CSE 484 / CSE M 584: Computer Security and Privacy

# Cryptography [Intro]

Spring 2020

Franziska (Franzi) Roesner
franzi@cs.washington.edu

Thanks to Dan Boneh, Dieter Gollmann, Dan Halperin, Yoshi Kohno, Ada Lerner, John Manferdelli, John Mitchell, Vitaly Shmatikov, Bennet Yee, and many others for sample slides and materials ...

# Admin

- Lab 1: Really, seriously make sure you can access it now!
- Research readings: OK to spill onto 2nd page
- Zoom chat:
  - Please keep to direct questions/answers to limit distractions
  - But don't stop other conversations and feedback! → Discussion board, email, office hours, direct chat messages…

# Software Security: More on What To Do

# General Principles

- Check inputs
- Check all return values
- Least privilege
- Securely clear memory (passwords, keys, etc.)
- Failsafe defaults
- Defense in depth
  - Also: prevent, detect, respond

- NOT: security through obscurity

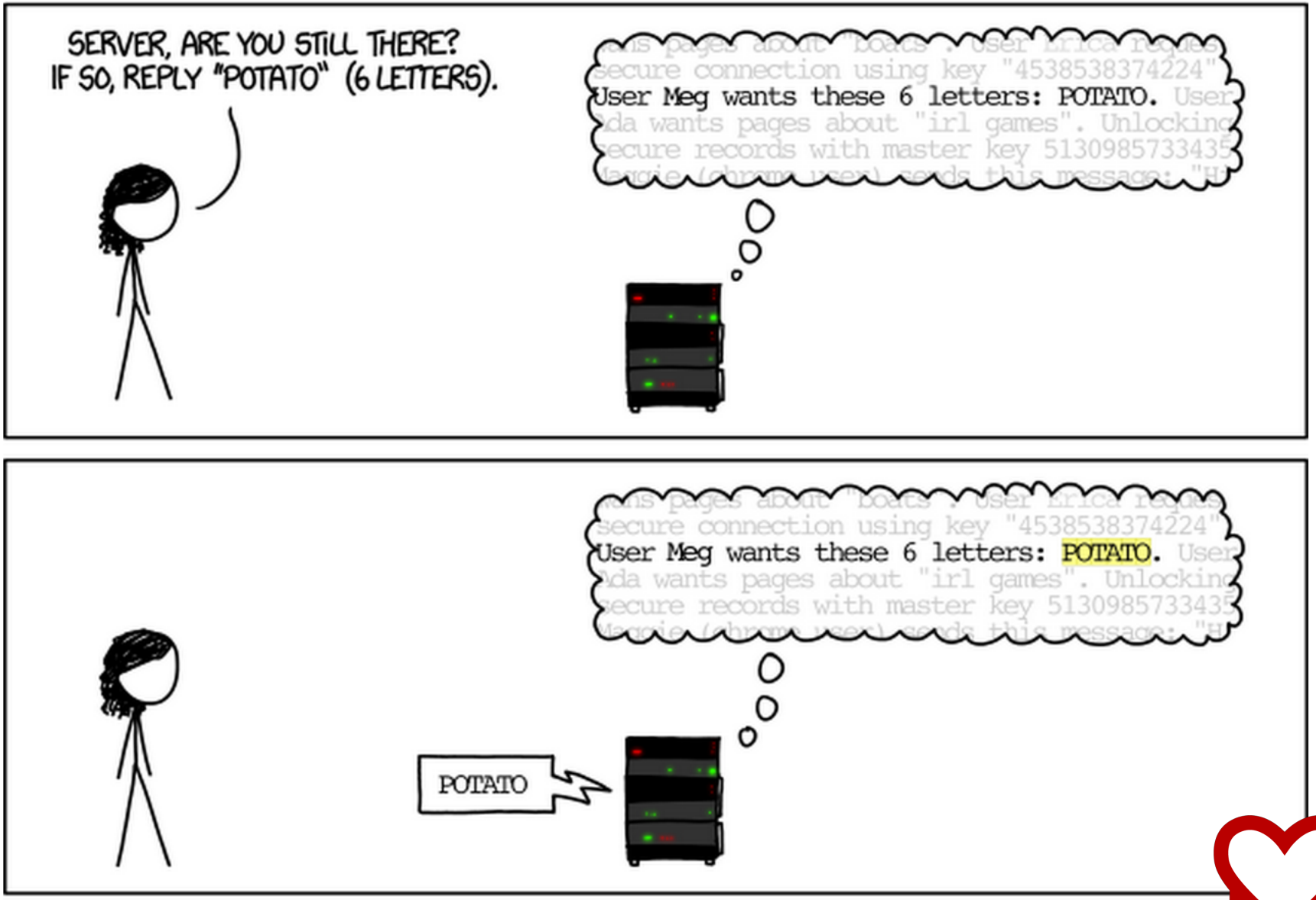
# General Principles

- Reduce size of trusted computing base (TCB)
- Simplicity, modularity
  - But: Be careful at interface boundaries!
- Minimize attack surface
- Use vetted components
- Security by design
  - But: tension between security and other goals
- Open design? Open source? Closed source?
  - Different perspectives

