WILLIAM & MARY
CHARTERED 1693

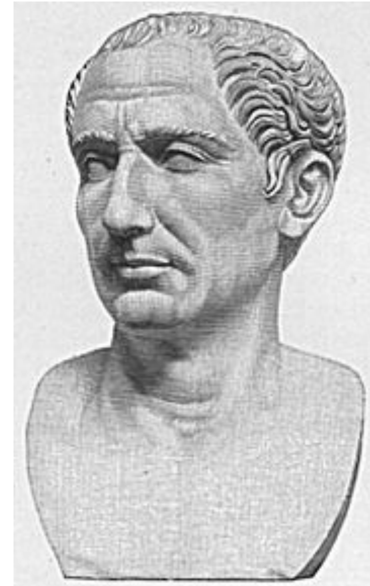
CSCI 667:
Computer & Network Security

Lecture 3

Prof. Adwait Nadkarni

(Derived from slides by William Enck, Micah Sherr, Patrick McDaniel, and Peng Ning)

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

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

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

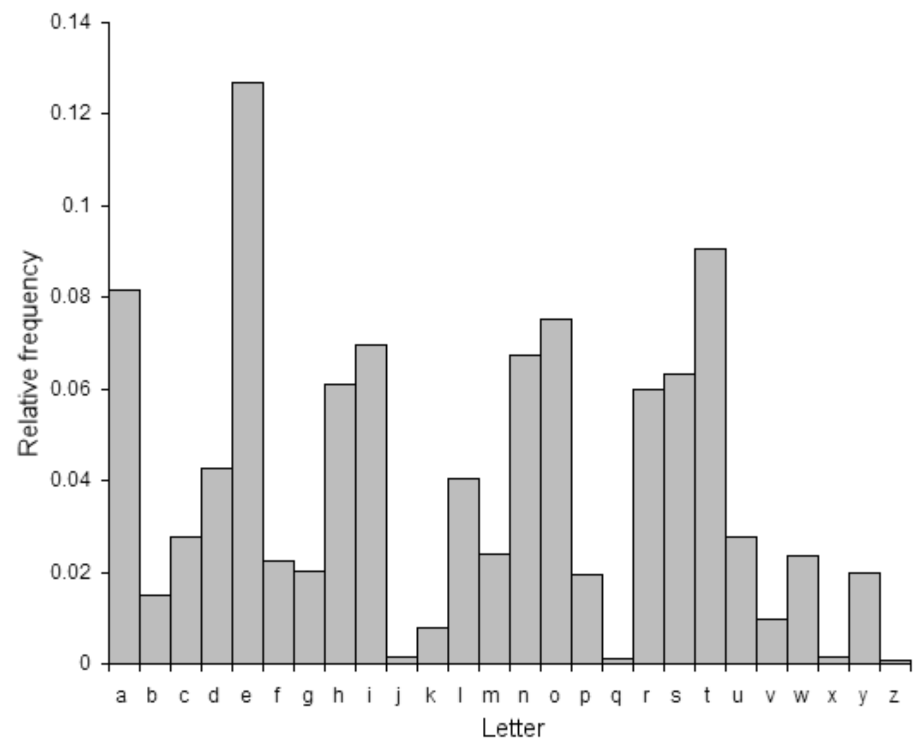
V H F X U L W B D Q G S U L Y D F B

Cryptanalyze this …

"IWXH XH P VGTPI RAPHH HTRJGXIN XH UJC!"

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



Substitution Cipher

- aka Monoalphabetic Substitution Cipher
- Map each letter of the alphabet to another letter of the alphabet according to some fixed (but random) permutation
- E.g., cryptogram puzzles
- "Key size" is 26! (that's factorial, not, holy cow, 26!)
- E.g., (H→Z,E→A,L→Q,O→O): ZAQQO
- Cryptanalysis:
  - frequency analysis (single letters, bigrams, trigrams)
  - pattern analysis: (double Qs could be double Ds, Es, Ls, etc.)

