

CSE 484 / CSE M 584: Cryptography: Introduction

Fall 2024

Franziska (Franzi) Roesner
franzi@cs

UW Instruction Team: David Kohlbrenner, Yoshi Kohno, Franziska Roesner. Thanks to Dan Boneh, Dieter Gollmann, Dan Halperin, John Manferdelli, John Mitchell, Vitaly Shmatikov, Bennet Yee, and many others for sample slides and materials ...

Announcements

- Things Due
 - Lab 1a, due Monday
 - Research Reading #2 (584M) due Thursday

Next major section of the course:

Cryptography

Terminology note: “blockchain” and “crypto”

- Rising interest, mostly in the cryptocurrency space
- For this course: crypto means “cryptography”

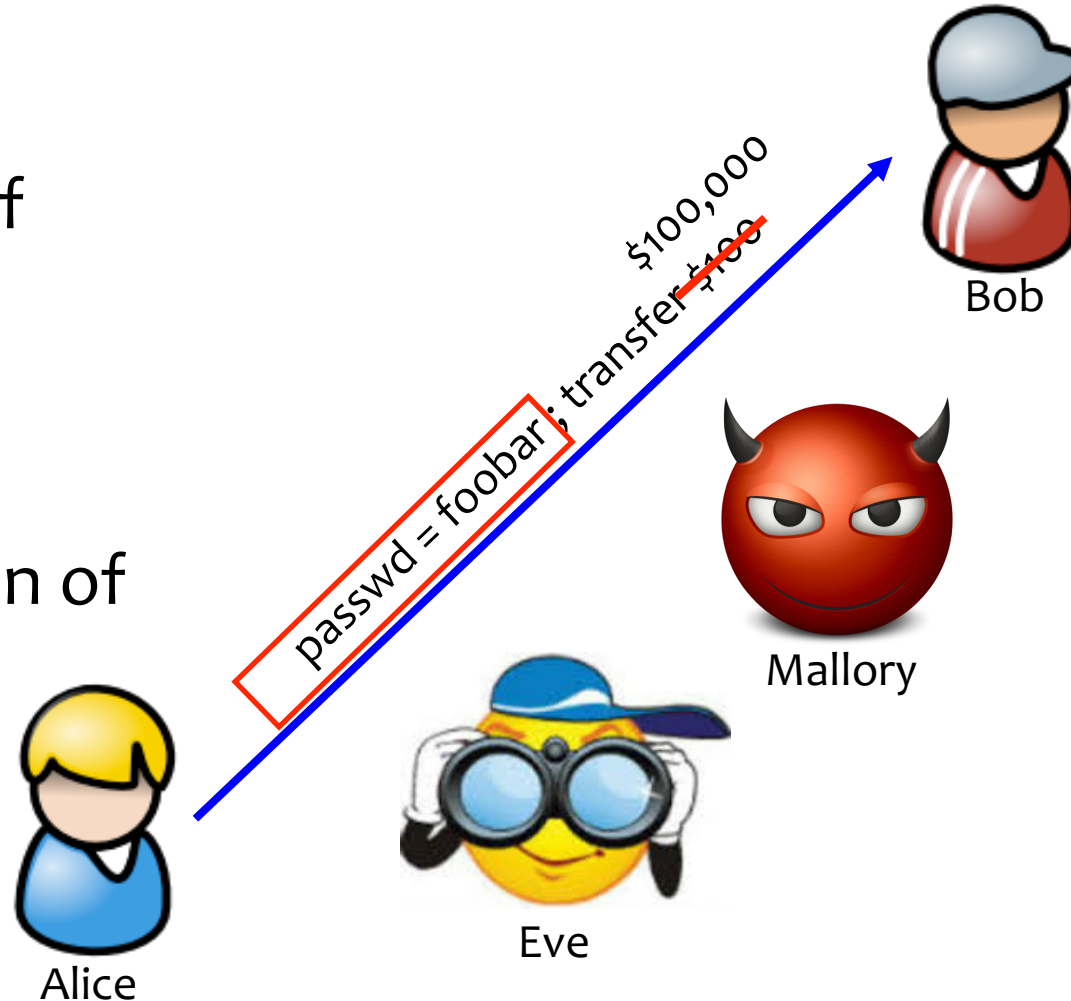
Common Communication Security Goals

Privacy of data:

