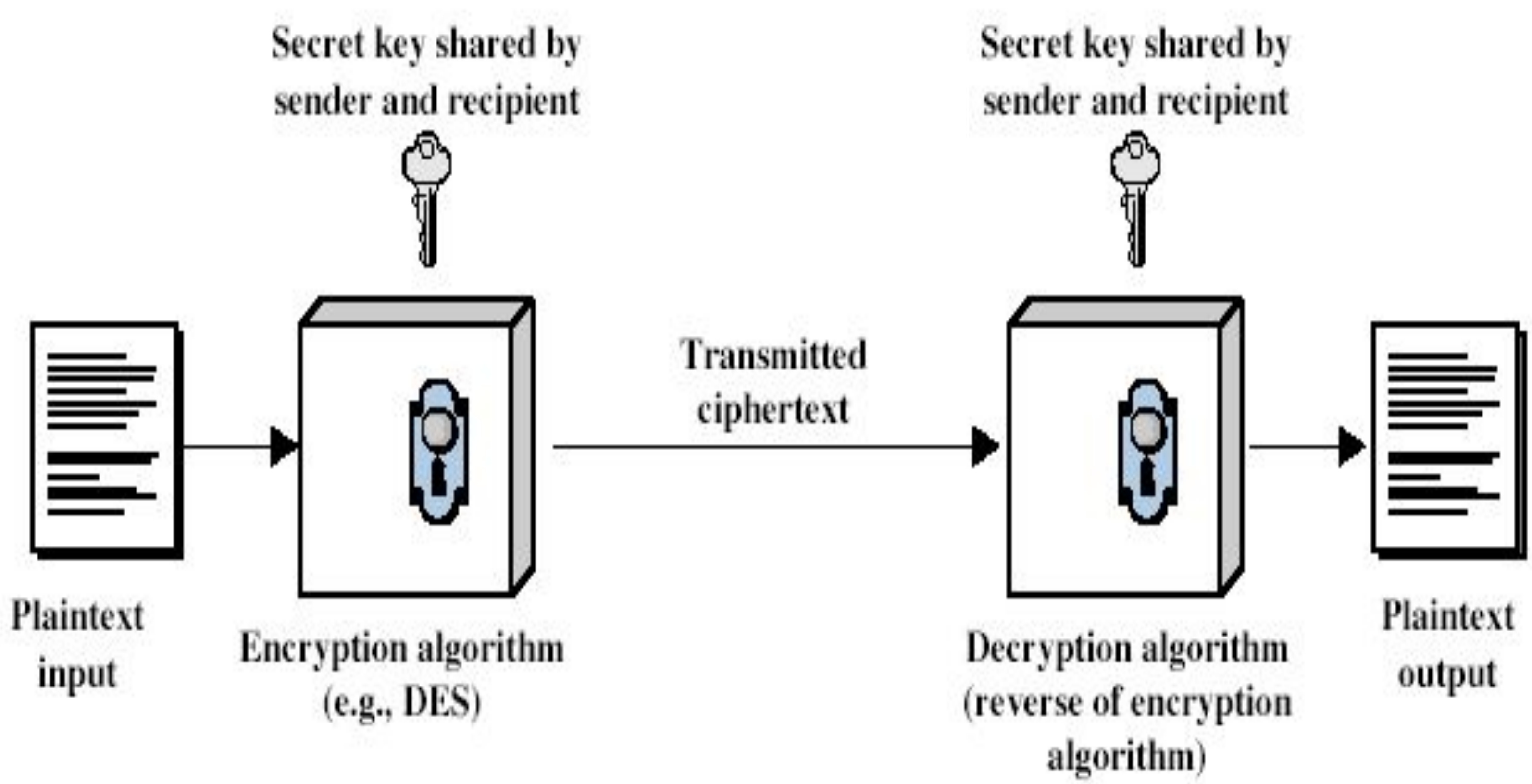
F2a: symmetric key cryptography

tkkwon@snu.ac.kr

Symmetric Encryption

- Classic ciphers
- also known as (AKA)
  - single key
  - Secret key
- sender and recipient share a common key
- was only type prior to invention of public-key cryptography
  - until second half of 1970's

Symmetric Cipher Model



source: William Stallings

Requirements

- two requirements for secure use of symmetric encryption:
  - a strong encryption algorithm
  - a secret key known only to sender / receiver