A	B	C	D	E	F	G	H	I	J	K	L	M
↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
C	M	T	E	F	H	P	U	D	X	N	Z	L

N	O	P	Q	R	S	T	U	V	W	X	Y	Z
↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
O	A	J	R	Y	I	G	W	V	B	S	Q	K

Substitution Cipher

- Vg gbbx n ybg bs oybbq, fj**r**ng naq g**r**nef gb t**r**g gb ju**r**e**r** j**r** ne**r** gbqnl, ohg j**r** uni**r** whfg o**r**tha. Gbqnl j**r** o**r**tva va **r**nea**r**fg gu**r** jbex bs znxvat fhe**r** gung **gur** jbeyq j**r** y**r**ni**r** bhe puvyqe**r**a vf whfg n yvggy**r** ovg o**r**gg**r**e guna **gur** ba**r** j**r** vaunovg gbqnl.
- It took a lot of blood, sw**e**at and t**e**ars to get to wh**e**r**e** w**e** ar**e** today, but w**e** hav**e** just b**e**gun. Today w**e** b**e**gin in **e**arn**e**st th**e** work of making sur**e** that **the** world w**e** l**e**av**e** our childr**e**n is just a littl**e** bit b**e**tt**e**r than **the** on**e** w**e** inhabit today.

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

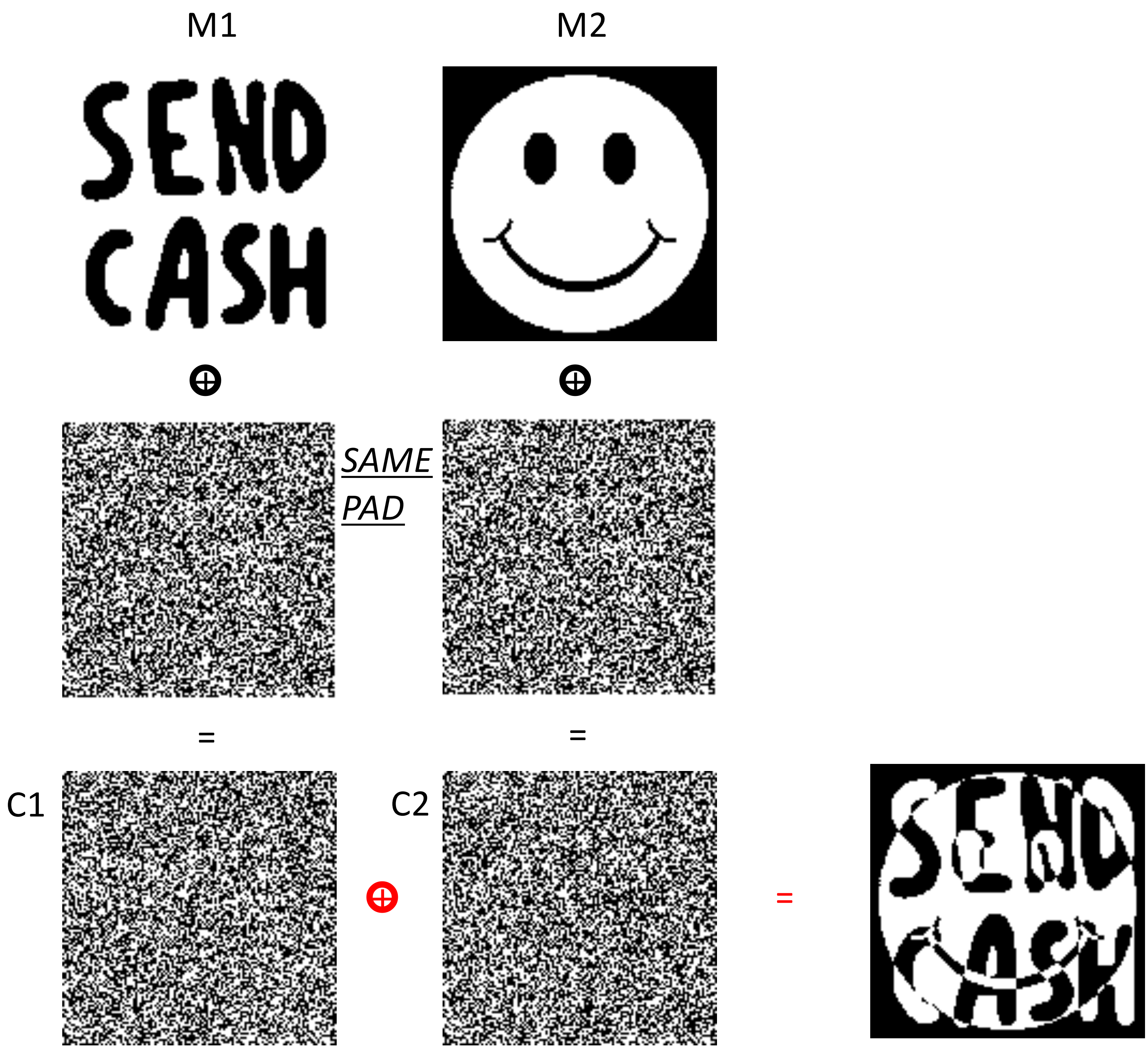
key: deceptivedeceptivedeceptive
plaintext: wearediscoveredsaveyourself
ciphertext: ZICVTWQNGRZGVTWAVZHCQYGLMGJ