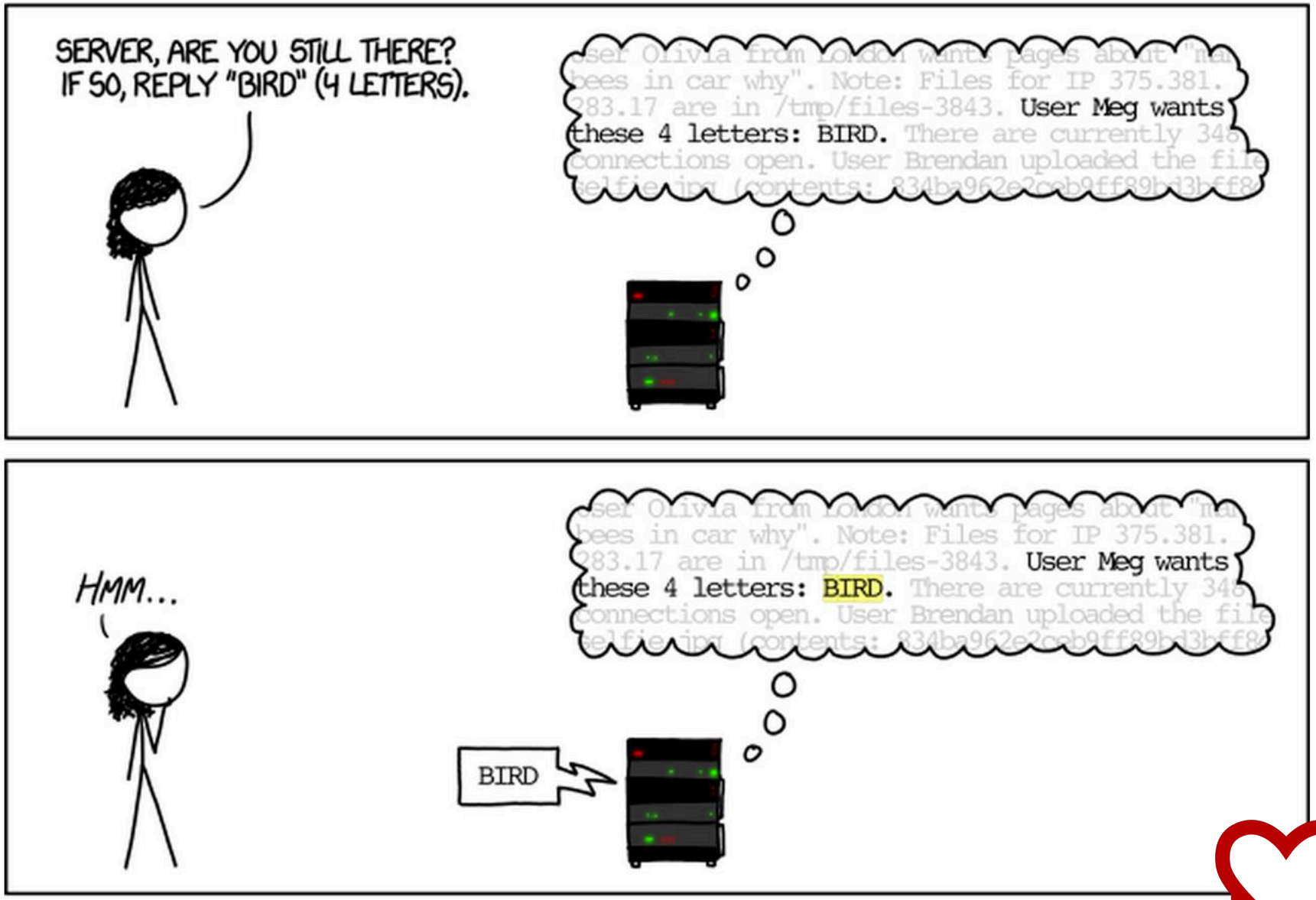
# Does Open Source Help?

- Different perspectives…
- Happy example?
  - Linux kernel backdoor attempt thwarted (2003)
    (http://www.freedom-to-tinker.com/?p=472)
- Sad example?
  - Heartbleed (2014)
    - Vulnerability in OpenSSL that allowed attackers to read arbitrary memory from vulnerable servers (including private keys)

