Classic Symmetric Cryptography

Demetrios Glinos

University of Central Florida

CIS3360 - Security in Computing

Readings

- "Computer Security: Principles and Practice", 3rd Edition, by William Stallings and Lawrie Brown
  - Section 2.1
- [OPTIONAL] Cryptography and Network Security: Principles and Practice, Sixth Edition, William Stallings, Pearson – Prentice Hall, 2014, ISBN-13: 978-0-13-335469-0.
  - Sections 2.2 and 2.3

For More Information

- You do NOT need to get a copy of Stallings' cryptography text to find out more about the algorithms presented in this lecture.
- Instead, you can find a great deal of information online about each of the algorithms.
- HOWEVER, different authors sometimes use different variations on the algorithms.
- THEREFORE, if what you find is different from what is presented in these slides, you should go with what is on these slides (because that is what you will be tested on).
  - Example: We generally pad with “x”, but real systems often use random letters
  - Example: For Playfair, we combine “I” and “J”, but other authors drop “Q” instead; also, special case rules are sometimes different from what we use

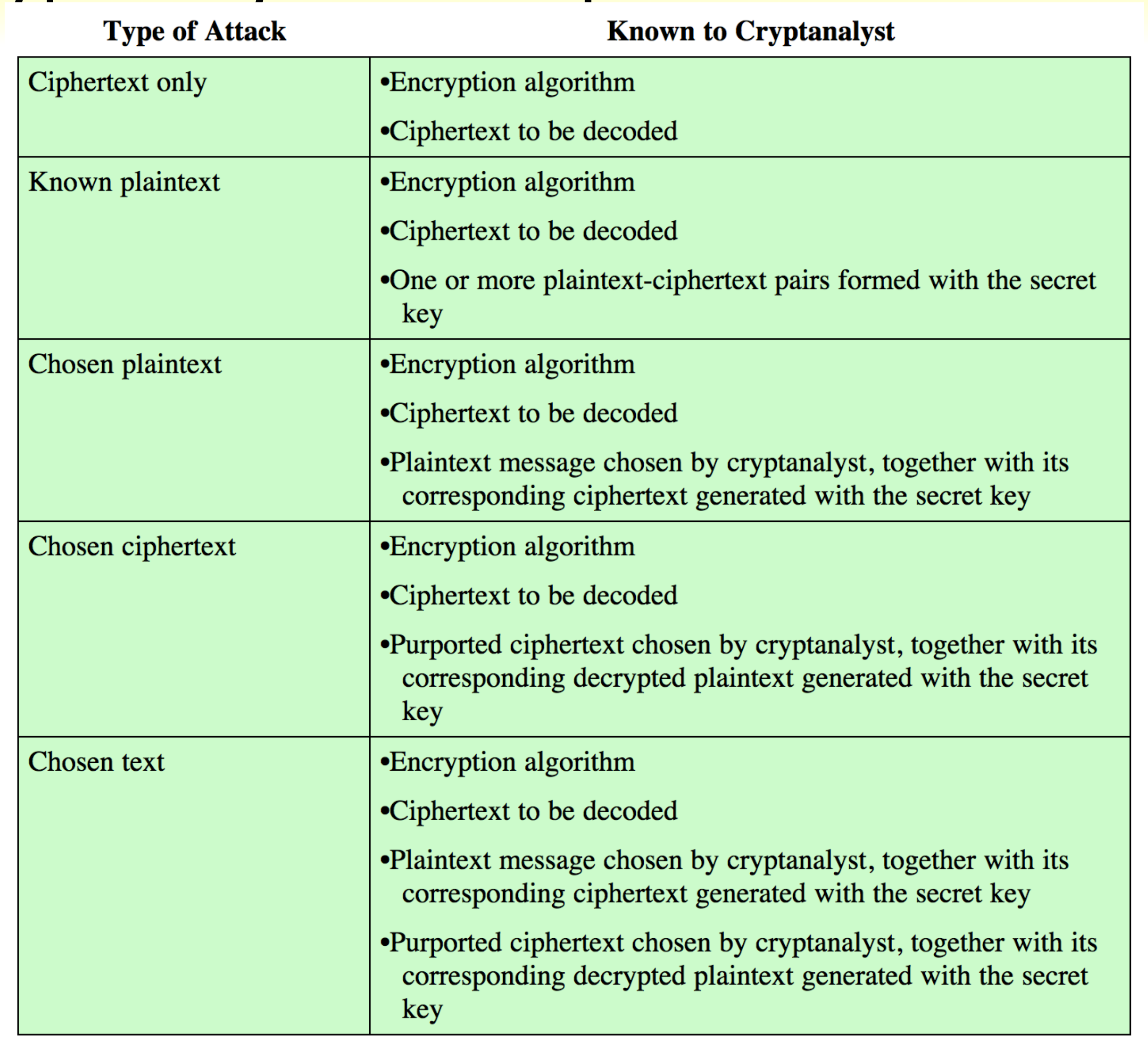
Outline

Cryptographic Terminology

- **plaintext** - original message
- **ciphertext** – the encrypted message
- **cipher** – the encryption algorithm for transforming plaintext to ciphertext
- **key(s)** - info used by cipher algorithm, should be known only to sender/receiver
- **encipher (encrypt)** - converting plaintext to ciphertext
- **decipher (decrypt)** - recovering plaintext from ciphertext
- **cryptography** - study of encryption principles/methods
- **cryptanalysis (codebreaking)** - study of principles/ methods of deciphering ciphertext *without* knowing the key
- **cryptosystem** – a set of keys, algorithms, and specific language that form a system for encrypting and decrypting text

Cryptanalysis Concepts

- Five basic types of attacks, distinguished by what is available to the cryptanalyst and when:



Type of Attack	Known to Cryptanalyst
Ciphertext only	•Encryption algorithm •Ciphertext to be decoded
Known plaintext	•Encryption algorithm •Ciphertext to be decoded •One or more plaintext-ciphertext pairs formed with the secret key
Chosen plaintext	•Encryption algorithm •Ciphertext to be decoded •Plaintext message chosen by cryptanalyst, together with its corresponding ciphertext generated with the secret key
Chosen ciphertext	•Encryption algorithm •Ciphertext to be decoded •Purported ciphertext chosen by cryptanalyst, together with its corresponding decrypted plaintext generated with the secret key