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?



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

Modern Cryptography

Two flavors of confidentiality

- **Unconditional** or **probabilistic security**: cryptosystem offers provable guarantees, irrespective of computational abilities of an attacker
  - given ciphertext, the probabilities that bit i of the plaintext is 0 is p and the probability that it is 1 is (1-p)
  - e.g., one-time pad
  - often requires key sizes that are equal to size of plaintext
- **Conditional** or **computational security**: cryptosystem is secure assuming a computationally bounded adversary, or under certain hardness assumptions (e.g., P<>NP)
  - e.g., DES, 3DES, AES, RSA, DSA, ECC, DH, MD5, SHA
  - Key sizes are much smaller (~128 bits)
- Almost all deployed modern cryptosystems are conditionally secure

An aside about key sizes

- Original DES used 56-bit keys
- 3DES uses 168-bit keys
- AES uses 128-, 192- or 256-bit keys
- Are these numbers big enough?
  - DES has 2^{56} = 72,057,594,037,927,936 possible keys
    - In Feb 1998, distributed.net cracked DES in 41 days
    - In July 1998, the Electronic Frontier Foundation (EFF) and distributed.net cracked DES in 56 hours using a $250K machine
    - In Jan 1999, the team did in less than 24 hours
  - **Each additional bit adds 2X brute-force work factor (exponential security for linear keysize increase)**
  - There are approximately 2^{250} atoms in the universe, so don't expect 256-bit keys to be brute forced anytime in the next trillion years.
- Takeaway: 128-keys are reasonably secure

$2^{256} =$

115,792,089,237,316,195,
423,570,985,008,687,907,
853,269,984,665,640,564,
039,457,584,007,913,129,
639,936

Cryptanalysis

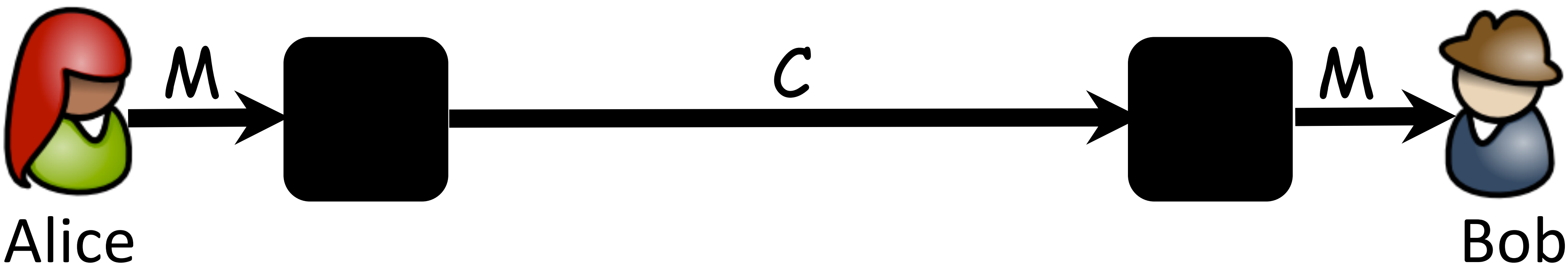
- Goal: learn the key
- Classifications:
  - **ciphertext-only** attack: Eve has access only to ciphertext
  - **known-plaintext** attack: Eve has access to plaintext and corresponding ciphertext
  - **chosen-plaintext** attack: Eve can choose plaintext and learn ciphertext
  - **chosen-ciphertext** attack: Eve can choose ciphertext and learn plaintext

Which of these are passive/active attacks?

