CIS 6930:
IoT Security

Lecture 2

Prof. Kaushal Kafle

Spring 2025

Derived from slides by Adwait Nadkarni, William Enck, Micah Sherr and Patrick McDaniel

Class Notes and Clarifications

- High-level Topics (and goals):
  - Basics of crypto *(this isn't a crypto course)*
  - Information Flow Control
  - Web and Network Security
  - IoT Security
  - Engineering/research trade-offs
  - **How to read/write/present security research papers**
- Check the syllabus!
- **Note:** I reserve the right to adapt the syllabus throughout the semester… *but I will provide sufficient notice for any changes*

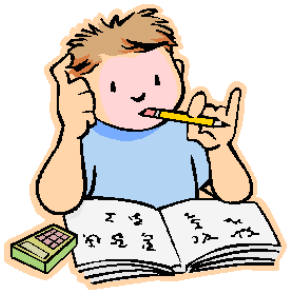
Non Goals

- Familiarization with the latest tools
- Professional Security Certification

Let us begin..

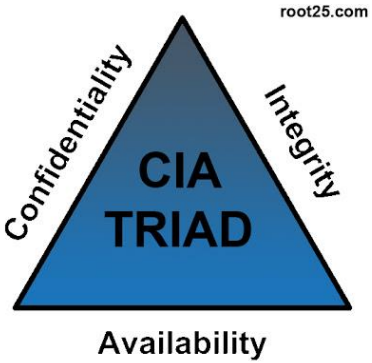
What is security?

- Garfinkel and Spafford (1991)
  - "A computer is secure if you can depend on it and its software to behave as expected."
- Harrison, Ruzzo, Ullman (1978)
  - "Prevent access by unauthorized users"
- Not really satisfactory – does not truly capture that security speaks to the behavior of others
  - Expected by whom?
  - Under what circumstances?



Security Goals

- *Confidentiality*: Prevention of unauthorized disclosure of information
- *Integrity*: Prevention of unauthorized modification of information
- *Availability*: Prevention of unauthorized withholding of information or resources



Brinkley and Schell, "Concepts and Terminology for Computer Security."

Security Goals (continued)

- *Authenticity*: Related to integrity, but also speaks to the *sender*, as well as *freshness*
- *Secrecy*: Similar to confidentiality, but often used when discussing specific mechanisms, e.g., access control
- *Non-repudiation*: Prevent a party from denying that some action took place e.g., signed through private key, HMACs
- *Privacy*: The ability/right to control access to one's information. There are many definitions. Often conflated with confidentiality/secrecy.

Risk

- *Assets* are valued resources that can be misused
  - Monetary, data (loss or integrity), time, confidence, trust
- *Risk* is the potential for an asset to be misused
  - Many different formulas, e.g., (Risk = likelihood * impact)
  - What does being misused mean?
    - Privacy (personal)
    - Confidentiality (communication)
    - Integrity (personal or communication)
    - Availability (existential or fidelity)

Q: What about a real-world system, say banking?

Threats

- A *threat* is a specific means by which an attacker can put a system at risk
  - An ability/goal of an attacker (e.g., eavesdrop , fraud, access denial)
  - Independent of what can be compromised
- A *threat model* is a collection of threats that deemed important for a particular environment
  - A collection of attacker(s) abilities
  - E.g., A powerful attacker can read and modify all communications and generate messages on a communication channel

Vulnerabilities (attack vectors)

- A *vulnerability* is a systematic artifact that exposes the user, data, or system to a threat
- E.g., buffer-overflow, WEP key leakage
- What is the source of a vulnerability?
  - Bad software (or hardware)
  - Bad design, requirements
  - Bad policy/configuration
  - System Misuse
  - Unintended purpose or environment
    - E.g., student IDs for liquor store

Adversary

- An *adversary* is any entity trying to circumvent the security infrastructure (sometimes called *attacker*)
  - The curious and otherwise generally clueless (e.g., script-kiddies)
  - Casual attackers seeking to understand systems
  - Venal people with an ax to grind
  - Malicious groups of largely sophisticated users (e.g, chaos clubs)
  - Competitors (industrial espionage)
  - Governments (seeking to monitor activities)

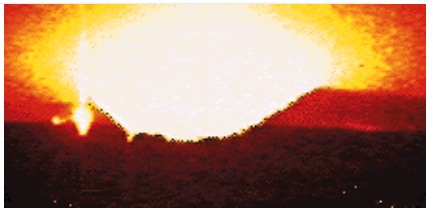
Are users adversaries?

This is known as the insider adversary!

- Have you ever tried to circumvent the security of a system you were authorized to access?
- Have you ever violated a security policy (knowingly or through carelessness)?

Attacks

- An attack occurs when someone attempts to exploit a vulnerability
- Kinds of attacks
  - **Passive** (e.g., eavesdropping)
  - **Active** (e.g., password guessing)
  - **Denial of Service** (DOS)
    - Distributed DOS – using many endpoints



- A compromise occurs when an attack is successful
  - Typically associated with taking over/altering resources

Participants

- *Participants* are expected system entities
  - Computers, agents, people, enterprises, …
  - Depending on context referred to as: servers, clients, users, entities, hosts, routers, …
  - Security is defined with respect to these entites
    - Implication: every party may have unique view
- A *trusted third party*
  - Trusted by all parties for some set of actions
  - Often used as introducer or arbiter

Q: Example of a trusted third party?

Trust

- *Trust* refers to the degree to which an entity is expected to behave
- What the entity not expected to do?
  - E.g., not expose password
- What the entity is expected to do (obligations)?
  - E.g., obtain permission, refresh
- A *trust model* describes, for a particular environment, who is trusted to do what?
- Note: you make trust decisions every day
  - Q: What are they?
  - Q: Whom do you trust?