Chosen text	•Encryption algorithm •Ciphertext to be decoded •Plaintext message chosen by cryptanalyst, together with its corresponding ciphertext generated with the secret key •Purported ciphertext chosen by cryptanalyst, together with its corresponding decrypted plaintext generated with the secret key

source: Table 20.1

Cryptanalysis Concepts (cont'd)

- An encryption scheme is ***computationally secure*** if the ciphertext generated by the scheme satisfies either or both of these conditions:
  - ***cost of breaking*** the cipher exceeds the value of the encrypted information
  - ***time to break*** the cipher is longer than the useful lifetime of the information

- ***Unicity distance***
  - Different for each cryptosystem
  - It is the minimum number of ciphertext characters needed to have a unique plaintext map to it
  - It is a consequence of the built-in redundancy of every natural language
  - Cipher is broken when a readable text is recovered
  - For the strong AES symmetric cipher, it is only about 38 characters!
    - But note: having 38 characters of AES-encrypted text tells us that there is a unique AES key that produced it, but it does not tell us what the key is

Symmetric Encryption Context

- Symmetric cryptography
  - Also known as: *conventional or single-key encryption*
  - The sender and recipient share a common key
  - All classical encryption algorithms are symmetric
  - It was the only type of algorithm prior to the invention of public-key cryptosystems in 1970s
  - Still by far the most widely used
- Requirements
  - a strong encryption algorithm
  - a secret key known only to sender / receiver
  - ***security by obscurity*** approach (hoping the algorithm is unknown to the attacker) is unreliable
    - we should assume the attacker knows the encryption algorithm

    → **Security depends on the secrecy of the key, not the the algorithm**
  - Must have a secure means for distributing the key

Mathematical Notation

- Encryption and decryption are ***functions*** in the mathematical sense
- We use this notation for encrypting plaintext message M using encryption algorithm E with key k:

$$C = E_K(M) \quad \text{or} \quad C = E(k, M)$$

- And similarly for decrypting ciphertext C using decryption algorithm D with key k:

$$M = D_K(C) \quad \text{or} \quad M = D(k, C)$$

NOTE: When it is clear from the context which key is involved, we often use the notation **$C = E(M)$ and $M = D(C)$** for convenience