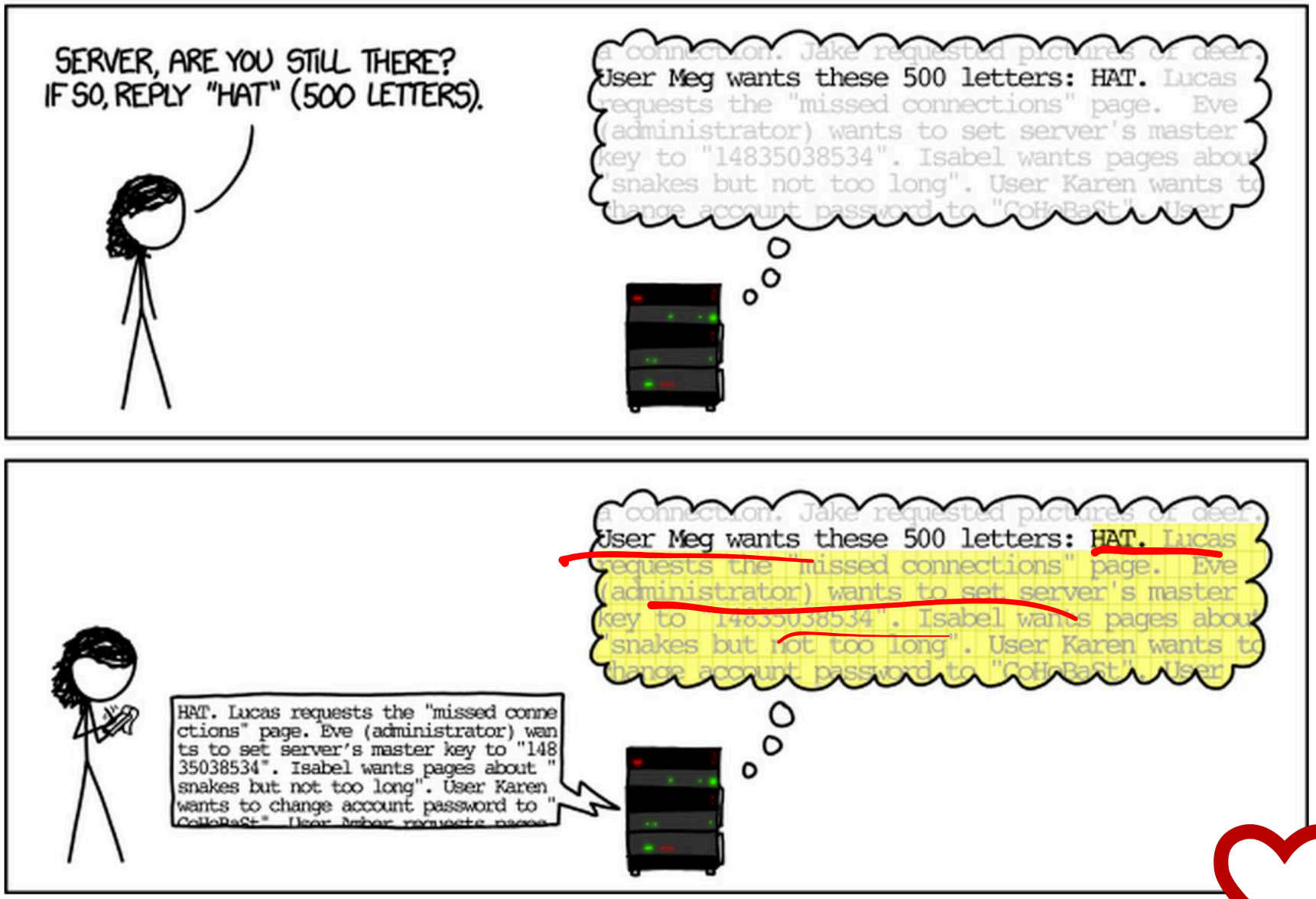
http://xkcd.com/1354/



http://xkcd.com/1354/



http://xkcd.com/1354/



# Vulnerability Analysis and Disclosure

- What do you do if you've found a security problem in a real system?
- Say
  - A commercial website?
  - UW grade database?
  - Boeing 787?
  - TSA procedures?

# Vulnerability Analysis and Disclosure

- Suppose companies A, B, and C all have a vulnerability, but have not made the existence of that vulnerability public
- Company A has a software update prepared and ready to go that, once shipped, will fix the vulnerability; but B and C are still working on developing a patch for the vulnerability
- Company A learns that attackers are exploiting this vulnerability in the wild
- *Should Company A release their patch, even if doing so means that the vulnerability now becomes public and other actors can start exploiting Companies B and C?*
- *Or should Company A wait until Companies B and C have patches?*