Prevent exposure of information

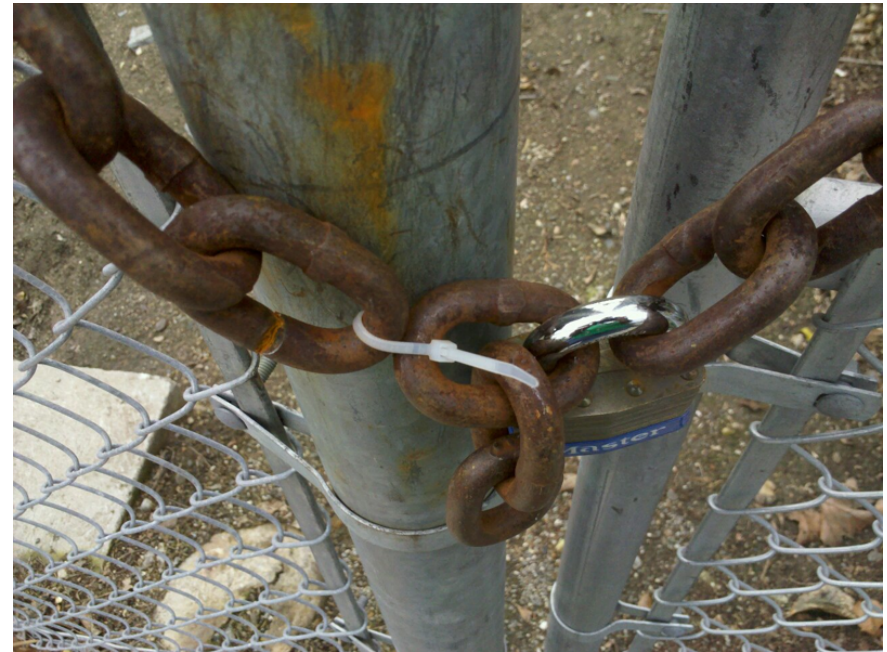
Integrity of data:

Prevent modification of information

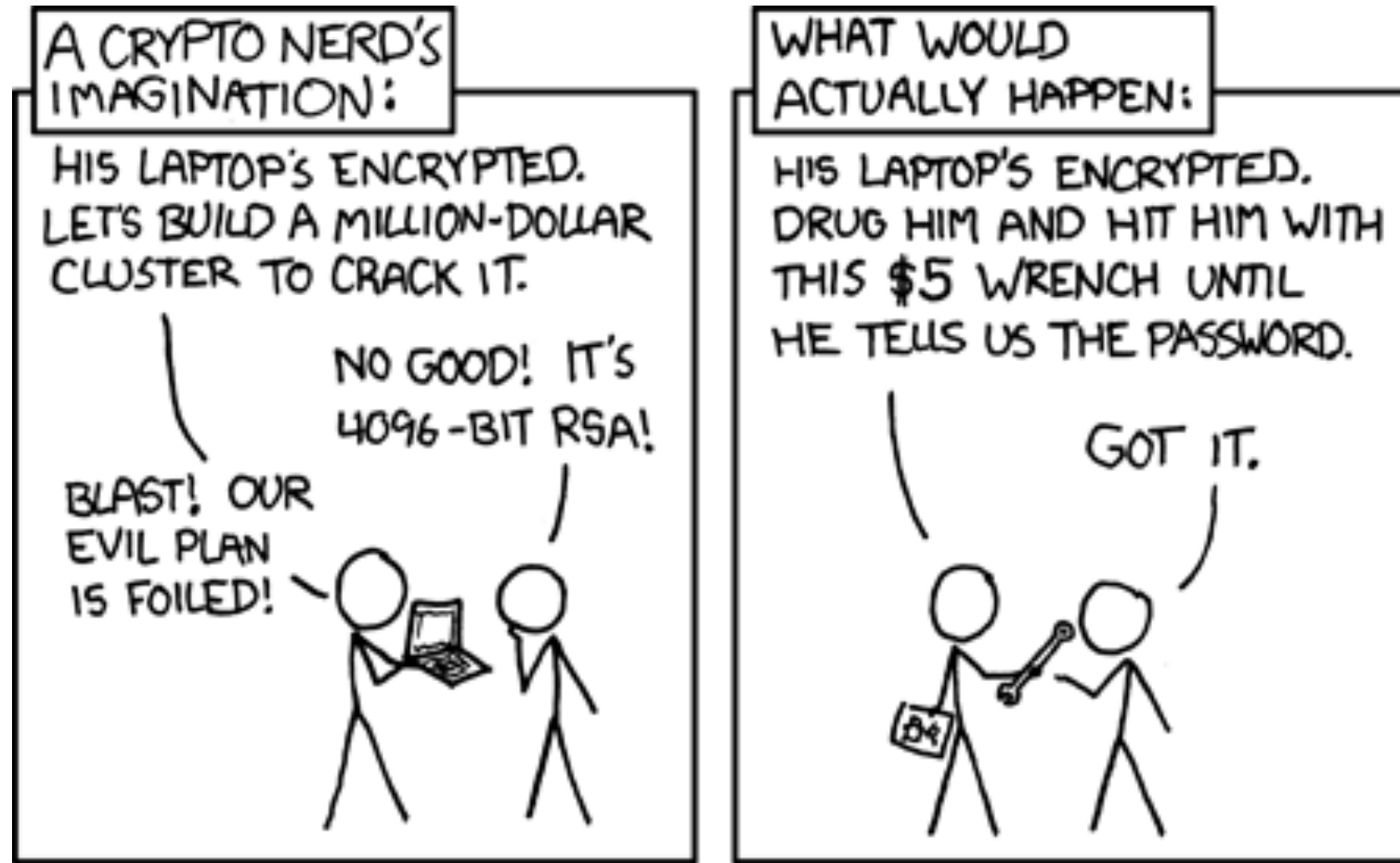


Recall Bigger Picture

- 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/>



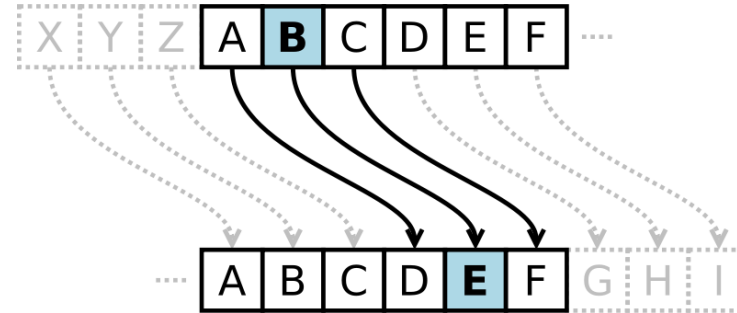
History

- 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.
- One way to implement: **Add a secret key**
- Example:
 - Plaintext: ABCDEFGHIJKLMNOPQRSTUVWXYZ
 - Cipher: **ZEBRAS** CDEFGHIJKLMNOPQTUVWXY
- **“State of the art”** for thousands of years

History: Substitution Cipher

- What is the key space?
- How to attack?
 - Frequency analysis.