Substitution Ciphers

- A **substitution cipher** is a class of cipher algorithms in which:
  - **Letters** of plaintext are **replaced** by other letters or by numbers or symbols
    - Example: Caesar cipher
  - Or, if the plaintext is viewed as a sequence of bits, then substitution involves replacing plaintext **bit sequences** with ciphertext bit sequences of the same length

- **How complicated can this be?**
  - Assume the English language and only letter substitutions (no numbers, etc.)
  - Then,
    - 26 choices for substitution for “a”
    - 25 choices for substitution for “b” since we can’t use what we chose for “a”
    - 24 choices for “c”, etc.

  → 26! Choices overall: ≈ 4.03×10^{26} = 403,000,000,000,000,000,000,000,000

Caesar Cipher

- A type of substitution cipher known as a ***shift cipher*** ← *every letter is shifted by the same amount*
  - Only 26 possible keys (including the zero shift)
- Earliest known substitution cipher, used by Julius Caesar
- First known use in military affairs
- Replaces each letter by 3rd letter following it, modulo the alphabet size
- Example: **meet me after the toga party**
  PHHW PH DIWHU WKH WRJD SDUWB
- Mathematically, we use modular arithmetic, as follows:
  - Assign numbers 0, 1, 2, ..., 25 to the letters a, b, c, ..., z
  - To encrypt/decrypt, we convert letters to numbers, compute, and then convert the resulting numbers back to letters
  - To encrypt, for each letter *m* we substitute $E(m) = (m + 3) \bmod 26$
  - To decrypt, for each letter *m* we substitute $D(m) = (m - 3) \bmod 26$
  - *Examples:* $E(\text{“g”}) = E(6) = (6 + 3) \bmod 26 = 9 \bmod 26 = 9 = \text{“j”}$
    $E(\text{“y”}) = E(24) = (24 + 3) \bmod 26 = 27 \bmod 26 = 1 = \text{“b”}$
    $D(\text{“j”}) = D(9) = (9 - 3) \bmod 26 = 6 \bmod 26 = \text{“g”}$

Keyword Cipher

- Use the keyword PROGRAM to encrypt the message "program".
- Procedure:
  - Lay out keyword and ***ignore duplicate letters*** (e.g., the second 'R' in program)
  - Finish out the mapping with the letters of the alphabet that were not used, using the standard ordering of the letters
- Plaintext and ciphertext alphabets are shown below:

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

Plaintext: party

Ciphertext: LPQTY

Number of possible ciphers is 26! / (26 – n)! *where n = keyword length*

For n = 6 (as above), this number is (26)(25)(24)(23)(22)(21) = 165,765,600

Monoalphabetic Cipher

- Instead of just *shifting* the alphabet we could *shuffle* (jumble) the letters arbitrarily
- Each plaintext letter maps to a different random ciphertext letter
- This is essentially a keyword cipher with a key that is 26 letters long
- Example

Plain:	ab	cd	ef	g h i	j k l	mno	pqr	s t	uv	wx	yz
Cipher:	DK	VQ	FI	BJW	PES	CXH	TMY	AU	OL	RG	ZN

Plaintext: i f w e w i s h t o r e p l a c e l e t t e r s

Ciphertext: WIR F RWAJUHYFT SDV FSFUUFYA

- Number of possible ciphers is 26! ≅ 4.03×10^{26}
- susceptible to *letter frequency analysis*