# Next Major Section of the Course: Cryptography

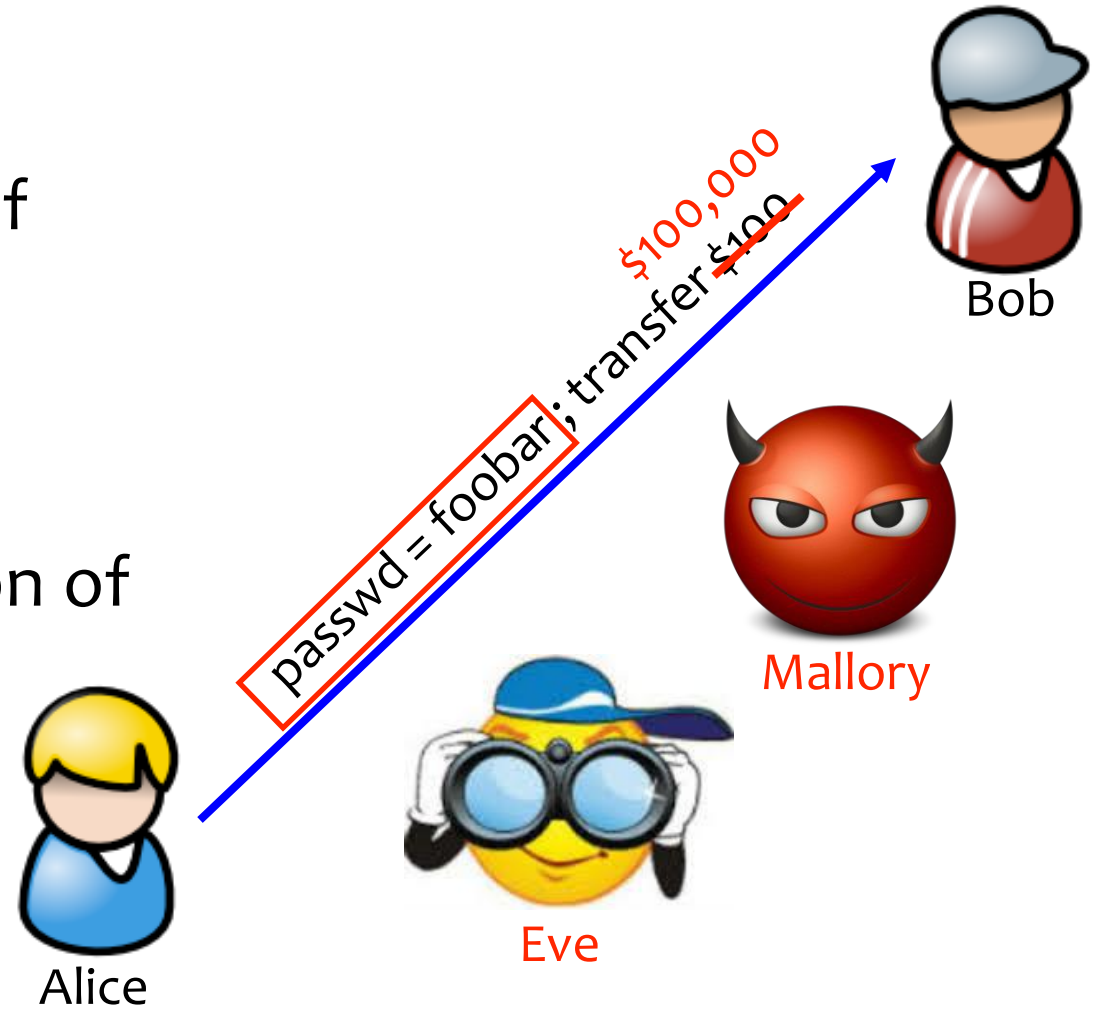
# Common Communication Security Goals

**Privacy of data:**

Prevent exposure of information

**Integrity of data:**

Prevent modification of information

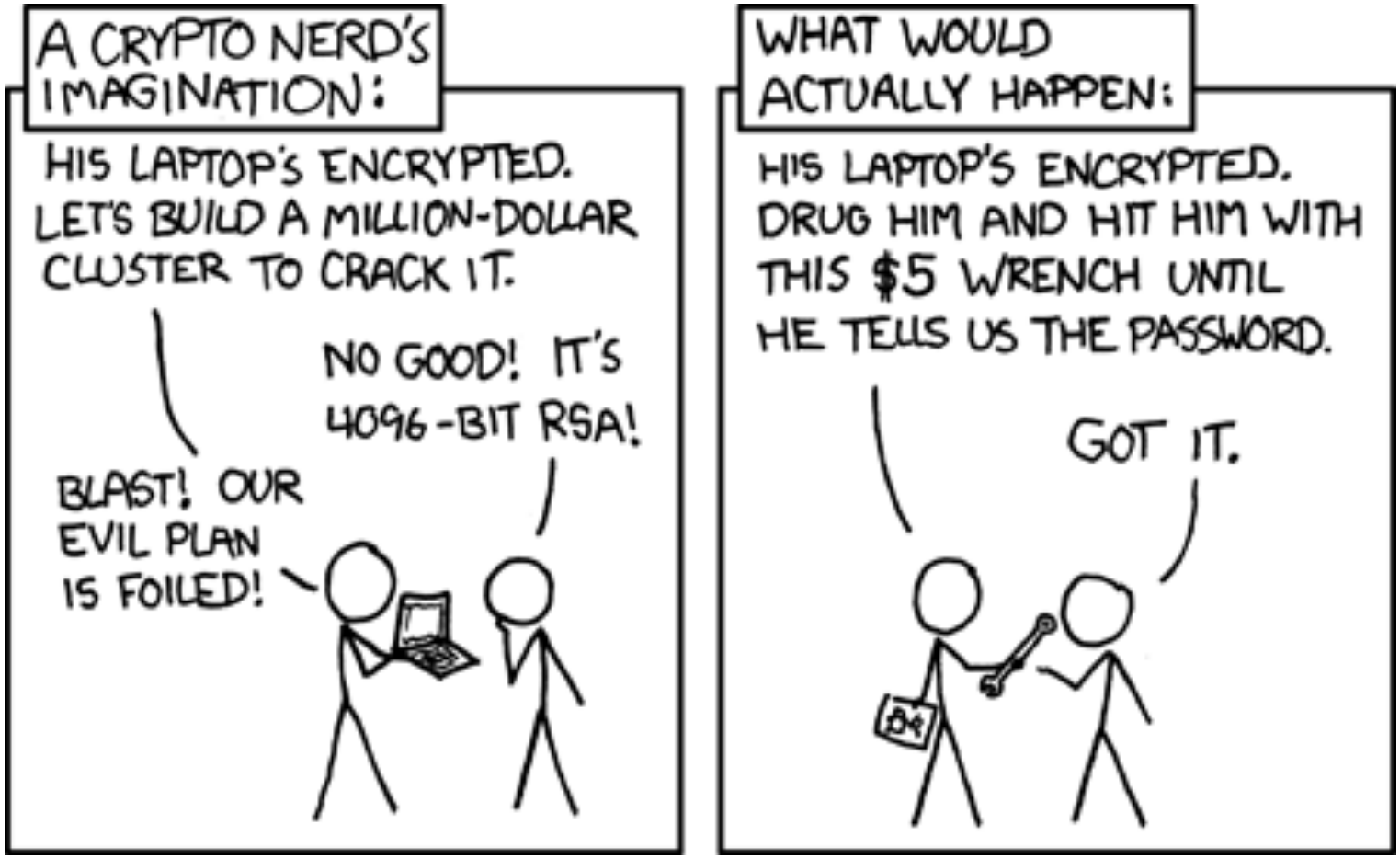


# Recall Bigger Picture

- Cryptography only one small piece of a larger system
- Must protect entire system
  - Physical security
  - Operating system security
  - Network security
  - Users
  - Cryptography (following slides)
- Recall the weakest link