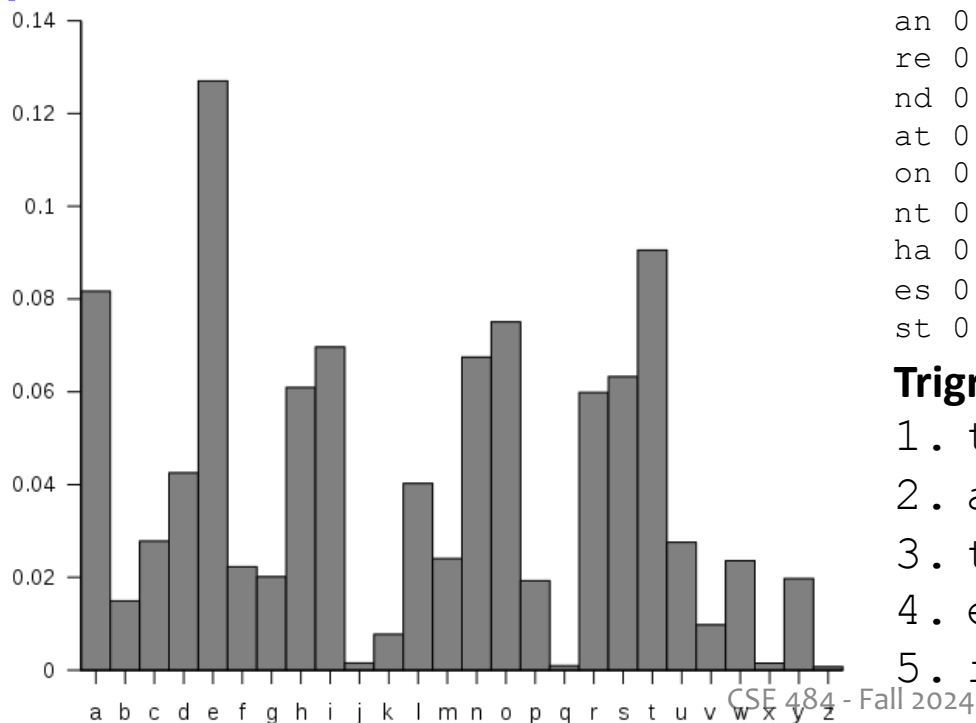
$$26! \approx 2^{88}$$

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%

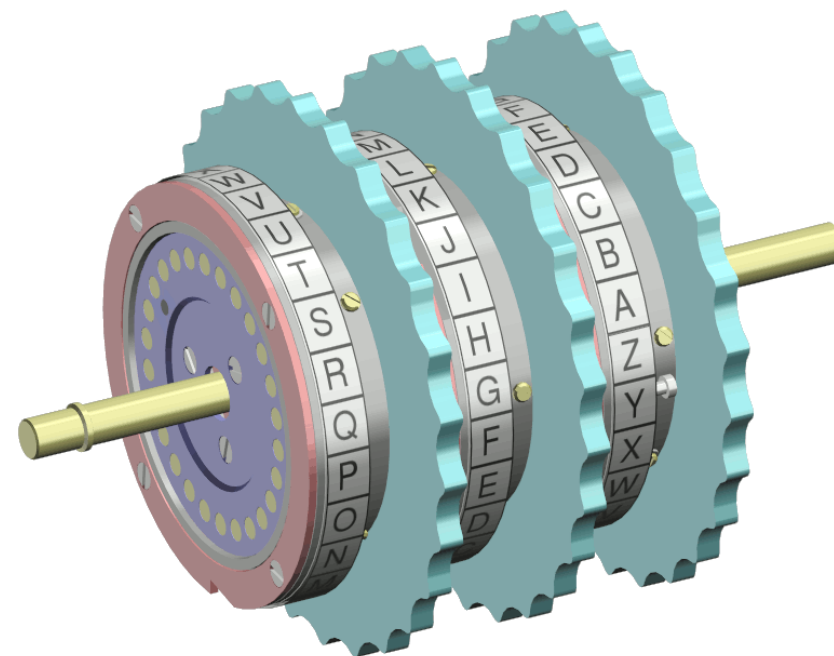
Trigrams:

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

How Cryptosystems Work Today

- **Layered approach:** Cryptographic protocols (like “CBC mode encryption”) built on top of cryptographic primitives (like “block ciphers”)
- **Flavors of cryptography:** Symmetric (private key) and asymmetric (public key)
- Public algorithms (Kerckhoff’s Principle – next slide)
- 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!)