Trusted vs. Trustworthy

- *Trusted*: a trusted system or component is one whose failure can break the security policy
- *Trustworthy*: a trusted system or component is one that won't fail

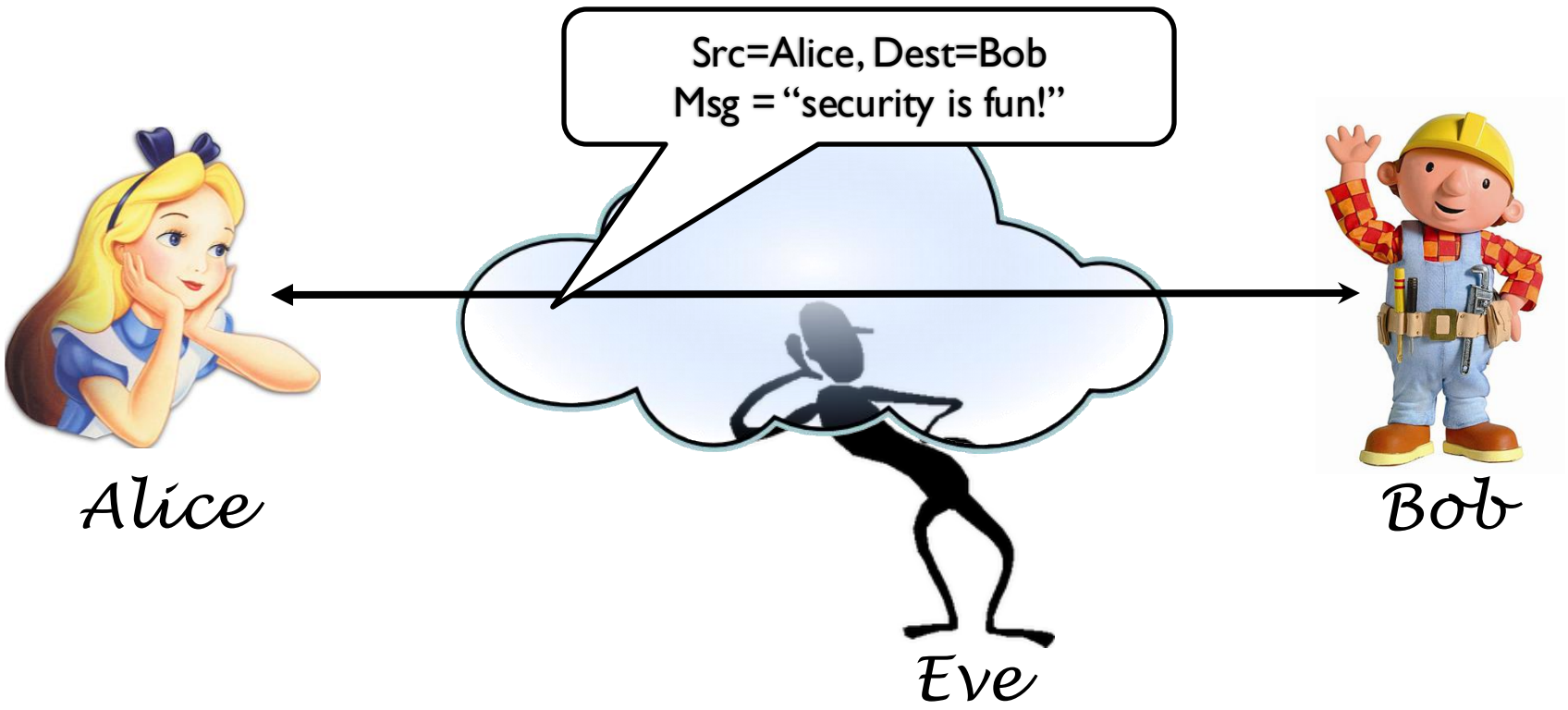
Anderson, "Security Engineering."

Security Model

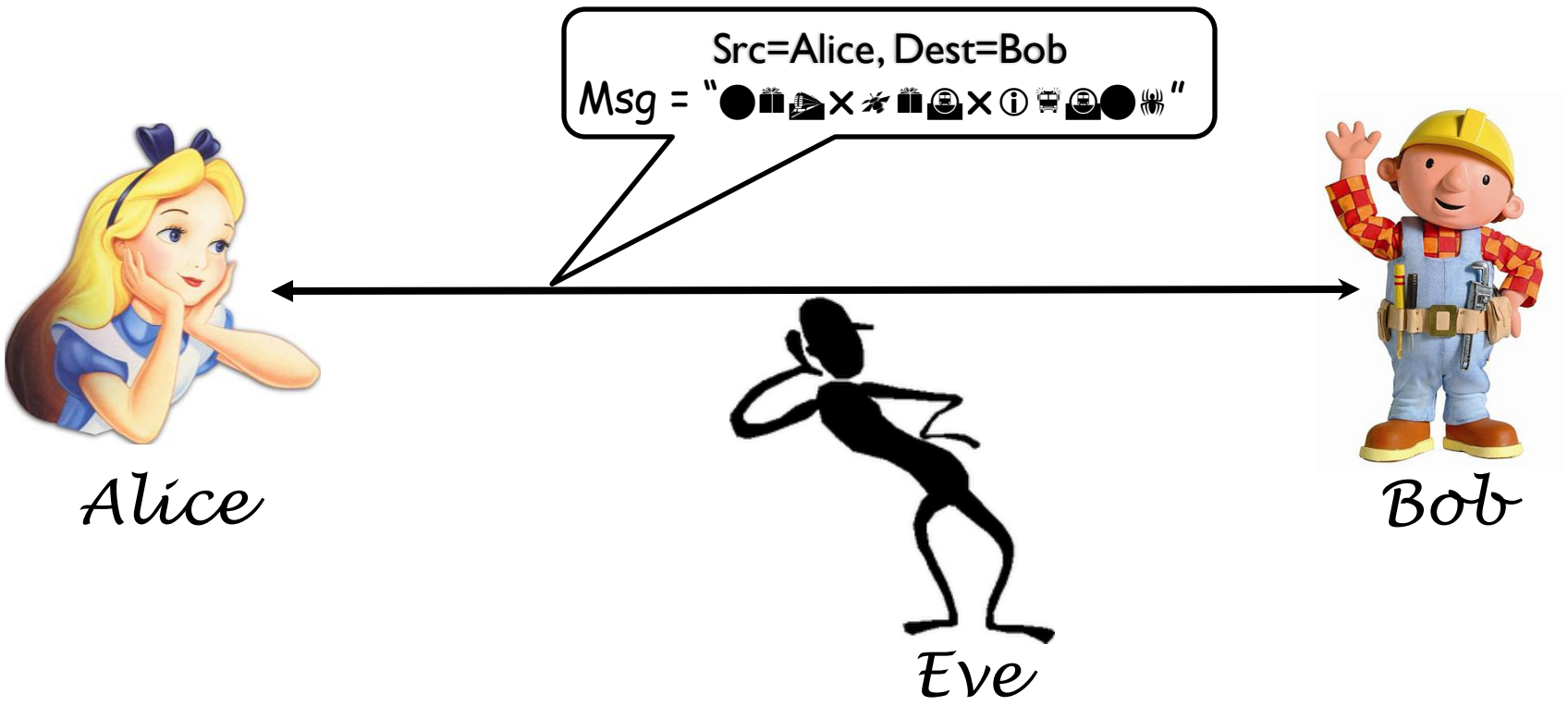
- A *security model* is the combination of a trust and threat models that address the set of perceived risks
  - The "security requirements" used to develop some cogent and comprehensive design
  - Every design must have security model
    - LAN network or global information system
    - Java applet or operating system
- The single biggest mistake seen in use of security is the lack of a coherent security model
  - It is very hard to retrofit security (design time)
- This class is going to talk a lot about security models
  - What are the security concerns (risks)?
  - What are the threats?
  - Who are our adversaries?
  - Who do we trust and to do what?
- Systems must be explicit about these things to be secure.