Other cryptanalysis …

- Brute force cryptanalysis
  - Just keep trying different keys and check result
- Not covered in this class:
  - Linear cryptanalysis
    - Construct linear equations relating plaintext, ciphertext and key bits that have a high bias
    - Use these linear equations in conjunction with known plaintext-ciphertext pairs to derive key bits
  - Differential cryptanalysis
    - Study how differences in an input can affect the resultant difference at the output
    - Use chosen plaintext to uncover key bits

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

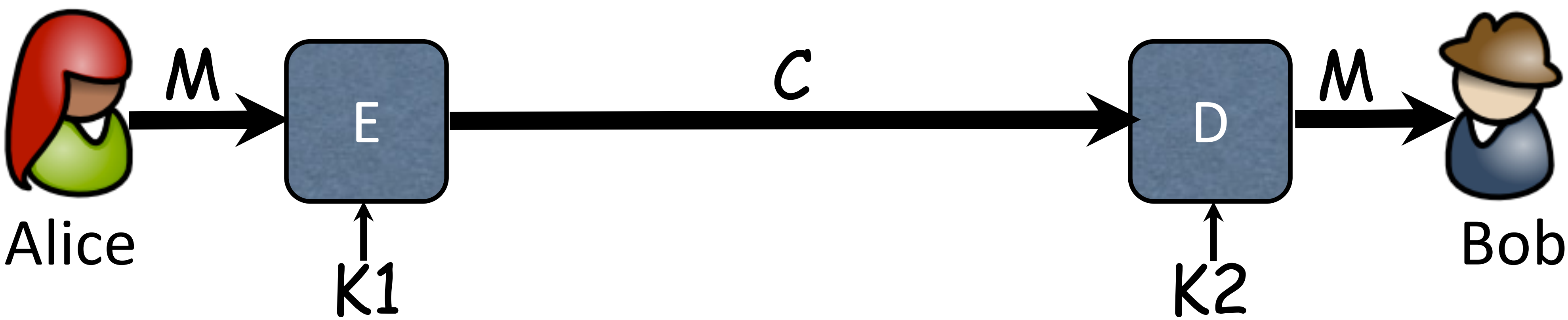
Kerckhoffs' Principles

- Modern cryptosystems use a key to control encryption and decryption
- Ciphertext should be undecipherable without the correct key
- Encryption key may be different from decryption key.
- **Kerckhoffs' principles** [1883]:
  - Assume Eve knows cipher algorithm
  - Security should rely on choice of key
  - If Eve discovers the key, a new key can be chosen