Kerckhoff's Principle

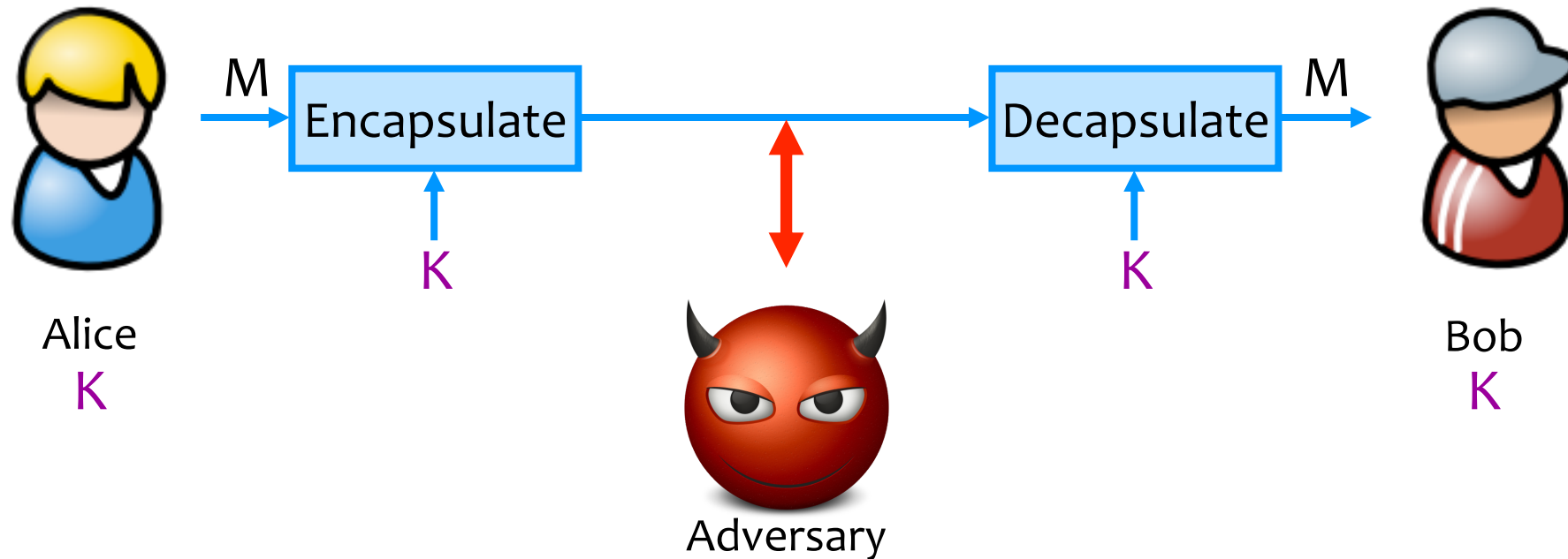
- Security of a cryptographic object should depend only on the secrecy of the secret (private) key.
- Security should not depend on the secrecy of the algorithm itself.
- (Foreshadowing: Need for randomness – the key to keep private)

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 fully understand, and revolutionary! Inventors won the Turing Award 😊*