$$Y = E_K(X) \text{ or } E(K,X)$$

$$X = D_K(Y) \text{ or } D(K,Y)$$

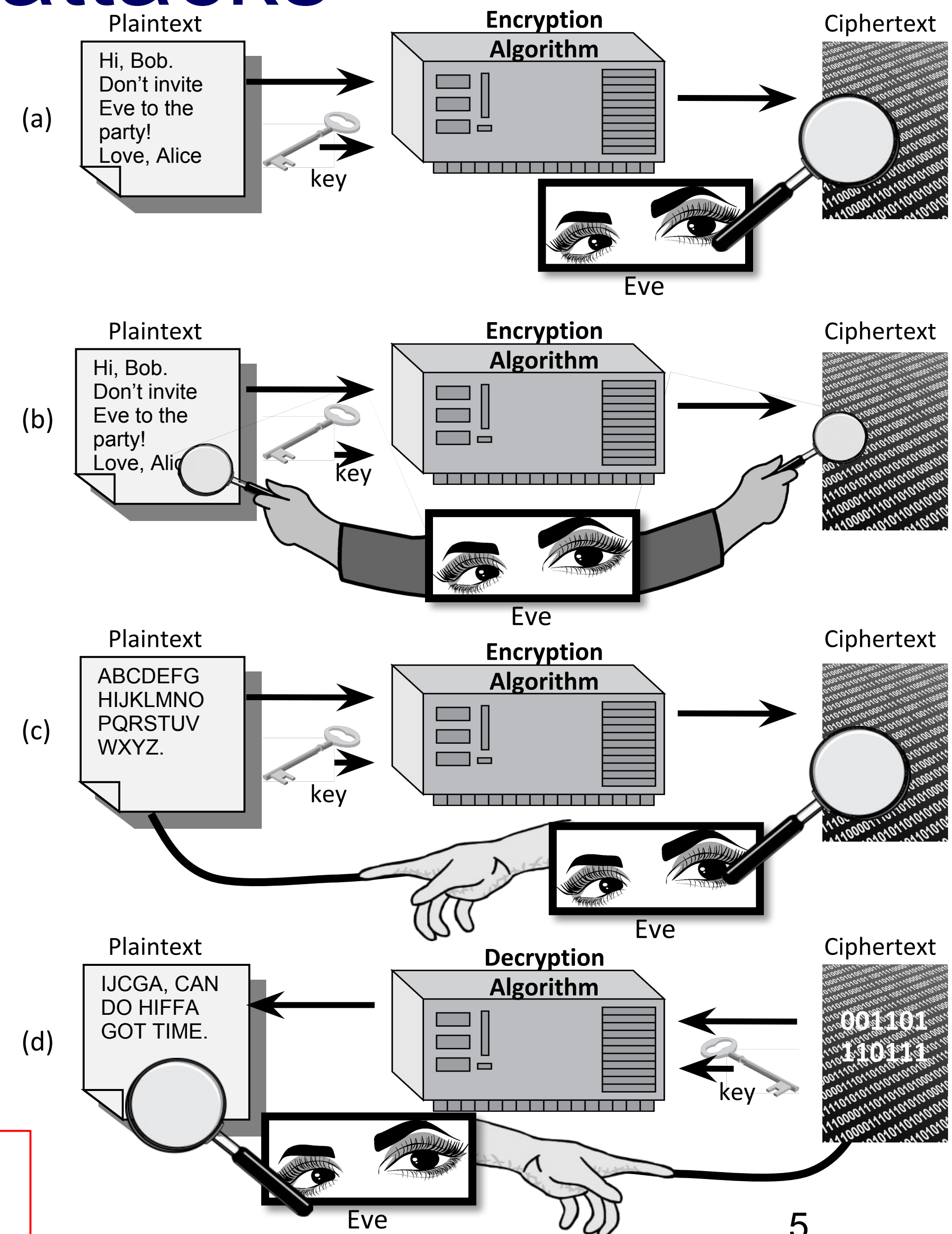
- assume encryption algorithm is known
  - Kerckhoffs's Principle: A cryptosystem should be secure even if everything about the system, except the key, is public knowledge
- imply a secure channel to distribute the key

Cryptographic attacks

- Attacker may have
  - a) collection of ciphertexts (ciphertext only attack)
  - b) collection of plaintext/ciphertext pairs (known plaintext attack)
  - c) collection of plaintext/ciphertext pairs for plaintexts selected by the attacker (chosen plaintext attack)
  - d) collection of plaintext/ciphertext pairs for ciphertexts selected by the attacker (chosen ciphertext attack)

COA ⊂ KPA ⊂ CPA ⊂ CCA

These attacks are also applicable to PKC;
The key is not known to the attacker



requirements for a secure cipher

- In cryptography, confusion and diffusion are two properties of the operation of a secure cipher [Claude Shannon]

- Diffusion hides the relationship between the ciphertext and the plaintext. For instance, if we change a character of the plaintext, then several characters of the ciphertext should change, and similarly, if we change a character of the ciphertext, then several characters of the plaintext should change

* avalanche effect

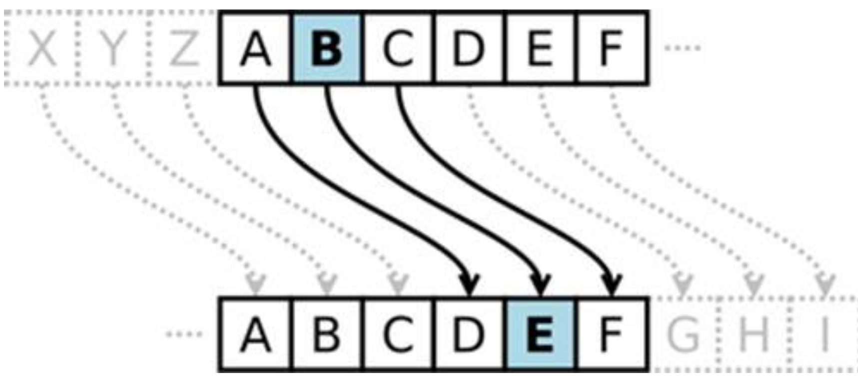
- Confusion hides the relationship between the ciphertext and the key. For instance, each character of the ciphertext should depend on several parts of the key

main primitives

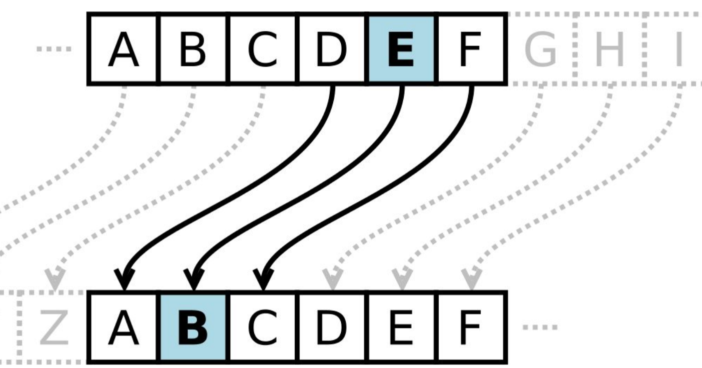
- Substitution
- Permutation/transposition
- exclusive-OR: $\oplus$

substitution (shift) cipher

- a method of encrypting by which units of plaintext are replaced with ciphertext, according to a fixed system
  - Units: characters, groups of characters...
- Caesar Cipher: E(x) = x+3 (mod 26)
  - monoalphabetic cipher



- then, how to decrypt? D(y)?
- is it secure?



source: https://en.wikipedia.org/wiki/Caesar_cipher

affine cipher

- another monoalphabetic substitution cipher
- for English alphabet, m is 26 below
- $E(x) = (ax+b) \bmod m$
- $D(y) = a^{-1}(y-b) \bmod m$

- how many keys?
  - 26 letters
  - 12 numbers for a, 26 numbers for b

- is it secure?