- Still, cryptography is a crucial part of our toolbox

# XKCD: http://xkcd.com/538/



# History

- Substitution Ciphers
  - Caesar Cipher
- Transposition Ciphers
- Codebooks
- Machines

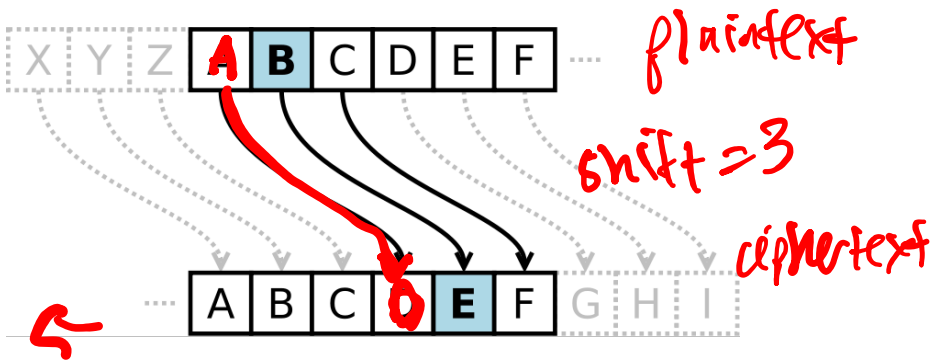
- Recommended Reading: **The Codebreakers** by David Kahn and **The Code Book** by Simon Singh.

# History: Caesar Cipher (Shift Cipher)

- Plaintext letters are replaced with letters a fixed shift away in the alphabet.
- Example:
  - Plaintext: `The quick brown fox jumps over the lazy dog`
  - Key: Shift 3