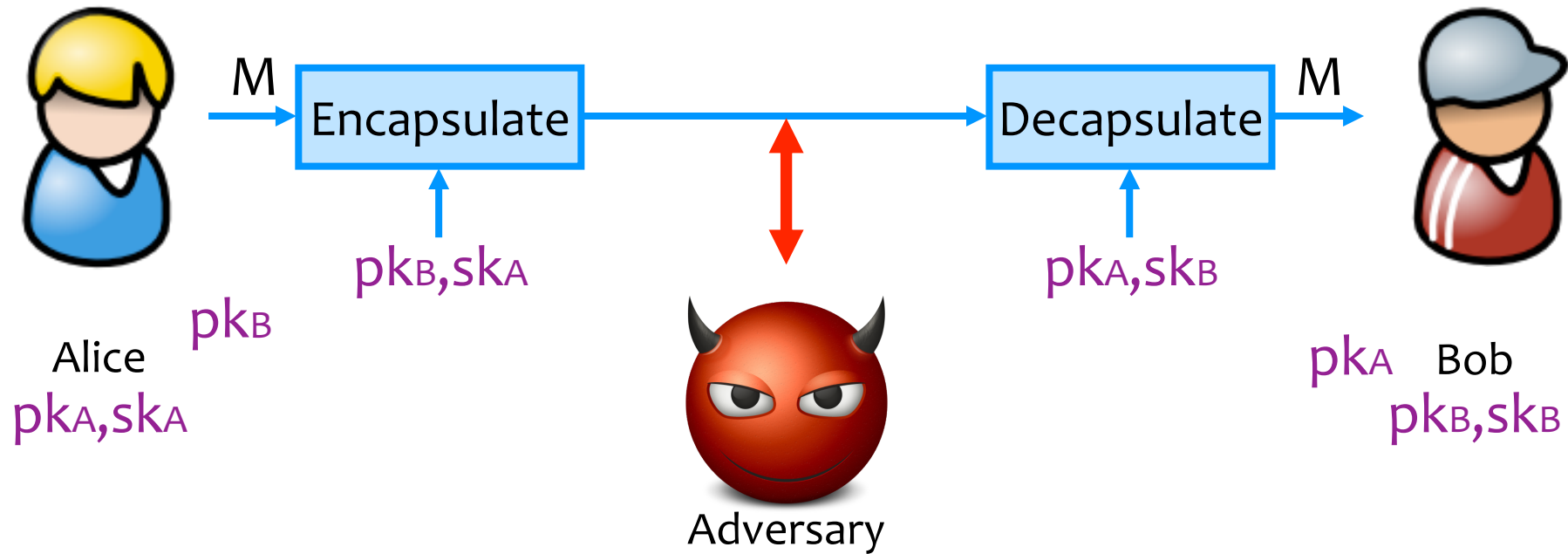
Symmetric Setting

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 .



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.

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?**
- **Key building block: Randomness** – something that the adversaries don't know, can't predict, can't figure out

Ingredient: Randomness

- Many applications (especially security ones) require randomness
- Explicit uses:
 - Generate secret cryptographic keys
 - Generate random initialization vectors for encryption
- Other “non-obvious” uses:
 - Generate passwords for new users
 - Shuffle the order of votes (in an electronic voting machine)
 - Shuffle cards (for an online gambling site)