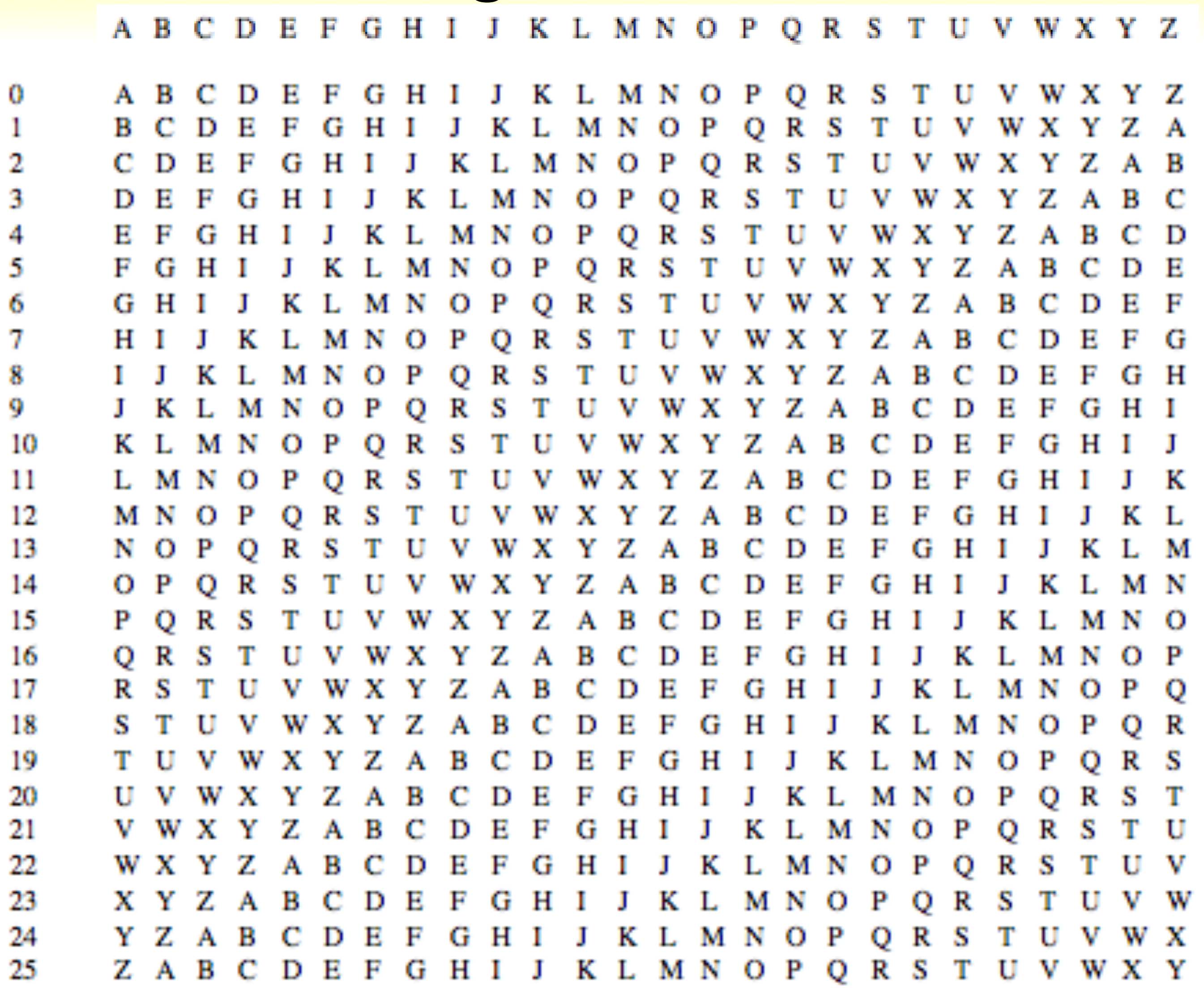
Polyalphabetic Ciphers

- A ***polyalphabetic cipher*** is a substitution cipher that
  - does not use a one-to-one (1:1) correspondence between plaintext letter and its corresponding ciphertext letter
  - Uses *multiple substitution alphabets* AND *switches among them systematically*
  - The same plaintext letter will generally be encrypted differently each time it appears (one-to-many function)
  - Similarly, the same ciphertext character will generally represent multiple plaintext characters
- The **Vigenère Cipher** is the best-known example
- The famous Enigma Cipher from World War II is a well-known more complex example

Vigenère Cipher

- simplest polyalphabetic substitution cipher
- uses multiple shift ciphers (in the form of the Vigenère table)
- *key* is multiple letters long **$K = k_1\ k_2 \ldots k_d$ ← d = key length**
- each letter of key specifies a different row of Vigenère table to use for substitution
- *Vigenere table* is simply a table with all possible *shifts*
- use the rows (alphabets) specified by the key letters sequentially
- repeat from start after ***d*** letters in message (here, we reuse keyword, unlike the keyword cipher)
- decryption simply performs same operations in reverse