Let’s look at some potentially desirable properties of a secure network system...

Meet the players.

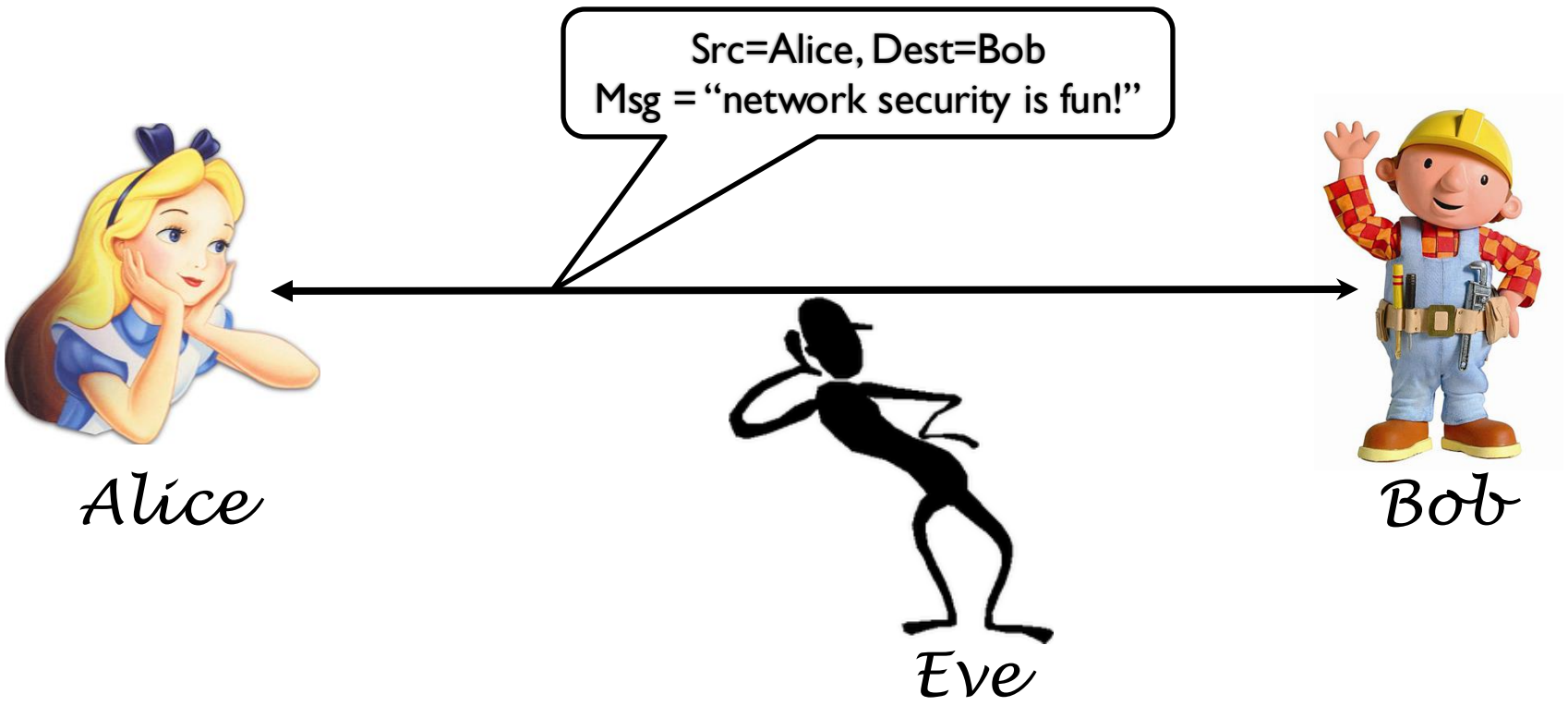


Confidentiality



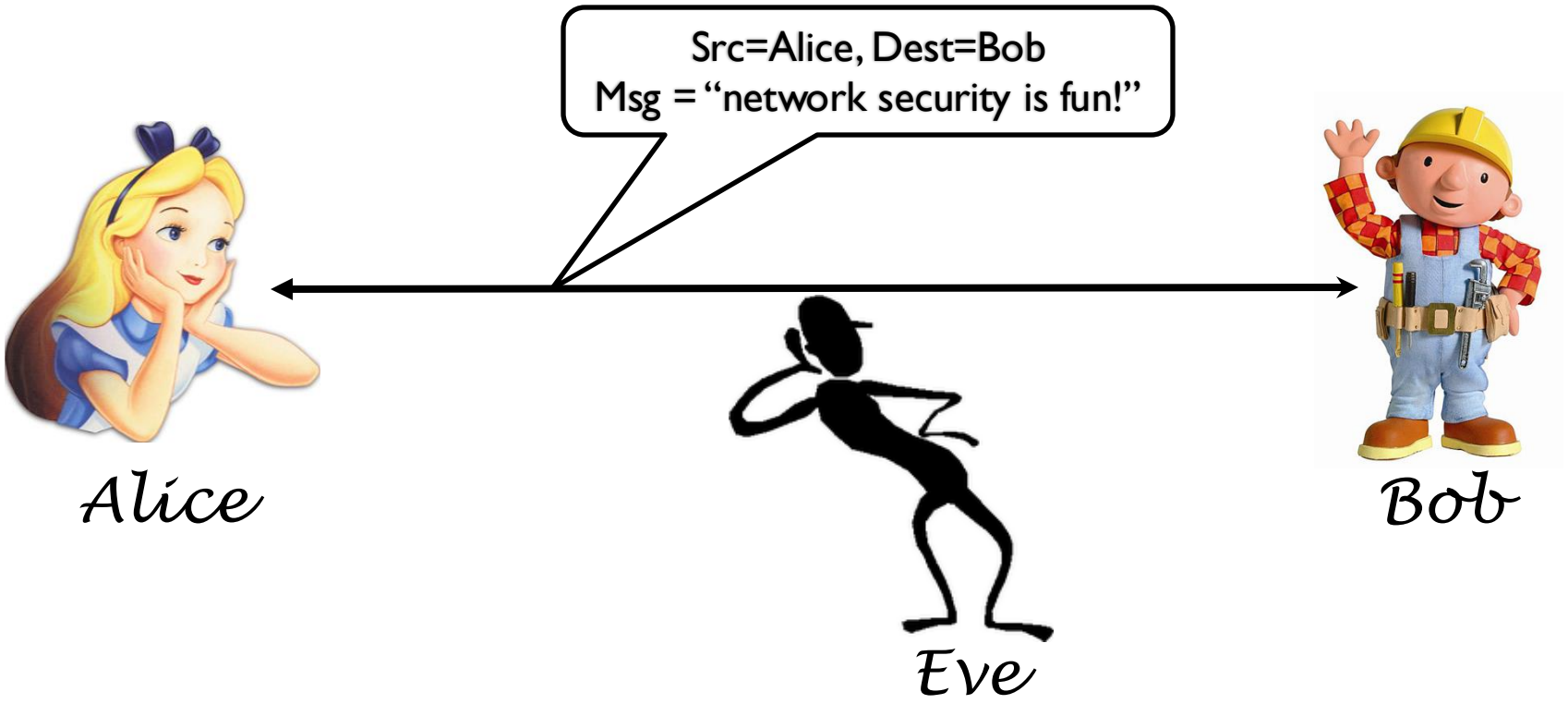
Alice and Bob want to communicate privately, preventing Eve from learning the contents of their communication

Integrity



Bob wants to verify that the message hasn't been altered in transit.