Monoalphabetic Substitution Cipher

- The key space: all permutations of $\Sigma = \{A, B, C, \ldots, Z\}$
- Encryption given a key π:
  - each letter X in the plaintext P is replaced with $\pi(X)$
- Decryption given a key π:
  - each letter Y in the ciphertext C is replaced with $\pi^{-1}(Y)$

Example:

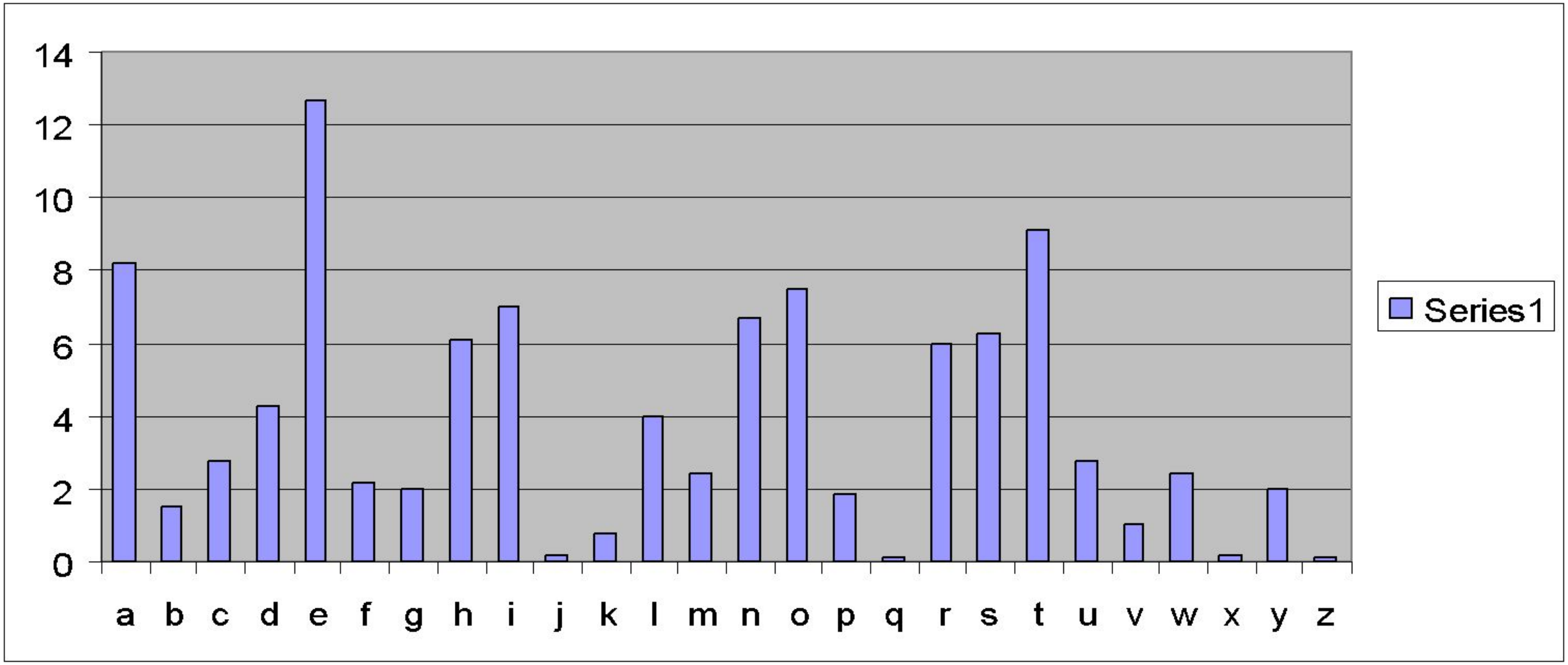
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
$\pi=$	C	A	D	B	Z	H	W	Y	G	O	Q	X	S	V	T	R	N	M	L	K	J	I	P	F	E	U

BECAUSE $\rightarrow$ AZDCJLZ

- how many keys?
- is it secure?

frequency analysis

- Frequency analysis is based on the fact that, in any given stretch of written language, certain letters and combinations of letters occur with varying frequencies.



another hints to guess

- Most common English bigrams (frequency in 1000 words)

th	he	an	re	er	in	on	at	nd	st	es	en	of	te	ed
168	132	92	91	88	86	71	68	61	53	52	51	49	46	46

Vigenère Cipher

polyalphabetic substitution

[A=0, B=1, …, Z=25], Z_n= {0, 1, …, n-1}, $P = C = Z_{26}$

Definition:

Given m (key length), and $K = (k_1, k_2, \ldots , k_m)$ a key,

Encryption:

$e_k(p_1, p_2 \ldots p_m) = (p_1+k_1, p_2+k_2, \ldots, p_m+k_m) \pmod{26}$

Decryption:

$d_k(c_1, c_2 \ldots c_m) = (c_1-k_1, c_2-k_2, \ldots, c_m- k_m) \pmod{26}$

Example:

Plaintext: C R Y P T O G R A P H Y

Key (m=4): L U C K L U C K L U C K

Ciphertext: N L A Z E I I B L J J I

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
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

Vigenère Cipher

- An attacker can figure out key length m?
  - the attacker has only ciphertext
- If the attacker finds the key length, then what happens?

Index of coincidence

Hill Cipher

- polyalphabetic substitution cipher based on linear algebra

$$\begin{pmatrix} C1 \\ C2 \\ C3 \end{pmatrix} = \begin{pmatrix} 9 & 18 & 10 \\ 16 & 21 & 1 \\ 5 & 12 & 23 \end{pmatrix} \begin{pmatrix} p1 \\ p2 \\ p3 \end{pmatrix} \pmod{26}$$

```
C1 =  9*p1 + 18*p2 + 10*p3 (mod 26)
C2 = 16*p1 + 21*p2 +  1*p3 (mod 26)
C3 =  5*p1 + 12*p2 + 23*p3 (mod 26)
```

- how to decrypt?