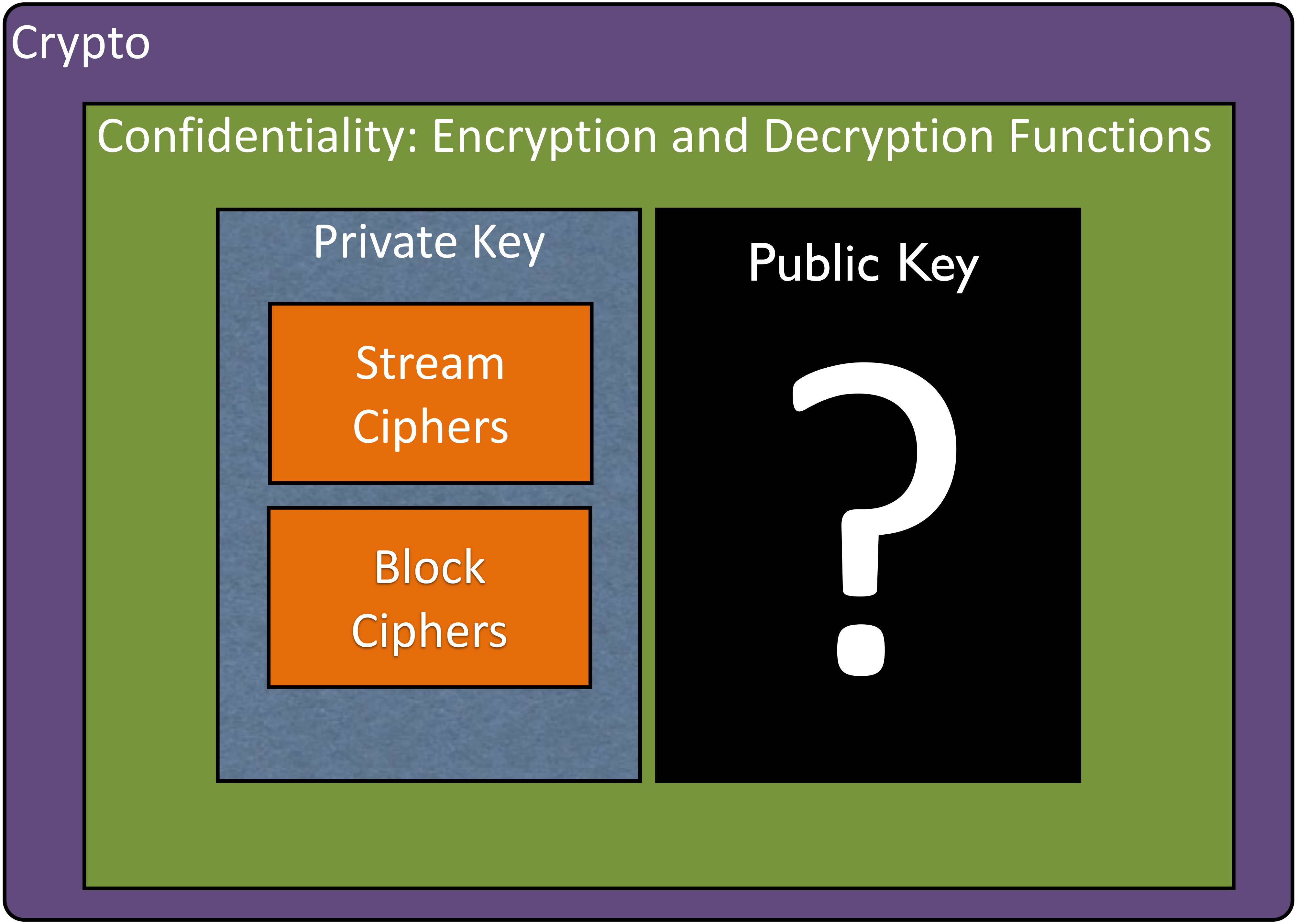
Kerckhoffs' Principles

- Kerckhoffs' Principles are contrary to the principle of "**security by obscurity**", which relies only upon the secrecy of the algorithm/cryptosystem
  - If security of a keyless algorithm compromised, cryptosystem becomes permanently useless (and unfixable)
  - Algorithms relatively easy to reverse engineer

