it gives us the same output as before
- Breaking a cryptographic hash function
- Imagine we want to find pre-image of a hash function
- We can brute force it by trying random inputs until it gives you the output you are looking for
- If we have b output bits it will take 2^b runs to break the hash
- Finding collision is way easier, finding two inputs that have the same hash is way faster
- Often there are shortcuts to break hash functions

Password protection

- naive method

    - create account
    - server stores username and password
    - log un
    - server checks that the user sent the right password
- Encryption, stored in the server as encrypted passwords: not good enough as if someone hacks the server they can get the key
- Storing hashes is the best method
- To get the password we have to break the hash
- Should the password be hashed on the local computer or server?
- You should hash on the server as if the hash database is leaked then the hacker can just login with the hash in the database
- We salt passwords to make them stronger
- This makes guessing harder as there are user-specific salts in the password fingerprint

Hash in Java

- `Hashcode()` is a method of the Objet class → every class inherits this
    - Different classes can implement `Hashcode()` differently
-