Vigenère Table



		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	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	← shift 0
	1	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	A	← shift 1
	2	C	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	← shift 2
Caesar →	3	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	.
	4	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	D	.
	5	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	.
	6	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	
	7	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	
	8	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	
	9	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	
	10	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	
	11	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	
	12	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	
	13	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	
	14	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	
	15	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	
	16	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	
	17	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	
	18	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	
	19	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	
	20	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	.
	21	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	.
	22	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	.
	23	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	← shift 23
	24	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	← shift 24
	25	Z	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	← shift 25

Vigenère Cipher: Original Procedure

- This is the original procedure without using mathematics, using the Vigenère table
  - Write out the plaintext
  - write the keyword above it, repeated as needed, trimming off any excess
  - use each key letter as index to row of Vigenère table
  - find the column in the table for the plaintext letter
  - encrypt using the letter in the index row

- Example, using the keyword "*deceptive*"

key:	deceptivedeceptivedeceptivede
plaintext:	wearediscoveredsaveyourselves
ciphertext:	ZICVTWQNGRZGVTWAVZHCQYGLMGZHW

- Note: "e" maps to {I,T,G,H,M} and "G" represents {c,e,r,l}

Vigenère Cipher: Using Modular Arithmetic

D	E	C	E	P	T	I	**V**	E	D	E	C	E	P	T	**I**	V	E	D	**E**	C	E	**P**	**T**	I	**V**	E	D	E
3	4	2	4	15	19	8	**21**	4	3	4	2	4	15	19	**8**	21	4	3	**4**	2	4	**15**	**19**	8	**21**	4	3	4
W	E	A	R	E	D	I	**S**	C	O	V	E	R	E	D	**S**	A	V	E	**Y**	O	U	**R**	**S**	E	**L**	V	E	S
22	4	0	17	4	3	8	**18**	2	14	21	4	17	4	3	**18**	0	21	4	**24**	14	20	**17**	**18**	4	**11**	21	4	18
25	8	2	21	19	22	16	**13**	6	17	25	6	21	19	22	**26**	21	25	7	**2**	16	24	**6**	**11**	12	**6**	25	7	22
Z	I	C	V	T	W	Q	**N**	G	R	Z	G	V	T	W	**A**	V	Z	H	**C**	Q	Y	**G**	**L**	M	**G**	Z	H	W

key:	deceptivedeceptivedeceptivede
plaintext:	wearediscoveredsaveyourselves
ciphertext:	ZICVTWQNGRZGVTWAVZHCQYGLMGZHW

- Note: “e” maps to {I,T,G,H,M} and “G” represents {c,e,r,l}
- **Note: columns in red indicate where sum was reduced to get modulo value**

One-Time Pad

- The **one-time pad** is also known as a ***Vernam cipher***, named after one of its inventors
- Early implementation: keys were distributed on "pads" of paper
- **Essentially, a Vigenère cipher, BUT:**
  - **Keyword length *as long or longer* than the entire plaintext message; AND**
  - **Each shift amount must be completely random**
- If properly used, this is ***unbreakable***
  - Reason: Since no repetition and random, there are no patterns to exploit
  - Very difficult to use properly, since one inevitably runs out of pad
- The key must **never** be used more than once

Binary One-Time Pad

- Operates on bit-level representations of plaintext and ciphertext
- Takes advantage of the symmetry of the **exclusive-or (XOR)** operator:

$C = M \oplus P$ and $M = C \oplus P$

where P is the pad (the key)

- **Exclusive-or (XOR)** table:

a	b	$c = a \oplus b$
0	0	0
0	1	1
1	0	1
1	1	0

Exclusive-or rule: "one or the other, but not both"

Binary One-Time Pad Example

- Let M be the word “IF”, whose ASCII code is: 1001001 1000110
- Let pad P (the key) be the random bit pattern: 1010110 0110001

- **Encryption:**
  - Plaintext 1 0 0 1 0 0 1 1 0 0 0 1 1 0
  - Key 1 0 1 0 1 1 0 0 1 1 0 0 0 1
  - Ciphertext 0 0 1 1 1 1 1 1 1 1 0 1 1 1

- **Decryption:**
  - Ciphertext 0 0 1 1 1 1 1 1 1 1 0 1 1 1
  - Key 1 0 1 0 1 1 0 0 1 1 0 0 0 1 ← same key as above
  - Plaintext 1 0 0 1 0 0 1 1 0 0 0 1 1 0 ← plaintext recovered