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ
DEFGHIJKLMNOPQRSTUVWXYZABC
```

  - Ciphertext: `WKHTX LFNEU RZQIR AMXPS VRYHU WKHOD CBGRJ`



# History: Caesar Cipher (Shift Cipher)

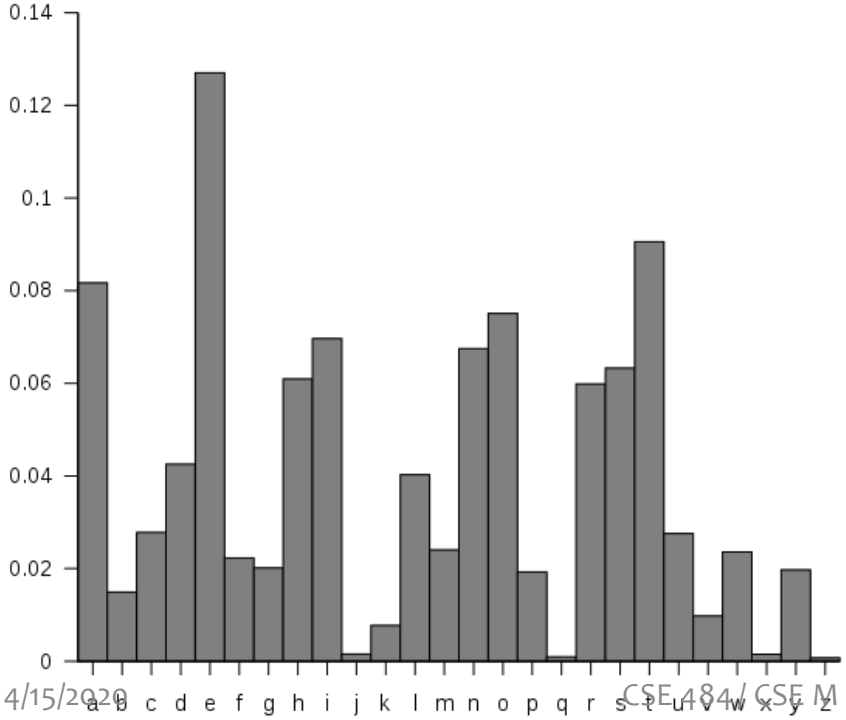
- ROT13: shift 13 (encryption and decryption are symmetric)
- What is the key space?
  - 26 possible shifts.
- How to attack shift ciphers?
  - Brute force.

# History: Substitution Cipher

- Superset of shift ciphers: each letter is substituted for another one.
- Add a secret key
- Example:
  - Plaintext: ABCDEFGHIJKLMNOPQRSTUVWXYZ
  - Cipher: ZEBRASCDFGHIJKLMNOPQTUVWXY
- "State of the art" for thousands of years

# History: Substitution Cipher

- What is the key space? 26! ~= 2^88
- How to attack?
  - Frequency analysis.




**Bigrams:**

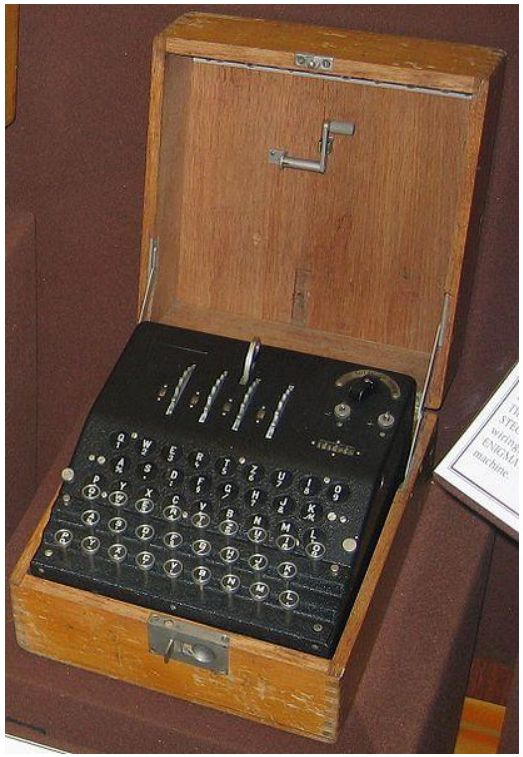
| | | |
|---|---|---|
| th 1.52% | en 0.55% | ng 0.18% |
| he 1.28% | ed 0.53% | of 0.16% |
| in 0.94% | to 0.52% | al 0.09% |
| er 0.94% | it 0.50% | de 0.09% |
| an 0.82% | ou 0.50% | se 0.08% |
| re 0.68% | ea 0.47% | le 0.08% |
| nd 0.63% | hi 0.46% | sa 0.06% |
| at 0.59% | is 0.46% | si 0.05% |
| on 0.57% | or 0.43% | ar 0.04% |
| nt 0.56% | ti 0.34% | ve 0.04% |
| ha 0.56% | as 0.33% | ra 0.04% |
| es 0.56% | te 0.27% | ld 0.02% |
| st 0.55% | et 0.19% | ur 0.02% |

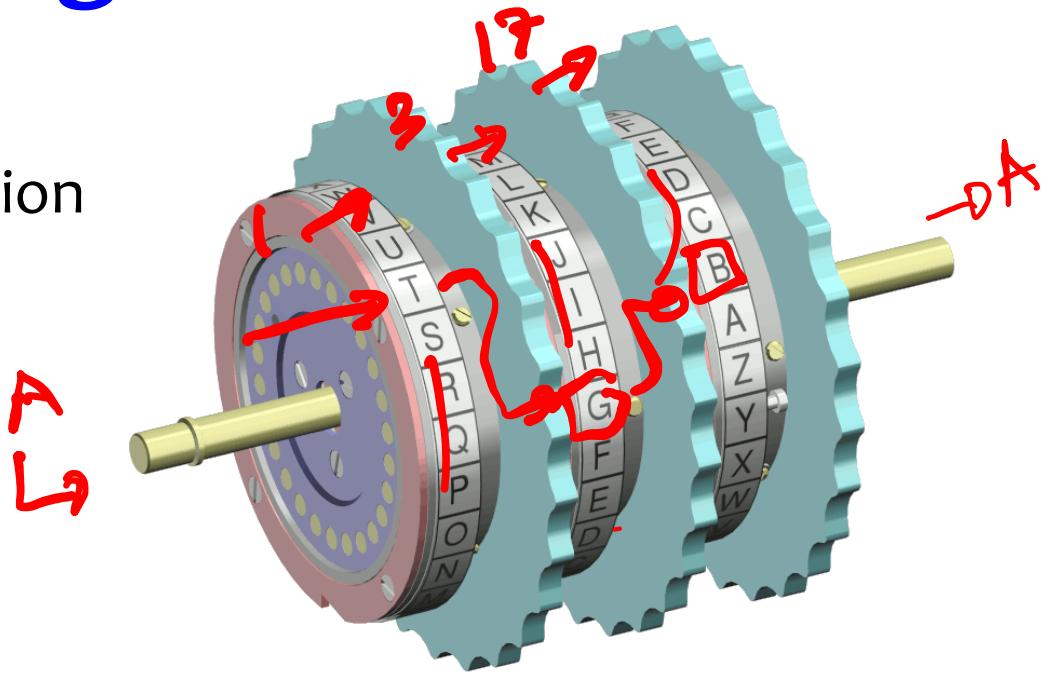
**Trigrams:**

| | | |
|---|---|---|
| 1. the | 6. ion | 11. nce |
| 2. and | 7. tio | 12. edt |
| 3. tha | 8. for | 13. tis |
| 4. ent | 9. nde | 14. oft |
| 5. ing | 10. has | 15. sth |

