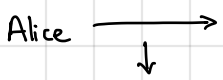
Instructor: David Wu (dwu4@cs.utexas.edu)
TA: Zhiyi Huang

Overarching goal of cryptography: securing communication over untrusted networks



third party should not be able to
1) eavesdrop of communication (confidentiality)
2) tamper with the communication (integrity)

Today: secure communication on web (https://...)
TLS protocol (transport layer security)
two components: handshake (key exchange)
record layer (confidentiality + integrity)
protecting data at rest: disk encryption

Most of this course: study mechanics for protecting confidentiality + data
- Encryption schemes for confidentiality
- Signature schemes for message integrity
- Key exchange for setting up shared secrets

End of this course: protecting communication ⇒ protecting computation
- Two users want to learn a joint function of their private inputs
  - ↳ training models on private (hidden) data
  - ↳ comparing two DNA sequences privately
  - ↳ private auction to determine winner without revealing bids
  - ↳ private voting mechanisms (can identify winner of election without revealing individual votes)
- We can show the following remarkable theorem:
  "Anything that can be computed with a trusted party can be computed without!"

Logistics and administrivia: ← course is primarily a theory course - we will assume familiarity with reductions and mathematical proofs!
- Course website: https://www.cs.utexas.edu/~dwu4/courses/fa24
- See Ed Discussion for announcements, notes will be posted to course website (1-2 days after lecture)
- Homework submission via Gradescope (enroll via Canvas) ← one of these is programming assignment (Python)
- Course consists of 5 homework assignments (worth 70%) and two in-class exams (worth 30%)
- Five late days for the semester: use in 24-hour increments, max 72 hours (3 late days) for any single assignment
- Some office hours will also be available on Zoom

This semester: Lectures will be recorded using Lectures Online
Please participate virtually if you are feeling unwell

A brief history of cryptography:

Original goal was to protect communication (in times of war)

Basic idea: Alice and Bob have a shared key k

Alice computes $c \leftarrow \text{Encrypt}(k, m)$

(c: ciphertext; k: key; m: message (plaintext))

Bob computes $m \leftarrow \text{Decrypt}(k, c)$ to recover the message

This tuple (Encrypt, Decrypt) is called a cipher

K, M, C are sets (e.g., $K = M = C = \{0,1\}^{128}$)

Definition. A cipher is defined over (K, M, C) where K is a key-space, M is a message space and C is a ciphertext space, and consists of two algorithms (Encrypt, Decrypt):

$\text{Encrypt} : K \times M \to C$

$\text{Decrypt} : K \times C \to M$

} functions should be "efficiently-computable"

theory: runs in probabilistic polynomial time [algorithm can be randomized]

practice: fast on an actual computer (e.g., < 10ms on my laptop)

Correctness: $\forall k \in K, \forall m \in M$:

$$\text{Decrypt}(k, \text{Encrypt}(k, m)) = m$$

"decrypting a ciphertext recovers the original message"

Early ciphers:

- Caesar cipher: "shift by 3"

A	↦ D
B	↦ E
C	↦ F
⋮	
X	↦ A
Y	↦ B
Z	↦ C

Not a cipher! There is no key!

Anyone can decrypt!

↳ Algorithm to encrypt is assumed to be public.

NEVER RELY ON SECURITY BY OBSCURITY!

- Harder to change system than a key
- Less scrutiny for secret algorithms

- Caesar cipher++: "shift by k" ($k = 13$: ROT-13)

k is the key

↳ Still totally broken since there are only 26 possible keys (simply via brute force guessing)

- Substitution cipher: the key defines a permutation of the alphabet (i.e., substitution)

A	↦ C
B	↦ X
C	↦ J
⋮	
Z	↦ T

ABC ↦ CXJ

← substitution table is the key

How many keys? For English alphabet, $26! \approx 2^{88}$ possible keys

↑ very large value, cannot brute force the key

Still broken by frequency analysis
- e is the most frequent character (~12%)
- q is the least frequent character (~0.10%)

Can also look at digram, trigram frequencies

- Vigenere cipher (late 1500s) — "polyalphabetic substitution"

key is short phrase (used to determine substitution table):

m = HELLO

k = CAT

Encrypt(k, m):

```
  HELLO
+ CATCA   <- repeat the key
-------
  KFFPP
```

↑ interpret letters as number between 1 and 26
addition is modulo 26

if we know the key length, can break using frequency analysis
otherwise, can try all possible key lengths $l = 1, 2, \ldots$

↳ general assumption: keys will be much shorter than the message (otherwise if we have a good mechanism to deliver long keys securely, then can use that mechanism to share messages directly)

- Fancier substitution ciphers: Enigma (based on rotor machines)

but... still breakable by frequency analysis

Today: encryption done using computers, lots of different ciphers
- AES (advanced encryption standard, 2000) "block cipher"
- Salsa (2005) / ChaCha (2008) "stream cipher"