Pseudo-Random Number Generators

- Random values are needed for many things, including secret keys (as in the one-time pad and other ciphers)
- The problem is, *computers can't generate truly random numbers.*
- Cryptographers use computers to generate **pseudo-random numbers** that behave randomly in the statistical sense (uniformly distributed, unbiased).
- A **pseudo-random number generator (PRNG)** is a computer algorithm for generating pseudo-random numbers
- One algorithm for doing this is the ***linear congruential generator (LCG)***, which is defined as follows:

$$\boxed{\begin{array}{l} x_0 = \text{initial value (the "seed")} \\ x_{i+1} = (a x_i + b) \bmod n \end{array}}$$

Where ***n*** is some large number, ***a*** is in the range [1,n-1] and relatively prime to n, and ***b*** and ***x_0*** are numbers in the range [0,n-1]

- **A linear congruential generator is not, however cryptographically secure.**
- More secure, however, is to use a strong encryption algorithm, such as DES or AES, to encrypt any stream that starts from a random number.

Linear Congruential Generator Example

- LCG form:

  x_0 = initial value (the "seed")

  $x_{i+1} = (ax_i + b) \bmod n$

- Example:

  Let $n = 1823$, $a = 17$, $b = 248$ and $x_0 = 362$

  Then $x_1 = (17 * 362 + 248) \bmod 1823$

  $= (6154 + 248) \bmod 1823$

  $= 6402 \bmod 1823$

  $= 933$

We repeat this calculation using the value of x_1 instead of x_0, obtaining $x_2 = 1525$

Similarly, we obtain $x_3 = 651$, $x_4 = 377$, $x_5 = 1188$, etc.

Playfair Cipher

- Recall: not even the large number of keys in a monoalphabetic cipher provides security
  - reason: letter frequency analysis
- another polyalphabetic approach to improving security was to encrypt multiple letters as a single unit
- the **Playfair Cipher (aka Playfair Square)** is an example
- invented by Charles Wheatstone in 1854, but named after Lord Playfair who promoted it
- a manual symmetric encryption technique that encrypts two letters at a time (*bigrams*)
- reasonably fast and more secure than simple substitution
  - Reason: there are 676 bigrams (2-letter combinations), versus 26 letters

Playfair Key Matrix

- A 5 x 5 matrix of letters based on a keyword
- Fill in letters of keyword (ignoring duplicate letters)
- Fill rest of matrix with other letters (but combine I/J)

- Example, using the keyword MONARCHY:

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

Note: to get 25 entries, we combine I/J but some other authors just drop Q

Playfair Encrypting and Decrypting

- Plaintext is encrypted ***two letters at a time (bigrams)***:
- **Encryption:** Each ***2-letter plaintext combination*** forms the opposite diagonal ***corners*** of a rectangle in the Playfair matrix. The corresponding cipher letters are found in the ***other*** two corners of that rectangle
  - *the first ciphertext letter is the one in the same row as the first plaintext letter*
- **Decryption:** performed in the same manner, except using the ciphertext as input

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

Example: plaintext "wh" is encrypted as "vy" (see next slide for details)

Playfair Encryption/Decryption Rules

1. if one plaintext character left over, pad with a 'Z' , except if the singleton is a 'Z' then replace it with an 'X'

2. if a bigram consists of two of the same letter, insert an 'X' after the first letter to make a bigram, and then move the second letter to be the first letter of the next bigram

3. if both letters fall in the same row, replace each with the letter to its right (wrapping back to start from end)

4. if both letters fall in the same column, replace each with the letter below it (wrapping from bottom to top)

5. otherwise each letter is replaced by the letter in the same row and in the column of the other letter of the pair (opposite diagonal corners of rectangle)

6. **(VERY IMPORTANT)** the first cipher letter is the corner that is in the same row as the first plaintext letter of the bigram; the second cipher letter is the opposite diagonal corner from the first cipher letter (**also: when choosing a ciphertext letter from the "I/J" matrix element, always choose "I"**).