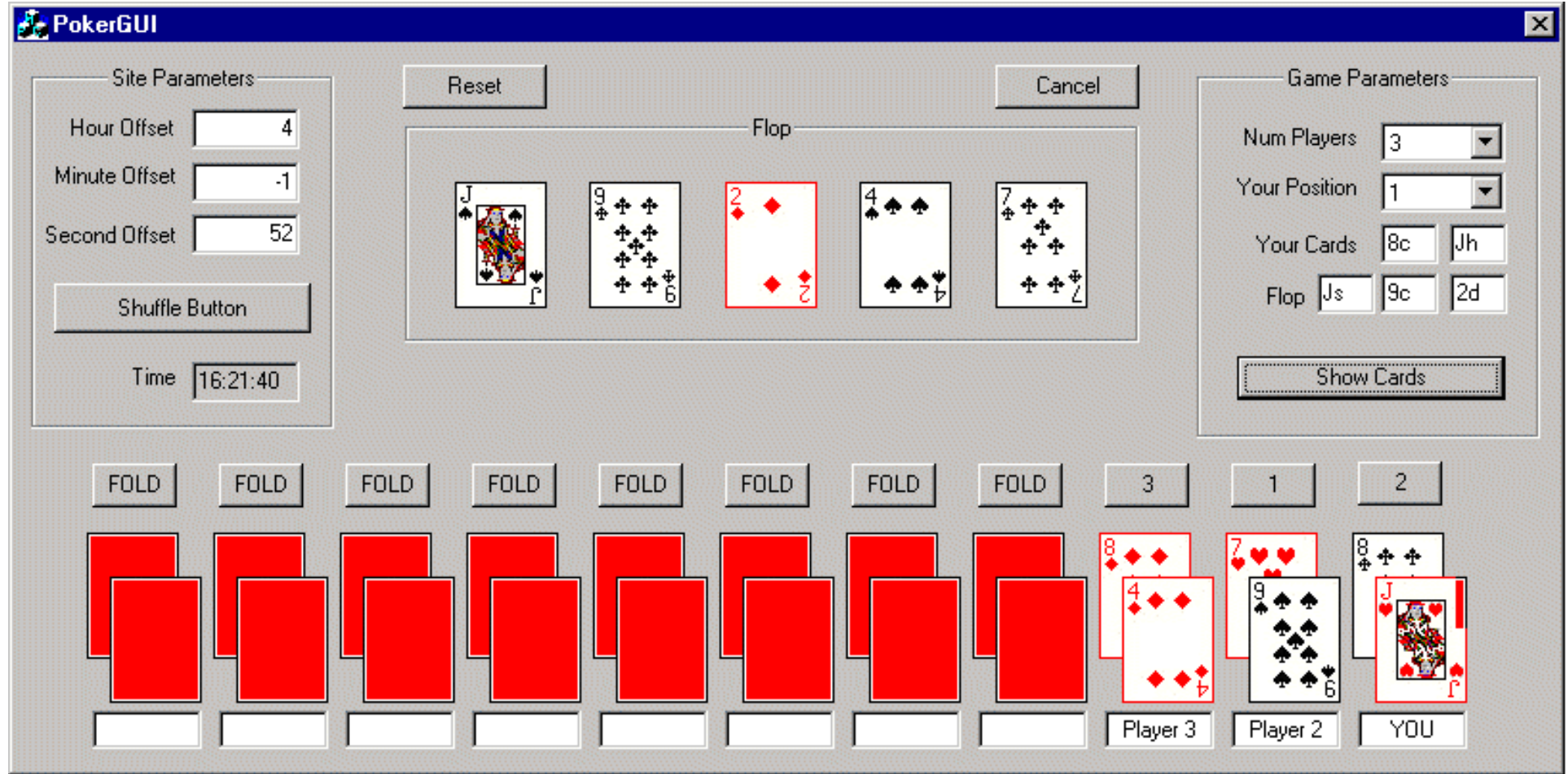
C's rand() Function

- C has a built-in random function: `rand()`

```
unsigned long int next = 1;
/* rand:  return pseudo-random integer on 0..32767 */
int rand(void) {
    next = next * 1103515245 + 12345;
    return (unsigned int)(next/65536) % 32768;
}
/* srand:  set seed for rand() */
void srand(unsigned int seed) {
    next = seed;
}
```

- Problem: don't use `rand()` for security-critical applications!
 - Given a few sample outputs, you can predict subsequent ones





More details: "How We Learned to Cheat at Online Poker: A Study in Software Security"

http://www.cigital.com/papers/download/developer_gambling.php

PS3 and Randomness

Hackers obtain PS3 private cryptography key due to epic programming fail? (update)

<http://www.engadget.com/2010/12/29/hackers-obtain-ps3-private-cryptography-key-due-to-epic-programm/>

- 2010/2011: Hackers **found/released private root key** for Sony's PS3
- Key used to sign software – **now can load any software on PS3** and it will execute as “trusted”
- Due to bad random number: **same “random” value used to sign all system updates**