Hill cipher: A key is a matrix

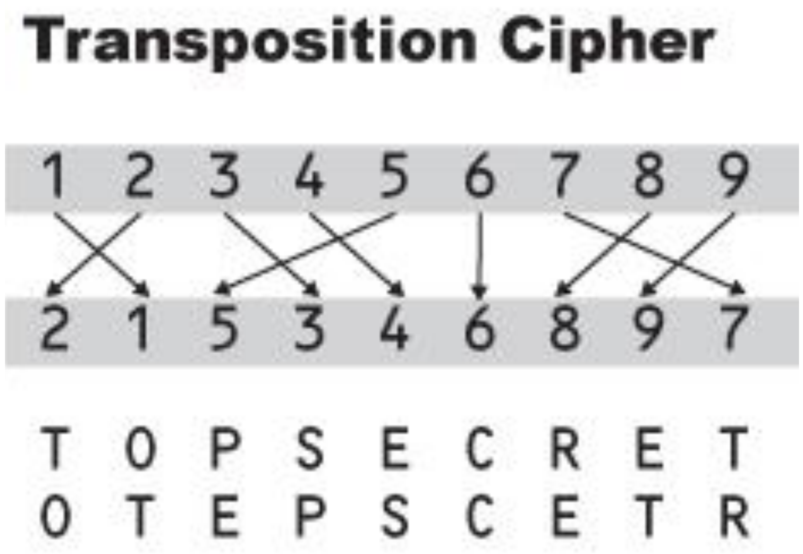
$$\begin{pmatrix} \mathbf{k11} & \mathbf{k12} & \mathbf{k13} \\ \mathbf{k21} & \mathbf{k22} & \mathbf{k23} \\ \mathbf{k31} & \mathbf{k32} & \mathbf{k33} \end{pmatrix}$$

- Generalize to any size, larger blocks
- matrix multiplication can provide diffusion
- Matrix must be invertible

- is it secure?
  - is vulnerable to a known-plaintext attack

Transposition cipher

- a method of encryption by which the positions held by units of plaintext are shifted according to a regular system
  - Units: characters, groups of characters...
- the ciphertext constitutes a permutation of the plaintext
  - aka permutation cipher



transposition cipher: a variant

columnar cipher

- Plaintext:

MESSAGE FROM MARY STUART KILL THE QUEEN

Plaintext in

4	9	1	7	5	3	2	8	6
M	E	S	S	A	G	E	F	R
O	M	M	A	R	Y	S	T	U
A	R	T	K	I	L	L	T	H
E	Q	U	E	E	N			

With 9 columns, we have 9! = 362,880 possible keys

Columns are then transposed

Ciphertext out

- Ciphertext: SMTUESLGYLNMOAEARIERUHSAKEFTTEMRQ

SMTUE SLGYL NMOAE ARIER UHSAK EFTTE MRQ

X-or($\oplus$) in cryptography

- Sender wants to send M to receiver
- M (plaintext): 1010
- K (Key): 0011
- C = M $\oplus$ K = 1001 (ciphertext)

1001
transmitted

- Receiver already knows K
- C$\oplus$K = (M$\oplus$K)$\oplus$K = 1001 $\oplus$ 0011 = 1010 = M

original message is restored!

* If an attacker knows M and C, can she know K?

A crucial property of X-or

- If Y has an arbitrary distribution over $\{0,1\}^n$
- And if X is indep. uniformly distributed over $\{0,1\}^n$
- Then $Z = X \oplus Y$ is also uniformly distributed

- Proof (n=1)

  P[Z=0] = ?

  P[Z=1] = ?

Y	P[Y]
0	p0
1	p1

X	P[X]
0	1/2
1	1/2

X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	0

X	Y	Z	P[Z]
0	0	0	p0*1/2
0	1	1	p1*1/2
1	0	1	p0*1/2
1	1	0	

one time pad (OTP)

- The one-time pad, which is a provably secure cryptosystem, Gilbert Vernam in 1918.
  - aka Vernam cipher
- The message is represented as a binary string (a sequence of 0's and 1's using a coding mechanism such as ASCII coding.
- The key is a truly random sequence of 0's and 1's of the same length as the message.
- The encryption is done by adding the key to the message modulo 2, bit by bit. This process is often called exclusive or, XOR ($\oplus$)

OTP: Example

- message ='IF'
- then its ASCII code =(1001001 1000110)
- key = (1010110 0110001)
- *Encryption:*
  - 1001001 1000110 plaintext
  - 1010110 0110001 key
  - 0011111 1110111 ciphertext
- *Decryption:*
  - 0011111 1110111 ciphertext
  - 1010110 0110001 key
  - 1001001 1000110 plaintext

OTP problems

- Key should be as long as plaintext
  - key should not be reused
- Key distribution & Management difficult

Before talking about perfect secrecy

$|K| = |M| = |C|$

- A cipher (E, D) is defined over (K, M, C)
  - *E: encryption alg., D: decryption alg.*
  - *Key space K, message space M, ciphertext space C*
- *P(M=m)* is what the adversary believes the probability that the plaintext is *m*, before seeing the ciphertext
  - Maybe they are very sure, or maybe they have no idea
- *P(M=m | C=c)* is what the adversary believes after seeing that the ciphertext is *c*
- *P(M=m | C=c) = P(M=m)* means that after knowing that the ciphertext is *c*, the adversary's belief does not change
  - Intuitively, the adversary learned **nothing** from the ciphertext

Perfect secrecy

- Basic idea: ciphertext should reveal no information about plaintext
- If an algorithm offers perfect secrecy then:
  - For a given ciphertext c, all possible corresponding plaintexts are possible decryptions
- Def. perfect secrecy of a cipher

A cipher (E, D) over (K, M, C) has perfect secrecy if

$P[E(k, m_0) = c] = P[E(k, m_1) = c]$ for all $m_0, m_1 \in M$, $c \in C$

where k is uniform in K

* $P[E(k, m_0)=c] = P[E(k, m_1)=c] \leftrightarrow P[M=m|C=c] = P[M=m]$

OTP has perfect secrecy

- For all m and c,

$P[E(k, m) = c] = \#\{k \in K, \text{ s.t. } E(k, m)=c]\} / |K|$

$P[E(k, m_0) = c] = 1/|K|$

$P[E(k, m_1) = c] = 1/|K|$

Condition for Perfect secrecy-1

- In any perfectly-secure encryption scheme defined by (E,D), the key space K must be at least as large as the message space M, i.e. $|K| \geq |M|$, or key-length ≥ msg-length.