Playfair Examples

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

Wireless: E(wi re le sx sz) =XG MK UL XA TX ← *note double "s" and singleton*

Monday: E(mo nd ay) = ON RY NB ← *note "m" and "o" are in same row*

Deem: E(de em) = CK LC ← no double-"e"; also, "e" and "m" in same column

NOTE: For the Playfair cipher, the "key" is the keyword + rules 1-6

Hill Cipher

- Invented in 1929 by Lester S. Hill
- Uses **linear algebra (matrix multiplication)**
- Letters are encoded as numbers modulo 26
- Blocks of letters are interpreted as vectors
- Key is a random matrix of same dimension as vectors
- Key must have an inverse modulo 26 (to permit decryption)
- Although elegant, still relatively easy to break
- But it introduces techniques used in stronger ciphers like AES

Vector Dot Product

- A ***vector*** is an ordered set of numbers
  - Ex. [3, 27, 16]
  - a vector can be represented horizontally (as above) or vertically, whichever is more convenient

- The ***dot product*** of two vectors is simply the sum of the products of each component
  - Ex. [1, 5, 4] · [4, 2, 6] = (1)(4) + (5)(2) + (4)(6) = 4 + 10 + 24 = 38
  - The input vectors must have the same length
  - The dot product is just a simple *number* (scalar)

Matrices

- A **matrix** is a rectangular arrangement of numerical values
  - similar to a vector, a matrix is just an ordered arrangement
    - in fact, a vector is a matrix at least one of whose dimensions is 1
    - we can consider each row or column as a vector
  - can be square
  - Note: no mathematical operations are included or implied by the arrangement
    - i.e., the entries in each row are not added, multipled, etc.

- We enclose matrices with square brackets

- Examples: $[7]$ $\begin{bmatrix} 23 & 34 & 16 \\ 11 & 53 & 4 \\ 22 & 123 & 463 \end{bmatrix}$ $\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} \end{bmatrix}$

Matrix Multiplication Modulo 26

For the Hill Cipher, we do matrix arithmetic modulo 26

To multiply a matrix by a vector, the matrix must be have as many columns as the vector has rows.

Each element of the product vector is the dot product of the corresponding row of the matrix (treated as a simple vector) and the entire input vector, reduced modulo 26.

Example:

$$\begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix} \begin{bmatrix} v_{11} \\ v_{21} \end{bmatrix} = \begin{bmatrix} m_{11} \bullet v_{11} + m_{12} \bullet v_{21} \\ m_{21} \bullet v_{11} + m_{22} \bullet v_{21} \end{bmatrix}$$

Notation: $\mathbf{m_{21}}$ denotes the matrix element in the **second** row, **first** column

Hill Cipher Matrix Algebra

- The Hill cipher uses a square matrix as its key.
  - We call this matrix the "key matrix" for the Hill cipher
  - It must have a matrix inverse so we can decrypt what we encrypt

- Basic idea: for the **key matrix K** has an inverse K^{-1}, then for <u>any</u> vector v:

$$\boxed{v = K^{-1} \bullet K \bullet v}$$

- So, to encrypt: $C = K \bullet M$

- And to decrypt: $M = K^{-1} \bullet C \quad (= K^{-1} \bullet K \bullet M)$

- Where M = plaintext, C = ciphertext, and K = Hill cipher key matrix

Padding: Use the letter "x" to pad the last block, if needed

Hill Cipher 2x2 Encryption Example

Let M = “next” = [13, 4, 23, 19]

And let $K = \begin{bmatrix} 5 & 8 \\ 17 & 3 \end{bmatrix}$

Since the matrix is 2x2, we split the message into blocks (vectors) of length 2 and encrypt each pair separately.

For example, the first 2 plaintext characters (“ne”) are encrypted as follows:

NOTE: we do the arithmetic modulo 26

$$C = \begin{bmatrix} 5 & 8 \\ 17 & 3 \end{bmatrix} \begin{bmatrix} 13 \\ 4 \end{bmatrix} = \begin{bmatrix} (5 \bullet 13 + 8 \bullet 4) \bmod 26 \\ (17 \bullet 13 + 3 \bullet 4) \bmod 26 \end{bmatrix} = \begin{bmatrix} 19 \\ 25 \end{bmatrix} = \text{TZ}$$

Hill Cipher 3x3 Encryption Example

Let M = “next” = [13, 4, 23, 19]

And let

$$K = \begin{bmatrix} 5 & 8 & 2 \\ 17 & 3 & 6 \\ 4 & 12 & 7 \end{bmatrix}$$

•Since the matrix is 3x3, this time we split the message into blocks of length 3 and encrypt each block separately.