Authentication

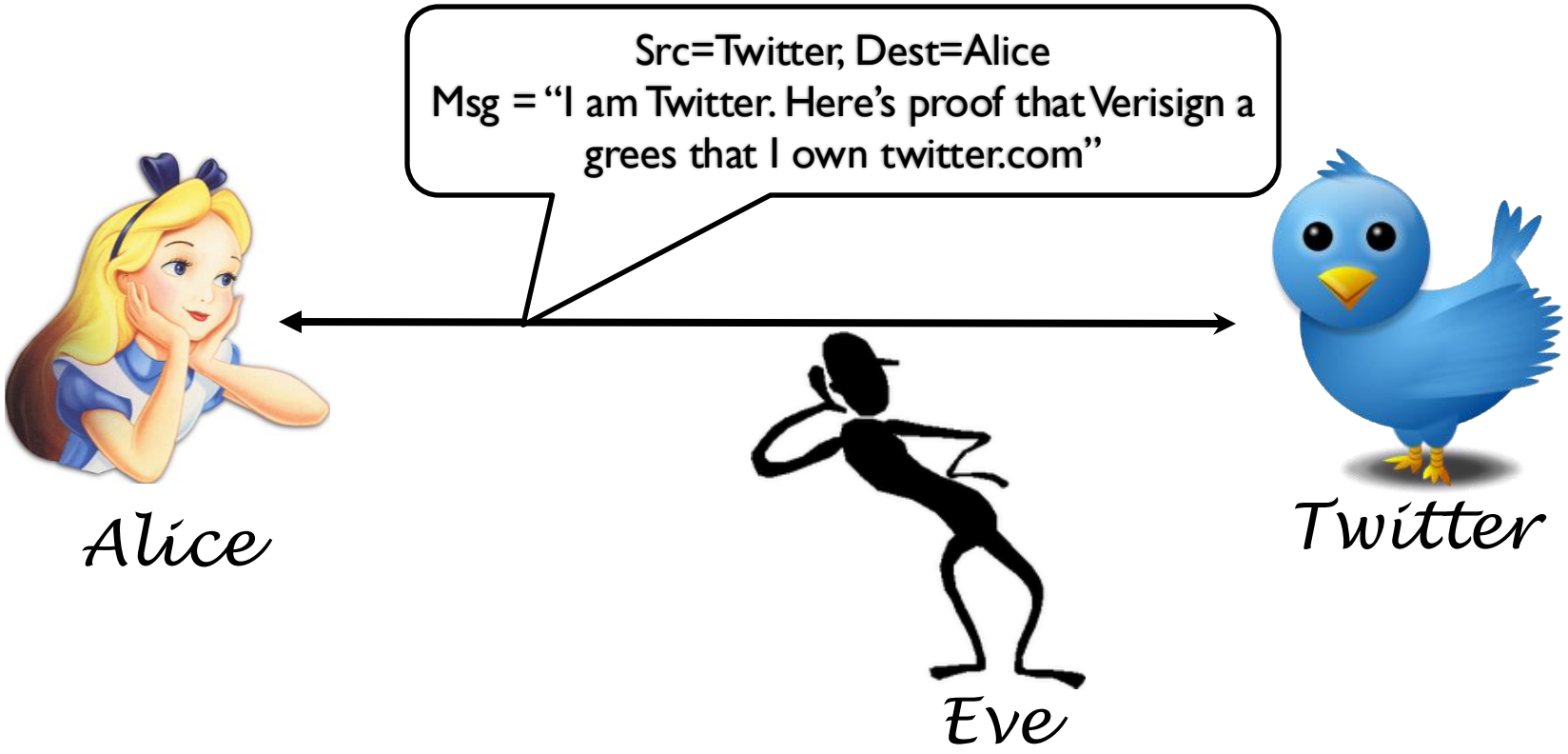


Bob wants to verify that the message is actually from Alice.

Client authentication

Alice wants to prove her identity to the service.

Server authentication



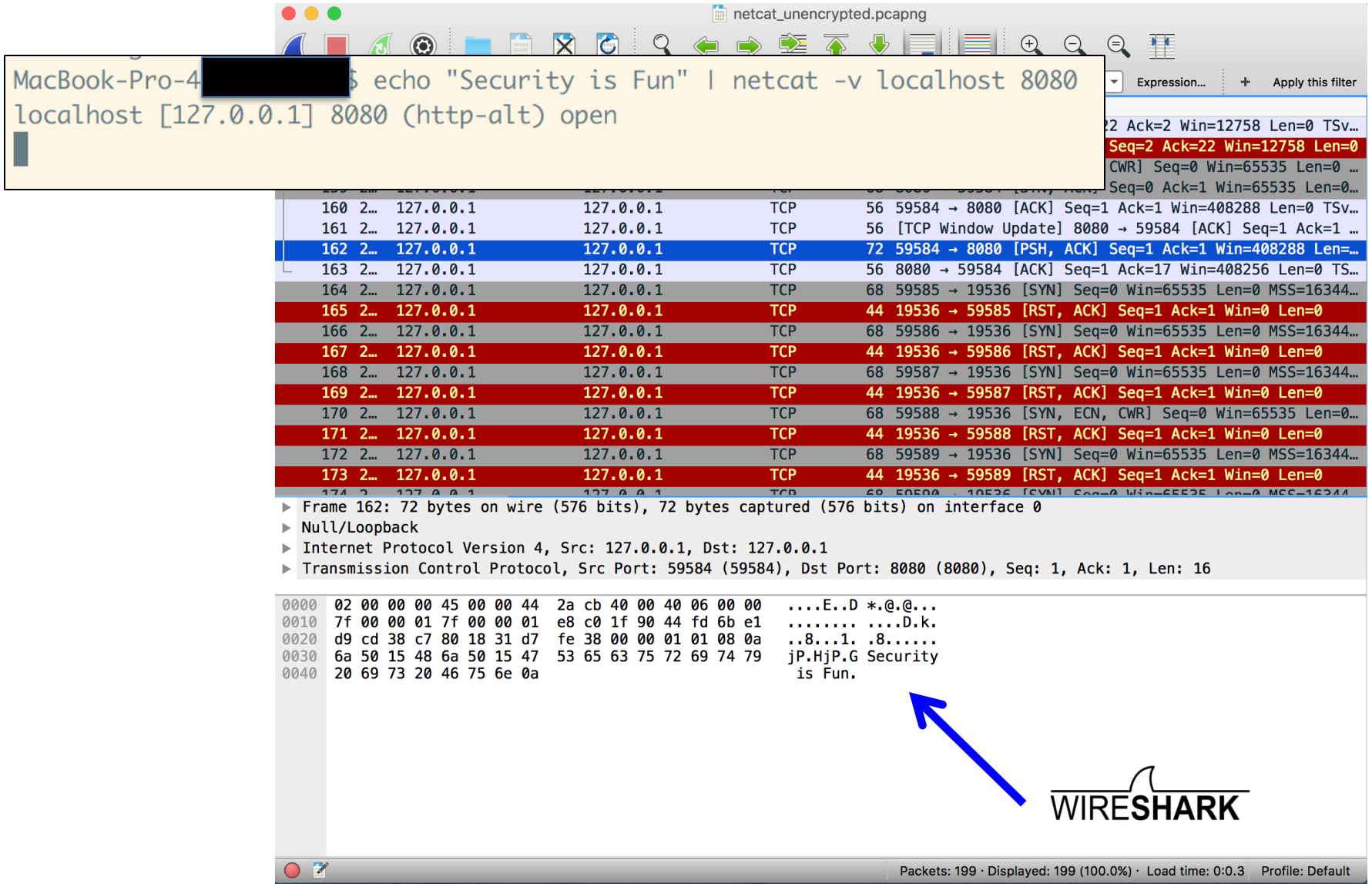
The service wants to prove its identity to Alice.