# History: Enigma Machine

Uses rotors (substitution cipher) that change position after each key.






Key = initial setting of rotors

Key space?

26^n for n rotors

Received April 4, 1977

# A Method for Obtaining Digital Signatures and Public-Key Cryptosystems

R.L. Rivest, A. Shamir, and L. Adleman*

**Abstract**

An encryption method is presented with the novel property that publicly revealing an encryption key does not thereby reveal the corresponding decryption key. This has two important consequences:

1. Couriers or other secure means are not needed to transmit keys, since a message can be enciphered using an encryption key publicly revealed by the intended recipient. Only he can decipher the message, since only he knows the corresponding decryption key.
2. A message can be "signed" using a privately held decryption key. Anyone can verify this signature using the corresponding publicly revealed encryption key. Signatures cannot be forged, and a signer cannot later deny the validity of his signature. This has obvious applications in "electronic mail" and "electronic funds transfer" systems.

# How Cryptosystems Work Today

- Layered approach:
  - **Cryptographic primitives**, like block ciphers, stream ciphers, hash functions, and one-way trapdoor permutations (examples: AES, SHA256, RSA)
  - **Cryptographic protocols**, like CBC mode encryption, CTR mode encryption, HMAC message authentication
- Public algorithms (Kerckhoff's Principle)