- Proof. Assume $|K| < |M|$. Also assume that the message space has such a distribution wherein every message occurs with non zero probability.

- Let $c \in C$ be a ciphertext that occurs with non-zero probability. Define a new set M(c) which contains all possible messages that are decryptions of c, i.e.,
- $M(c) := \{m | m = Dec(k,c) \text{ for some } k \in K\}$.

Condition for Perfect secrecy-2

- Thus clearly, $|M(c)| \leq |K|$ since for each message there will be at least one key $k \in K$ for which $m = Dec(k,c)$ since Dec() is deterministic
- Also $|K| < |M|$ from our assumption.
- Together they indicate that there exists at least one message $m_0 \in M$ which cannot be encrypted by any key, i.e. $m_0 \notin M(c)$.
- But then $P[M = m_0 \mid C = c] = 0 \neq P[M = m_0]$, which contradicts Def. of perfect secrecy and implies that the scheme is not perfectly secret.

"Two time pad" is insecure

- Do not use the same key twice!

Two Time Pad:

$c_1 = m_1 \oplus k$

$c_2 = m_2 \oplus k$

Enough redundancy in ASCII (and english) that $m_1 \oplus m_2$ is enough to know m_1 and m_2

Eavesdropper gets c_1 and c_2.
What is the problem?

$$\boxed{c_1 \oplus c_2 = m_1 \oplus m_2}$$

Two types of symmetric ciphers

- Stream cipher
  - Encrypts one bit/byte at a time
    - mimicking OTP
  - Typically faster
  - Generating a “random” keystream is difficult
  - do not provide integrity/authentication
  - e.g. RC4
- Block cipher
  - Encrypts a block of bits at a time
  - Usually have feedback between blocks, errors can be propagated
  - Some block ciphers can provide integrity/authentication
  - e.g. DES, AES

Stream cipher

- Try to make OTP practical
- Start with a secret key (“seed”)
  - Generate a keying stream
  - PRG is a function: Gen:$\{0,1\}^s \rightarrow \{0,1\}^n$
- Combine the keystream with the plaintext to produce the ciphertext (e.g. XOR)

* PR(N)G: pseudo random (number) generator

Key (seed)

Gen

Keystream

$\oplus$

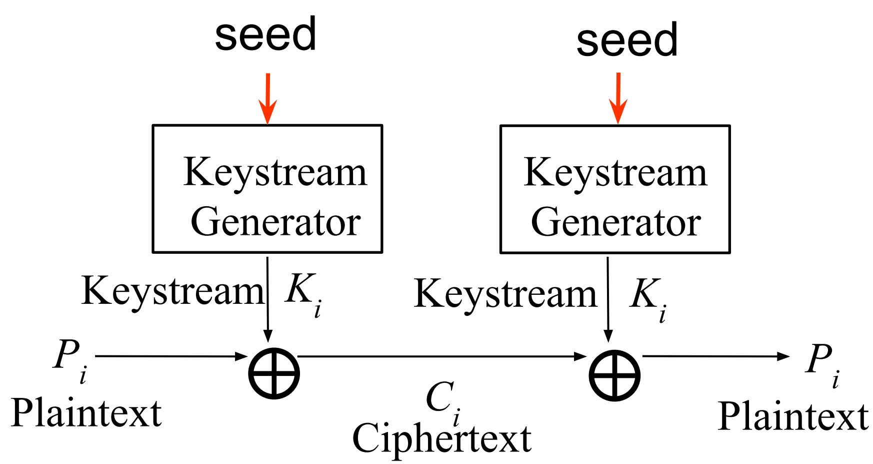
Plaintext

=

Ciphertext

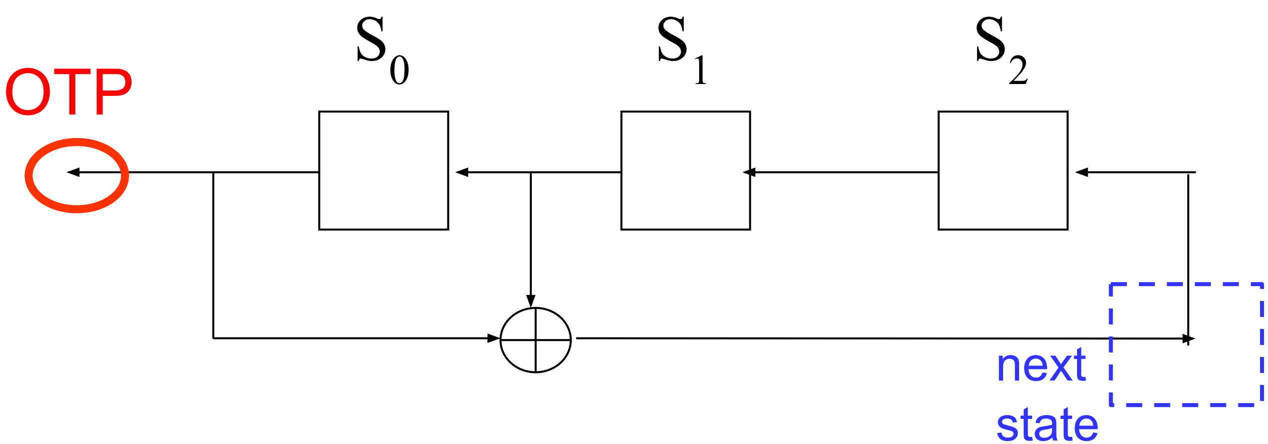
stream cipher

- Stream cipher does not have perfect secrecy
  – Key length is shorter than message length
- Its security depends on PRG (or PRNG)
- Stream cipher is a keystream generator