Cryptography

Crypto in IoT Apps

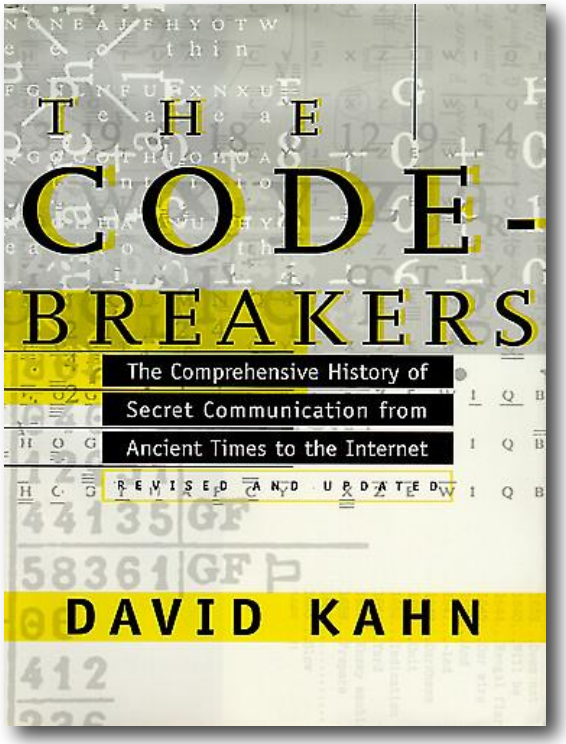
- Networks designed for data transport, *not for data confidentiality or privacy*
  - Internet eavesdropping is (relatively) easy
- Sensitive data is often stored locally on the device.
  - Other apps/root can get to it.
- Where have you seen crypto in practice?
- Crypto enables:
  - e-commerce and e-banking
  - confidential messaging
  - data transfer between IoT devices and cloud
  - protection of personal data
  - …

Why is crypto useful?



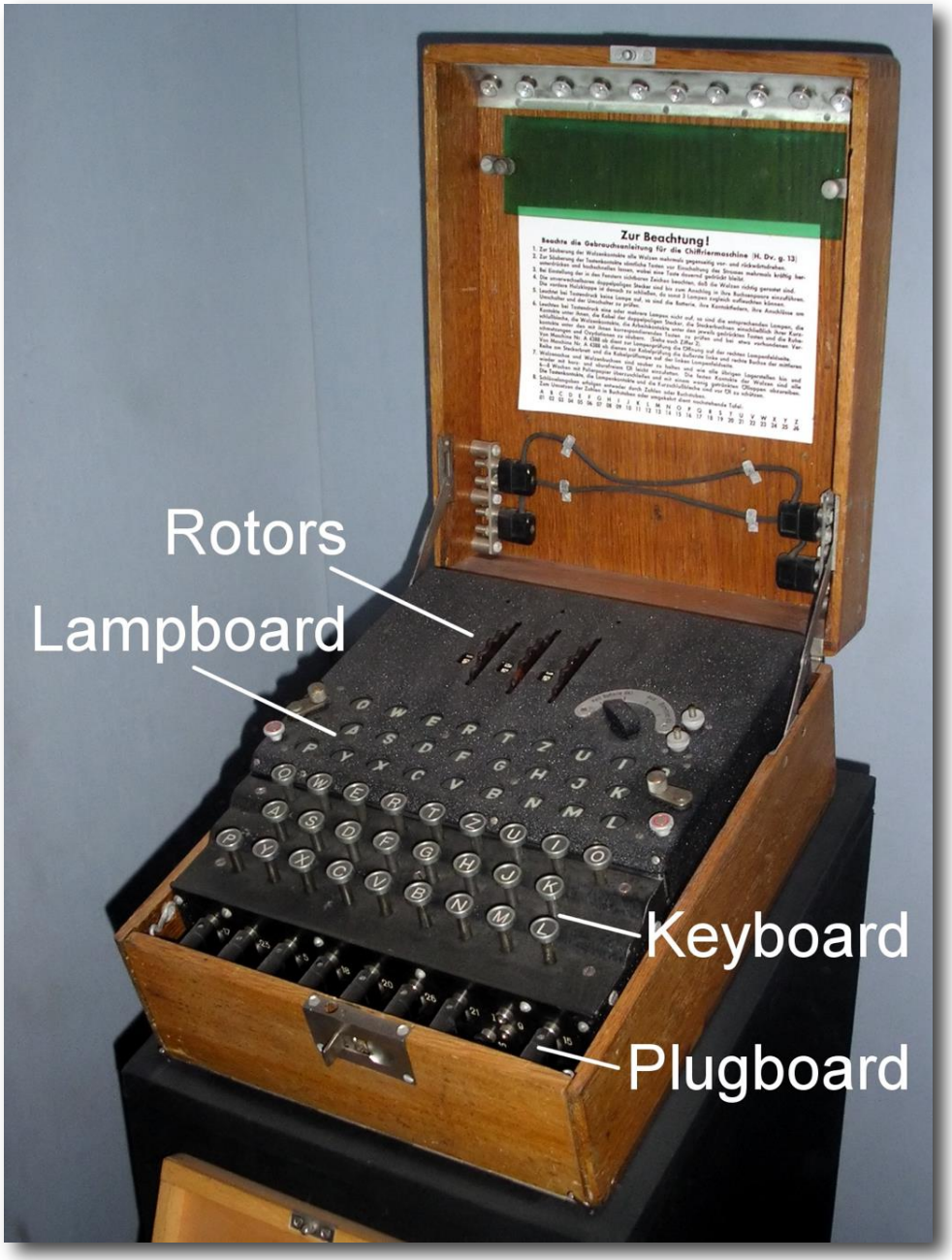
Cryptographic History

- hide secrets from your enemy
- ~4000 year old discipline
  - Egyptians' use of non-standard hieroglyphics
  - Spartans used *scytale* to perform transposition cipher
  - Italian Leon Battista Alberti ("founder of western cryptography") invents polyalphabetic ciphers in 1466



Enigma

- German WWII encryption device
- Used polyalphabetic substitution cipher
- Broken by Allied forces
- Intelligence called Ultra
- Codebreaking at Bletchley Park
- See original at the International Spy Museum at DC



Some terminology

- **cryptosystem**: method of disguising (encrypting) plaintext messages so that only select parties can decipher (decrypt) the ciphertext
- **cryptography**: the art/science of developing and using cryptosystems
- **cryptanalysis**: the art/science of breaking cryptosystems
- **cryptology**: the combined study of cryptography and cryptanalysis

What can crypto do?