•And since the second block is just the letter “**t**” we must pad it to make a full block; so, let’s just use “**x**” as the pad character. This gives us 2 blocks: “**nex**” and “**txx**”

•For example, the first block (“**nex**”) is encrypted as follows (**again using mod 26 arithmetic**):

$$C = \begin{bmatrix} 5 & 8 & 2 \\ 17 & 3 & 6 \\ 4 & 12 & 7 \end{bmatrix} \begin{bmatrix} 13 \\ 4 \\ 23 \end{bmatrix} = \begin{bmatrix} 5\bullet 13 + 8\bullet 4 + 2\bullet 23 \\ 17\bullet 13 + 3\bullet 4 + 6\bullet 23 \\ 4\bullet 13 + 12\bullet 4 + 7\bullet 23 \end{bmatrix} = \begin{bmatrix} 143 \bmod 26 = 13 \\ 371 \bmod 26 = 7 \\ 261 \bmod 26 = 1 \end{bmatrix} = \text{NHB}$$

Transposition Ciphers

- Mapping is performed by a **permutation** (*rearrangement or shuffling*) of the plaintext letters
- Many permutations are possible
- Every permutation has an inverse, so we can easily encrypt and decrypt
- Permutations work just as well on letters, numbers, and bits
- Transposition ciphers we will cover:
  - Rail fence
  - Columnar transposition

Rail Fence of Depth 2

- Write plaintext message in 2 rows, alternating letters from one row to the next
- Encryption performed by reading across each row
- Decrypt by laying out the ciphertext in 2 rows and reading back as if were encrypting

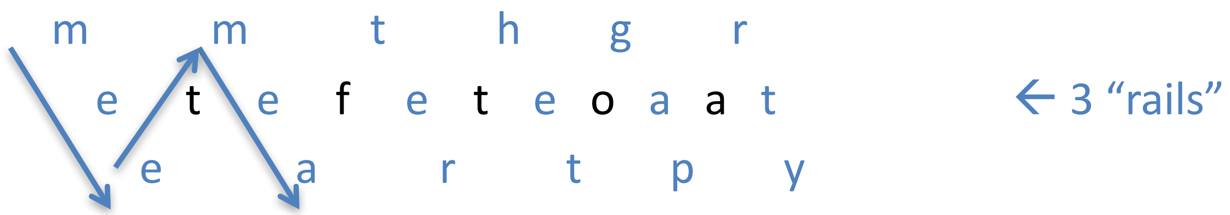
- Example: meet me after the toga party

```
m   e   m   a   t   r   h   t   g   p   r   y
  e   t   e   f   e   t   e   o   a   a   t        ← 2 "rails"
```

cipher: MEMATRHTGPRYETEFETEOAAT

Rail Fence of Depth 3

- Use 3 rows and weave back and forth from row 1 to row 2 to row 3, then back to row 2, then 1, and repeat
- Example: meet me after the toga party



cipher: MMTHGRETEFETEOAATEARTPY

Columnar Transposition Cipher

- Also known as "***transposition cipher with key***"
- Plaintext message is first written out in **rows** of a **fixed length**
  - the length of the rows is the number of columns used by the cipher
- **Pad** with **the letter 'x'** (random characters in general, but we will use 'x'), as needed to complete the last row
- Use a **numeric key** to encrypt by reading the characters down each column in the order specified by the key
  - Key is a **vector** listing the order in which to read the columns
  - The numbers in the vector are the column indices
  - Key length is same as row length

NOTE: Can be made more secure by re-encrypting with the same key (2 passes of encryption)

Columnar Transposition Cipher Example

Basic idea:

The key values above the columns give the order in which rows are transposed

Plaintext: “Attack postponed until two am”

Key:	**4**	**3**	**1**	**2**	**5**	**6**	**7**
Text:							
	a	t	t	a	c	k	p
	o	s	t	p	o	n	e
	d	u	n	t	i	l	t
	w	o	a	m	x	x	x

Here, we first read down the third column (below the “1”), then the fourth column (column “2”), and so on

→ Ciphertext: **TTNA | APTM | TSUO | AODW | COIX | KNLX | PETX**