stream cipher: LFSR

- linear feedback shift register (LFSR)



$$s_{t+3} = s_{t+1} + s_t$$

S_0	S_1	S_2
0	0	1
0	1	0
1	0	1
0	1	1
1	1	1
1	1	0
1	0	0

----

0 0 1

Initial fill determines the sequence of states: seed

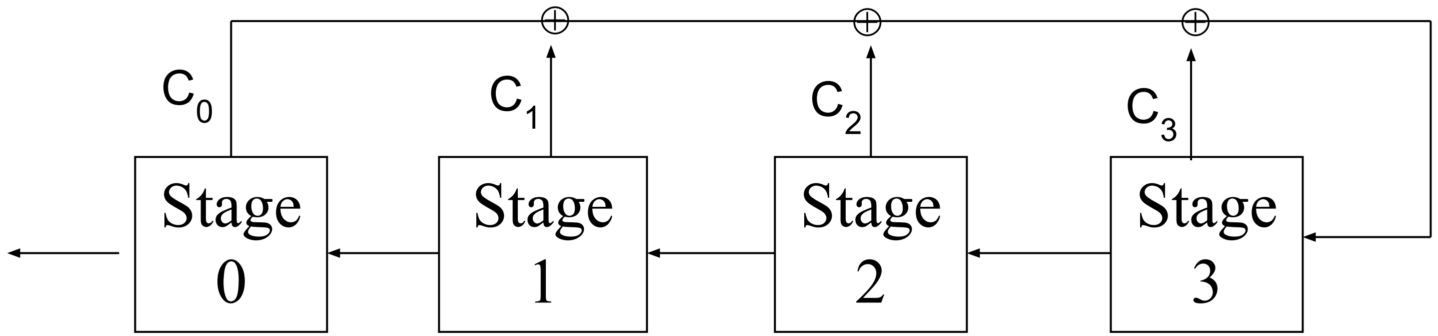
Generates a periodic sequence: 0010111...

Maximal period 2^3-1=7

n: # of states → Period: 2^n -1

Cryptanalysis of LFSR

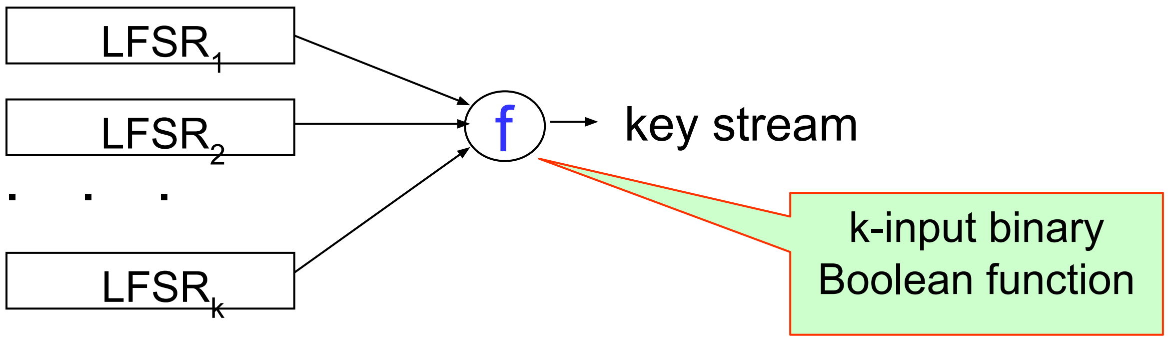
c_i can be 1 or 0 (no link/xor)



- Given a 4-stage LFSR, we know
  - $z_4 = z_3c_3 \oplus z_2c_2 \oplus z_1c_1 \oplus z_0c_0 \bmod 2$
  - $z_5 = z_4c_3 \oplus z_3c_2 \oplus z_2c_1 \oplus z_1c_0 \bmod 2$
  - $z_6 = z_5c_3 \oplus z_4c_2 \oplus z_3c_1 \oplus z_2c_0 \bmod 2$
  - $z_7 = z_6c_3 \oplus z_5c_2 \oplus z_4c_1 \oplus z_3c_0 \bmod 2$
- Knowing $z_0, z_1, \ldots, z_7$, one can compute c_0, c_1, c_2, c_4.
- In general, knowing 2n output bits, one can solve n-stage LFSR

Stream cipher from LFSRs

- Combine multiple LFSRs



Desirable properties of f:

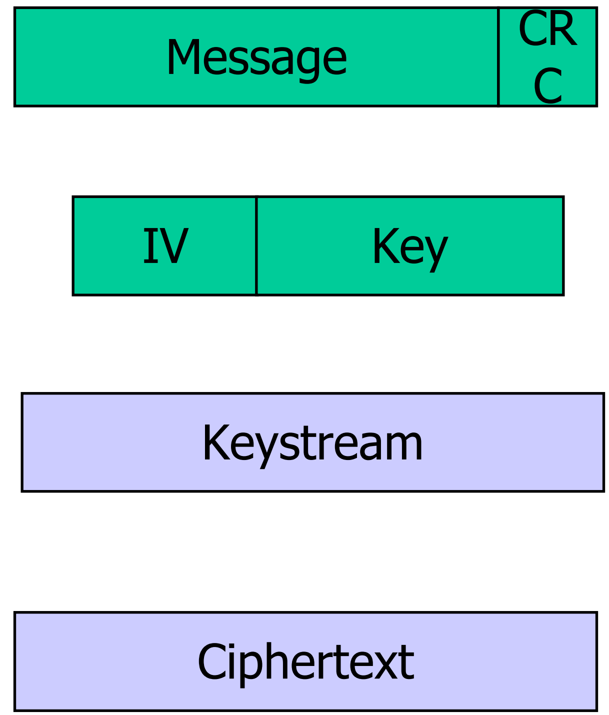
- high non-linearity
- long "cycle period" $(\sim 2^{n1+n2+...+nk})$
- low correlation with the input bits