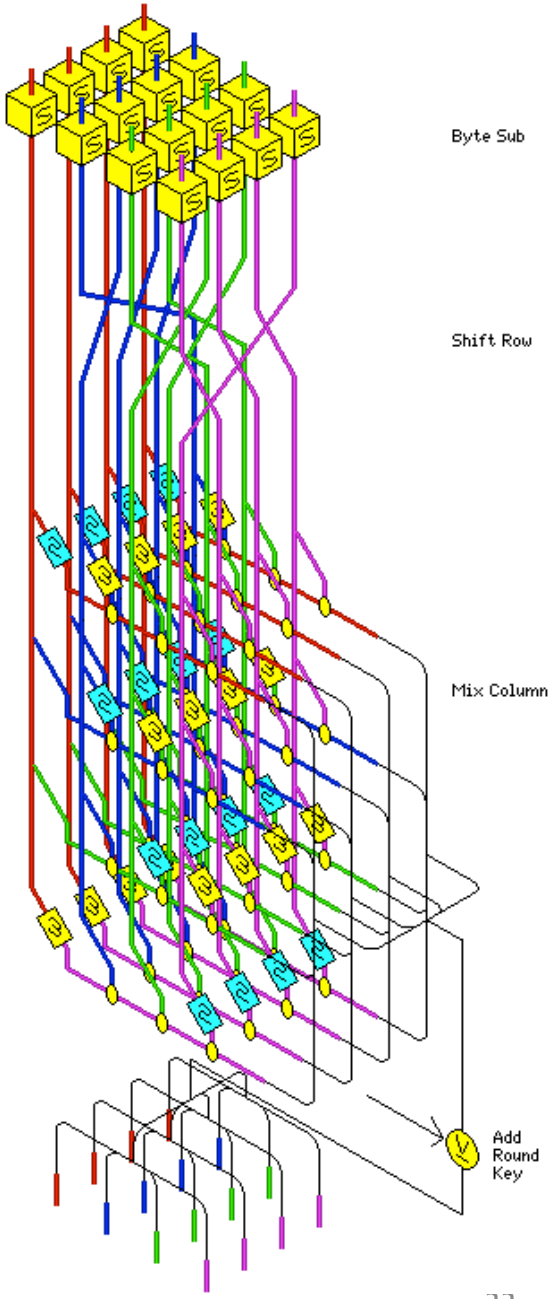
- **Confidentiality**
  - Keep data and communication secret
  - Encryption / decryption
- **Integrity**
  - Protect reliability of data against tampering
  - “Was this the original message that was sent?”
- **Authenticity**
  - Provide evidence that data/messages are from their purported originators
  - “Did Alice really send this message?”

cryptography < security

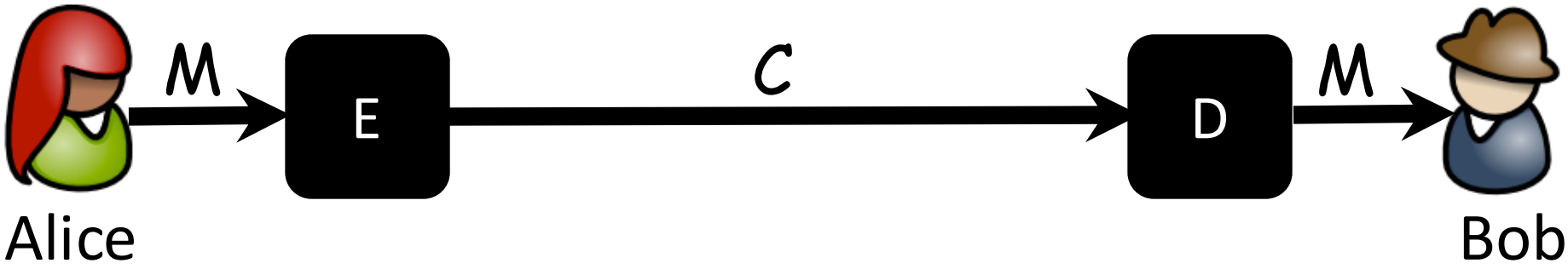
- Cryptography isn't the solution to security
  - Buffer overflows, worms, viruses, trojan horses, SQL injection attacks, cross-site scripting, bad programming practices, etc.
- It's a tool, not a solution
- **It is difficult to get right: choices.. choices….**
  - Choice of encryption algorithms (many tradeoffs)
  - Choice of parameters (key size, IV, ...)
  - Implementation (std. libraries work in most cases)
  - Hard to detect errors
    - Even when crypto fails, the program may still work
    - May not learn about crypto problems until after they've been exploited

Crypto is really, really, really, really, hard

- Task: develop a cryptosystem that is secure against all conceivable (and inconceivable) attacks, and will be for the foreseeable future
- If you are inventing your own crypto, you're doing it wrong
- Common security idiom: "no one ever got fired for using AES"



Encryption and Decryption



C=E(M)
M=D(C)
i.e.,
M=D(E(M))

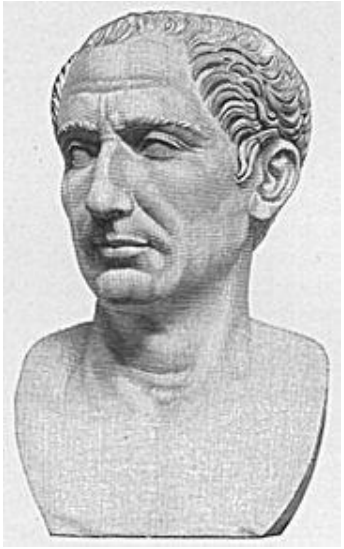
where

M = plaintext
C = ciphertext
E(x) = encryption function
D(y) = decryption function

What's missing?