- Security proofs based on assumptions (not this course)

- Be careful about inventing your own! (If you just want to use some crypto in your system, use vetted libraries!)
- Above terms will make more sense later!

# Kerckhoff's Principle

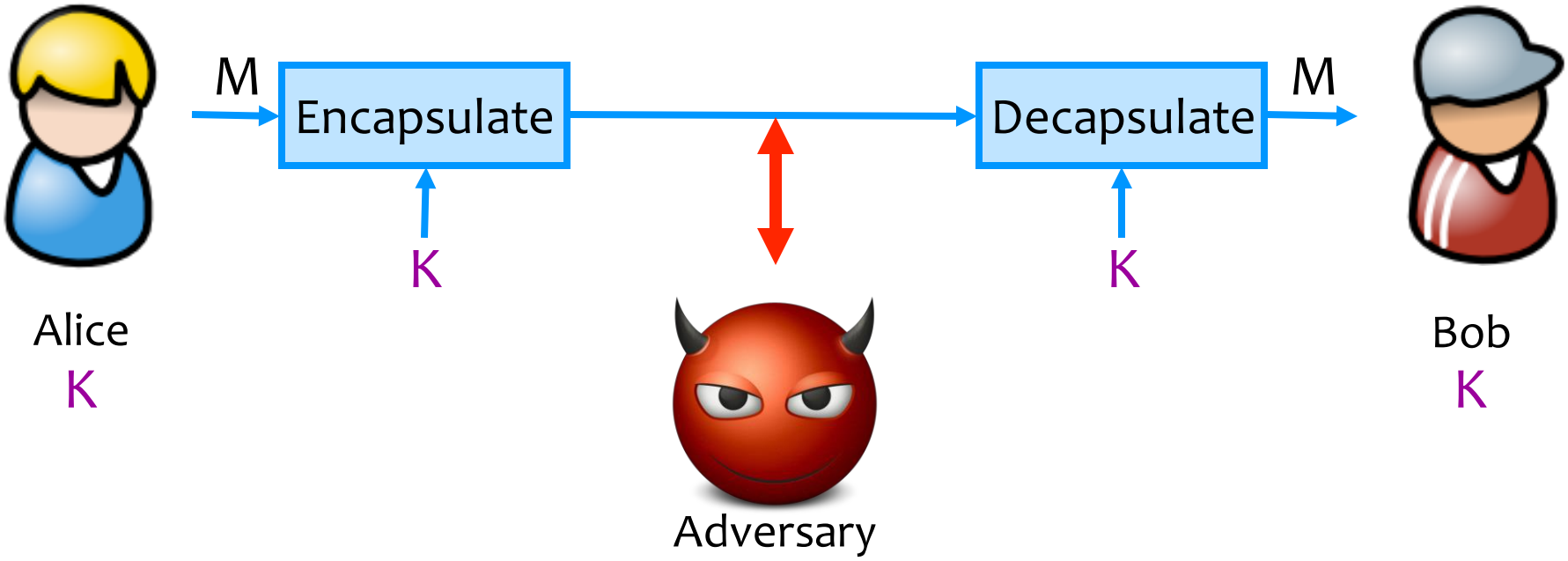
- Security of a cryptographic object should depend only on the secrecy of the secret (private) key.
- Security should not depend on the secrecy of the algorithm itself.

# Flavors of Cryptography

- Symmetric cryptography
  - Both communicating parties have access to a shared random string K, called the key.
- Asymmetric cryptography
  - Each party creates a public key pk and a secret key sk.
  - *Hard concept to understand, and revolutionary! Inventors won Turing Award* ☺

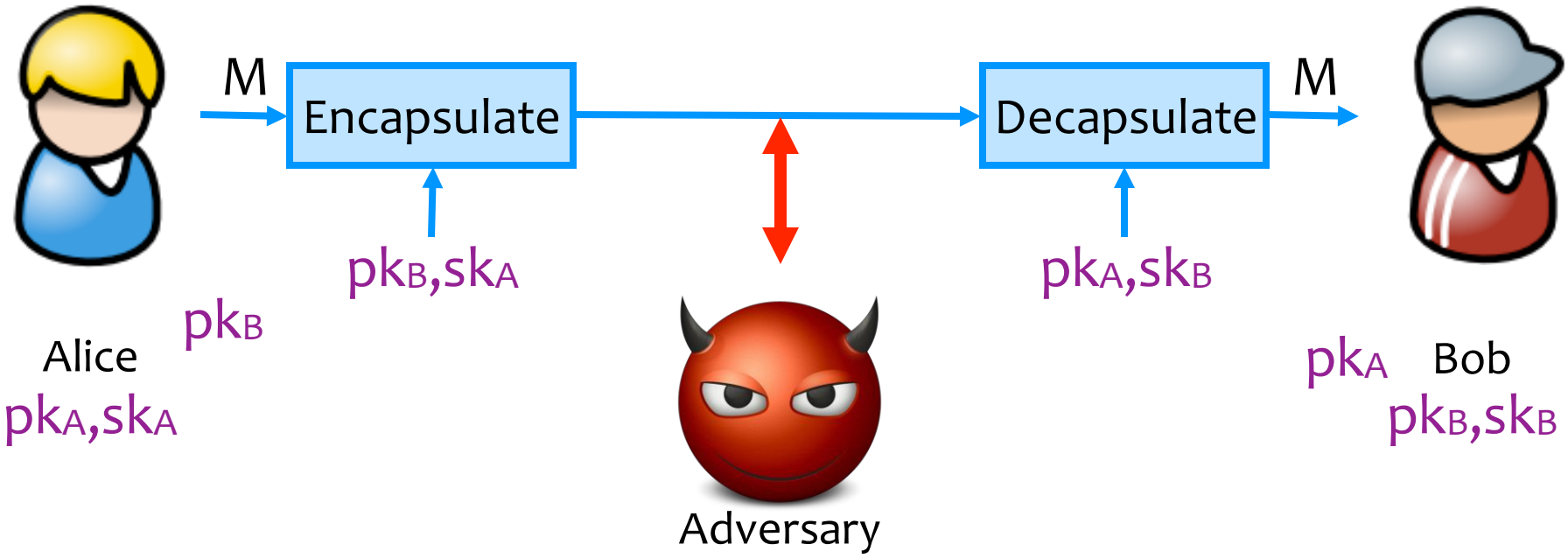
# Symmetric Setting

Both communicating parties have access to a shared random string K, called the key.



# Asymmetric Setting

Each party creates a public key pk and a secret key sk.



# Flavors of Cryptography

- Symmetric cryptography
  - Both communicating parties have access to a shared random string K, called the key.

- Asymmetric cryptography
  - Each party creates a public key pk and a secret key sk.

# Flavors of Cryptography

- Symmetric cryptography
  - Both communicating parties have access to a shared random string K, called the key.
  - Challenge: How do you privately share a key?
- Asymmetric cryptography
  - Each party creates a public key pk and a secret key sk.
  - Challenge: How do you validate a public key?