Case study: WiFi WEP

- WiFi 802.11b WEP
  - WEP is Wired Equivalent Privacy
  - Link-layer encryption
  - Defined in the IEEE 802.11b standard
  - It misuses the stream cipher RC4

WEP: the overall encryption

- Message: What you’re encrypting
- CRC: To verify the integrity of the message
- Plaintext: message + its CRC
- Initialization vector (IV): A 24-bit number which plays two roles (detailed soon)
- Key: A 104-bit number which is used to build the keystream
- Keystream: What is used to encrypt the plaintext
- Ciphertext: keystream$\oplus$(msg||CRC(msg))



||: concatenation

WEP encryption step-by-step

Message	CRC

sender

Step 1: Compute CRC for the message

- CRC-32 polynomial is used

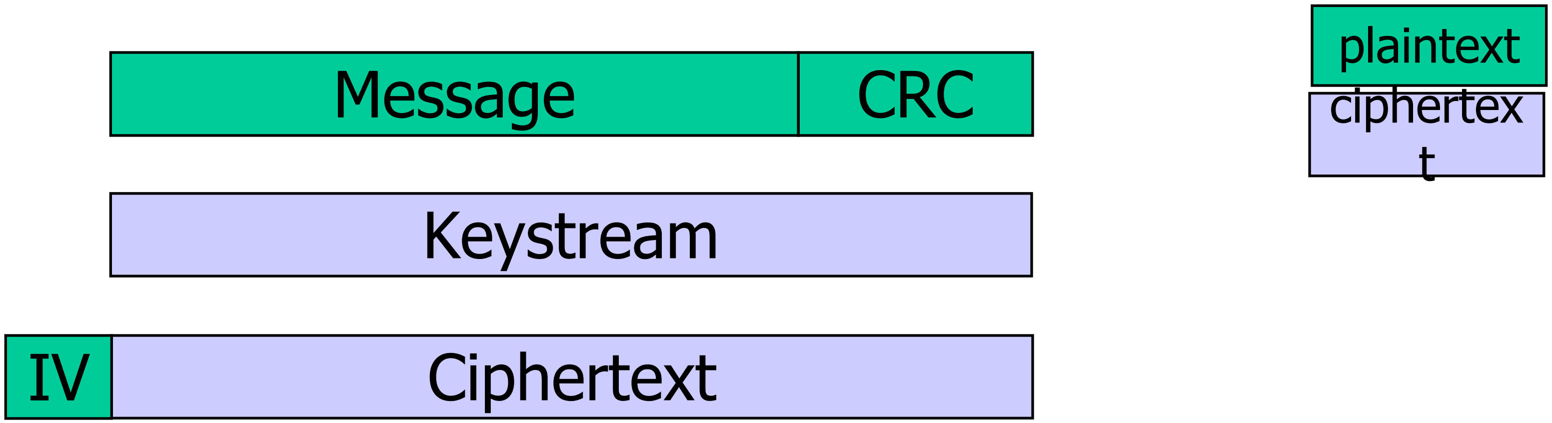
WEP encryption step-by-step

Keystream

Step 2: Compute the keystream

- IV (24bits) is concatenated with the key (104bits)
- RC4 Key Generation algorithm is used on 128 bit concatenation

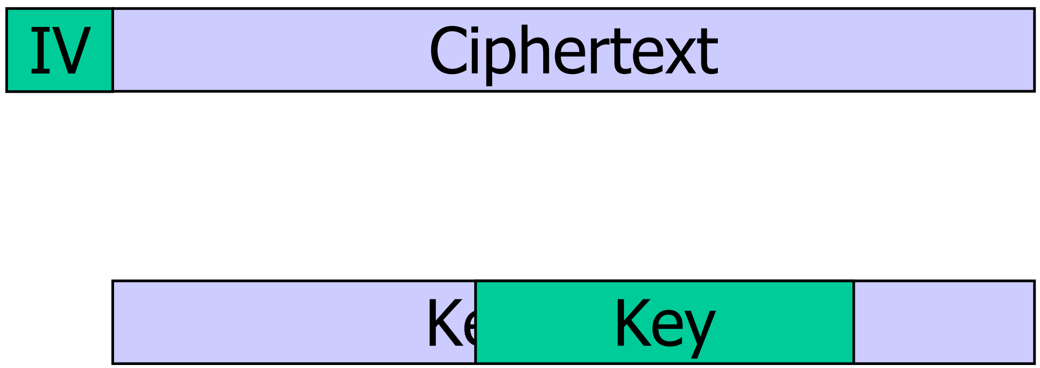
WEP encryption step-by-step



Step 3: Encrypt the plaintext

- The plaintext is XORed with the keystream to form the ciphertext
- The IV is prepended to the ciphertext
  - It is not encrypted

WEP decryption step-by-step

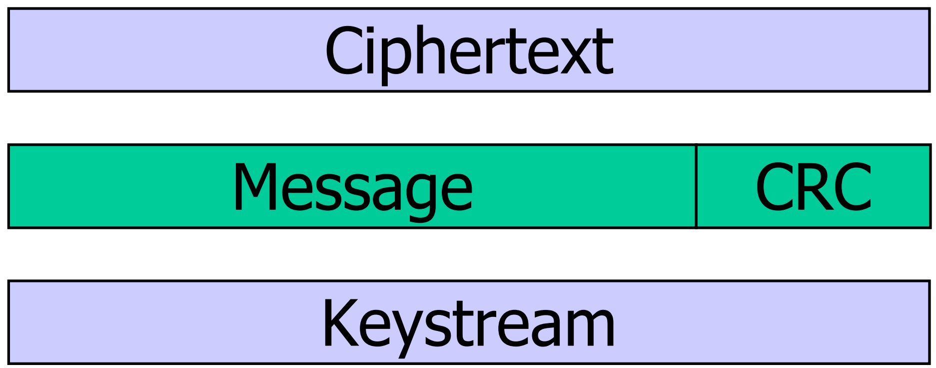


receiver

Step 1: Build the keystream

- Extract the IV from the incoming frame
- Prepend the IV to the key (already given)
- Use RC4 to build the keystream