Let’s look at some old crypto algorithms (don’t use these)

Caesar Cipher



- A.K.A. Shift Cipher or ROT-x cipher (e.g., ROT-13)
- Used by Julius to communicate with his generals
- x is the key:
- Encryption: Right-shift every character by x: c = E(x, p) = (p + x) mod 26
- Decryption: Left-shift every character by x: p = D(x, c) = (c – x) mod 26

S E C U R I T Y A N D P R I V A C Y

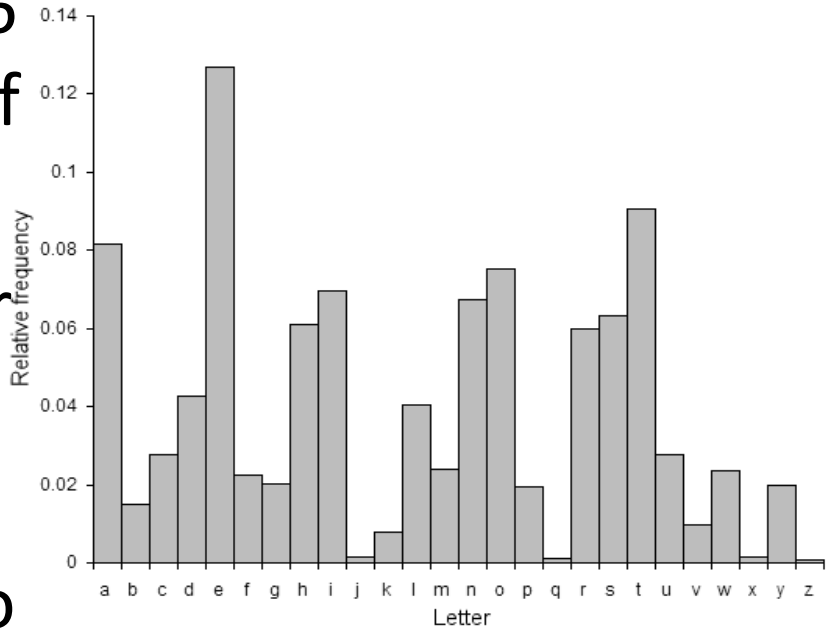
V H F X U L W B D Q G S U L Y D F B

Cryptanalyze this …

“GUVF VF N TERNG PYNFF”

Cryptanalyzing the Caesar Cipher

- Cryptanalysis:
  - **Brute-force attack:** try all 26 possible shifts (i.e., values of x)
  - **Frequency analysis:** look for frequencies of characters
  - Also, same plaintext (repetitions) *always* leads to same ciphertext, since *monoalphabetic*



Polyaphebetic Cipher

- Improves on the simple monoalphabetic ciphers by using multiple monoalphabetic subsitutions
- Example: Vigenère Cipher
  - A set of Caesar Ciphers where each cipher is denoted by a key letter that designates the shift
  - The key repeats for the length of the message

key: deceptivedeceptivedeceptive
plaintext: wearediscoveredsaveyourself
ciphertext: ZICVTWQNGRZGVTWAVZHCQYGLMGJ

One-time Pads

- To produce ciphertext, XOR the plaintext with the **one-time pad** (secret key)
  - E(M) = M $\oplus$ Pad
  - D(E(M)) = E(M) $\oplus$ Pad
- Requires sizeof(pad) == sizeof(plaintext)

- Offers **perfect secrecy**:
  - *a posteriori* probability of guessing plaintext given ciphertext equals the *a priori* probability
  - given a ciphertext without the pad, any plaintext of same length is possible input (there exists a corresponding pad)
  - Pr[M=m|C=c] = Pr[M=m] (you learn nothing from the ciphertext)

- **Never reuse the pad (hence "one-time")!** Why not?

XOR properties

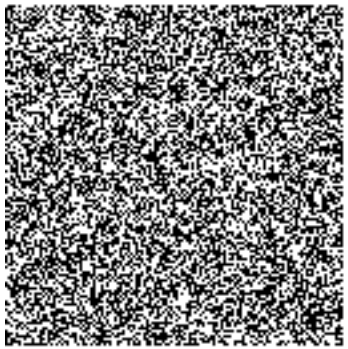
- $A \oplus A = ?$
  - 0
- $A \oplus 0 = ?$
  - A

- $C1 = M1 \oplus Pad,\ C2 = M2 \oplus Pad$
- $C1 \oplus C2 = ?$

$M1 \oplus M2!$

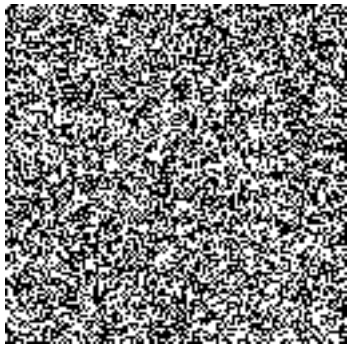
M1

⊕



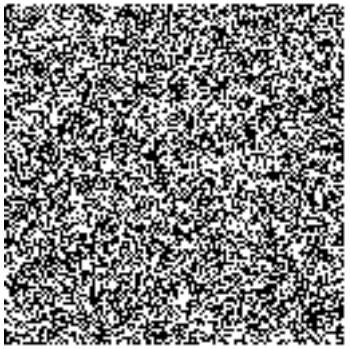
=

C1



M2

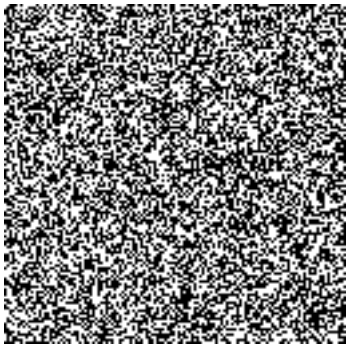
⊕



SAME PAD

=

C2



⊕

=

https://cryptosmith.com/2008/05/31/stream-reuse/