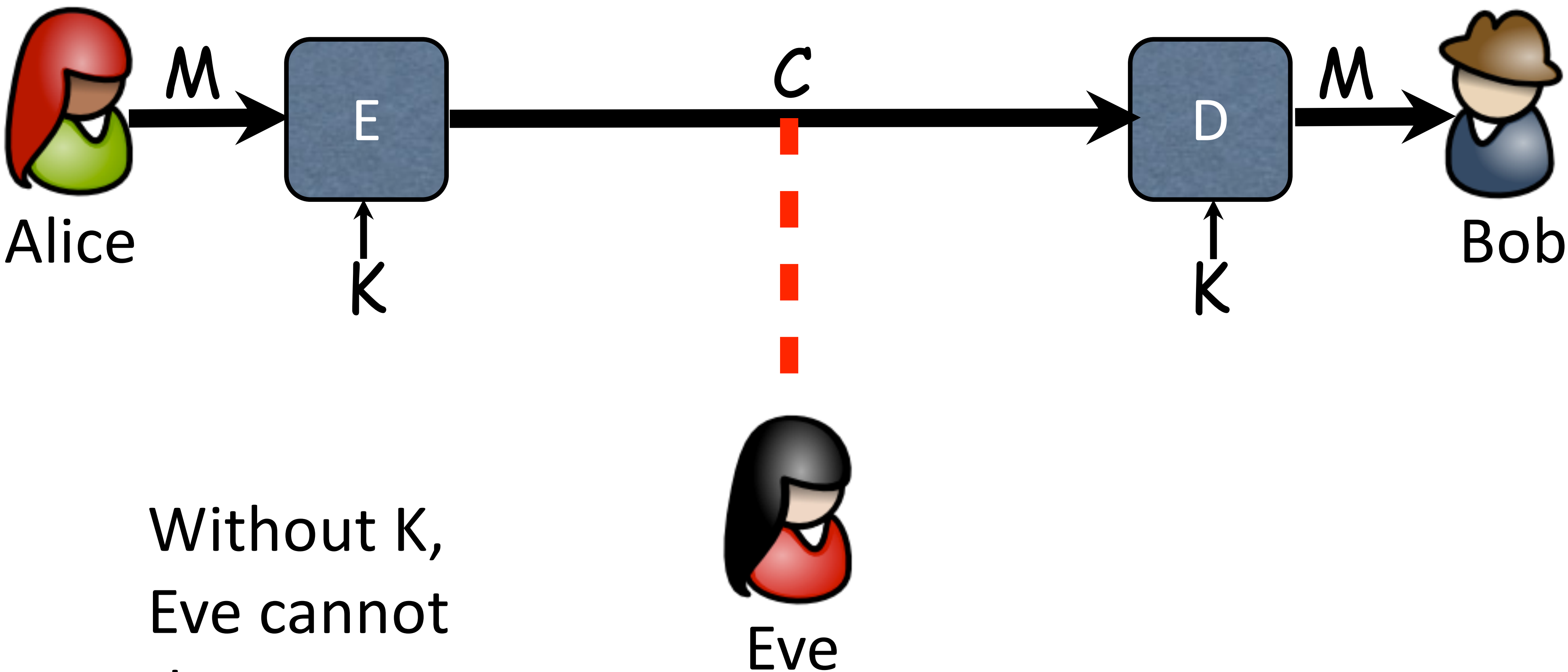
Symmetric and Asymmetric Crypto



- **Symmetric crypto:** (also called **private key crypto**)
  - Alice and Bob share the same key (K=K1=K2)
  - K used for both encrypting and decrypting
  - Doesn't imply that encrypting and decrypting are the same algorithm
  - Also called **private key** or **secret key** cryptography, since knowledge of the key reveals the plaintext
- **Asymmetric crypto:** (also called **public key crypto**)
  - Alice and Bob have different keys
  - Alice encrypts with K1 and Bob decrypts with K2
  - Also called **public key** cryptography, since Alice and Bob can publicly post their *public* keys


Crypto
Confidentiality: Encryption and Decryption Functions
Private Key
Stream Ciphers
Block Ciphers
Public Key
?

Secret/Symmetric/Private Key Crypto



Without K,
Eve cannot
decrypt C

Block ciphers vs. Stream ciphers

- **Stream Ciphers**
  - [C(K) = pseudorandom stream produced using key K]
  - Combine (e.g., XOR) plaintext M with C(K) to produce E(M)
  - Pseudorandom stream generated based on key
  - XOR with same bit stream to recover plaintext
  - E.g., RC4, FISH
- **Block Ciphers**
  - Encrypt fixed block-sized portions of plaintext
  - Combine encrypted blocks (more on this later)
  - E.g., DES, 3DES, AES

Stream Ciphers

- Useful when plaintext arrives as a stream (e.g., 802.11's WEP)
- Vulnerable if used incorrectly

Stream Ciphers

- **Key reuse:** [C(K) = pseudorandom stream produced using key K]
  - $E(M1) = M1 \oplus C(K)$
  - $E(M2) = M2 \oplus C(K)$
  - Suppose Eve knows ciphertexts E(M1) and E(M2)
  - $E(M1) \oplus E(M2) = M1 \oplus C(K) \oplus M2 \oplus C(K) = M1 \oplus M2$
  - M1 and M2 can be derived from $M1 \oplus M2$ using frequency analysis
- Countermeasure is to use IV (**initialization vector**)
  - IV <u>sent in clear</u> and is combined with K to produce pseudorandom sequence
  - E.g., replace C(K) with $C(K \oplus IV)$
  - IVs should never be reused and should be sufficiently large
  - WEP broken partly because IVs were insufficiently large
  - modern stream ciphers take IVs, but it's up to the programmer to generate them

Stream Ciphers

- **Substitution Attack:**
  - M = "Pay me $100.00"
  - E(M) = M $\oplus$ C(K)
  - Suppose Eve knows M and E(M) but doesn't know K
  - She can substitute M for M' by replacing E(M) with:
    - E'(M) = E(M) $\oplus$ M $\oplus$ M' = M $\oplus$ C(K) $\oplus$ M $\oplus$ M' = C(K) $\oplus$ M'
    - Eve can then replace E(M) with E'(M), which Bob will decrypt message as M' ("Pay me $900.00")
  - Countermeasure is to include message authentication code (more on this later) that helps detect manipulation (i.e., provides integrity and authenticity)

The End