WEP decryption step-by-step



Step 2: Decrypt the plaintext and verify

- XOR the keystream with the ciphertext
- Verify the extracted message with the CRC

Initialization vector (IV)

- It’s carried in plaintext
- It’s only 24 bits!
- IV must be different for every message transmitted.
- 802.11 standard doesn’t specify how IV is calculated.
  - There are no restrictions on IV reuse!
  - Usually simply increment by 1 for each frame

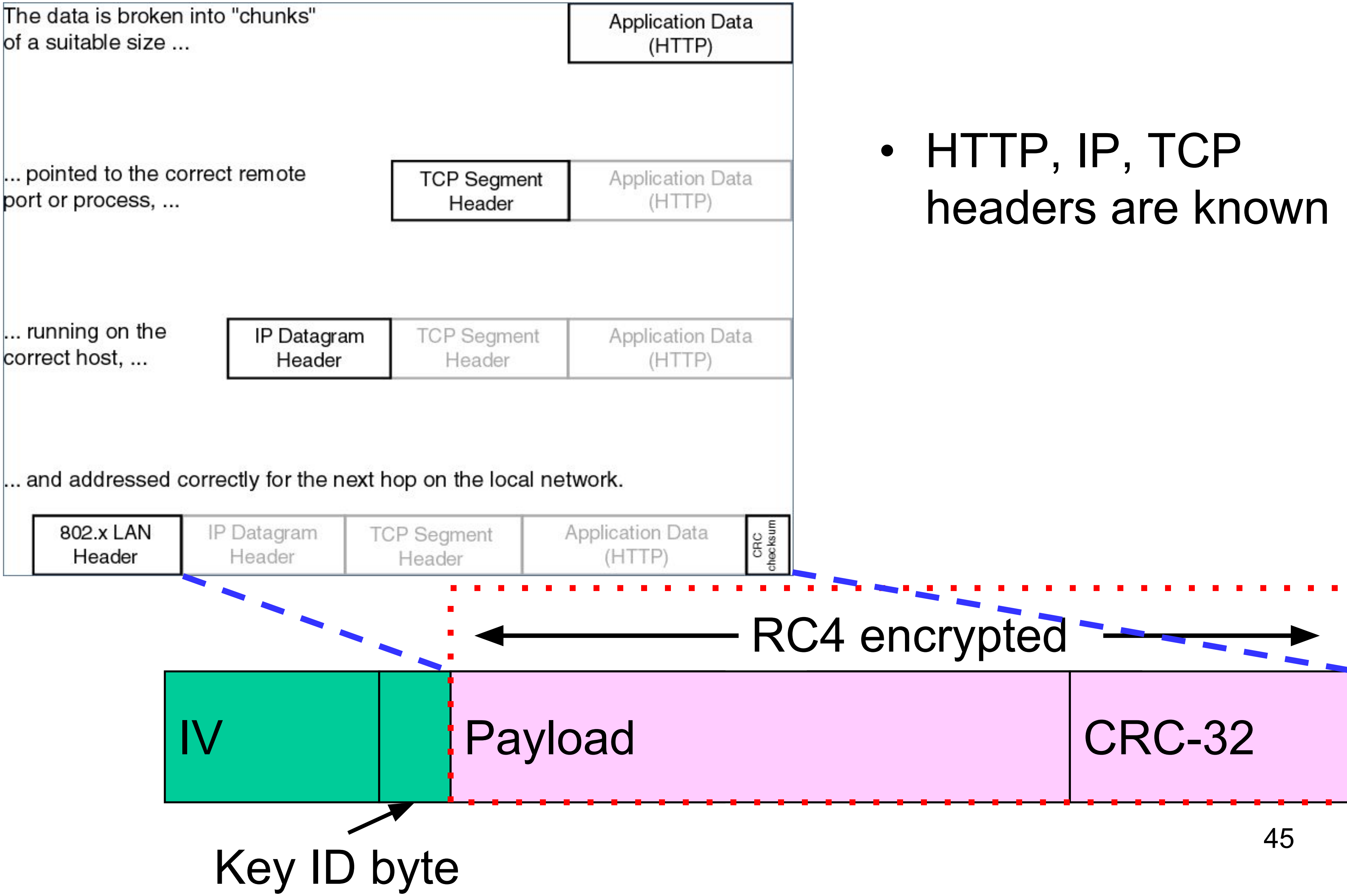
IV collision

- What if two messages use the same IV?
- Key is fixed, the period of IV is 2^24
  - Suppose 2^24 frames are captured
- Same IV ⇒ same keystream!
- $C1 \oplus C2 = P1 \oplus P2$
- If P1 is known, P2 is immediately available
- Otherwise, use expected distribution of P1 and P2 to discover contents
  - Much of network traffic contents predictable
  - Easier when three or more packets collide

$C1 = P1 \oplus \text{RC4-Gen}(IV1||key)$
$C2 = P2 \oplus \text{RC4-Gen}(IV2||key)$
$C1 \oplus C2 = P1 \oplus P2$

* In some implementations, IV is reset to 0 when reboot

Some part of plaintext is already known



- HTTP, IP, TCP headers are known

CRC algorithm

- The CRC is a linear function
  - $crc(a \oplus b) = crc(a) \oplus crc(b)$
- The CRC is an unkeyed function

Message modification

- Uses CRC-32 checksum
  - Good for detecting random error
  - Bad for malicious
- CRC-32 is linear:
  - $CRC(A \oplus B) = CRC(A) \oplus CRC(B)$
- RC4 is transparent to XOR
  - $C = RC4(M,CRC(M)) = S \oplus (M,CRC(M))$
  - $C' = C \oplus (X,CRC(X))$

    $= S \oplus (M,CRC(M)) \oplus (X,CRC(X))$

    $= RC4\ (M \oplus X,\ CRC(M \oplus X))$

* S is keystream