

# L2: Basic Cryptography



Hui Chen, Ph.D.  
Dept. of Engineering & Computer Science  
Virginia State University  
Petersburg, VA 23806

# Acknowledgement

---

- ❑ Many slides are from or are revised from the slides of the author of the textbook
  - Matt Bishop, Introduction to Computer Security, Addison-Wesley Professional, October, 2004, ISBN-13: 978-0-321-24774-5. [Introduction to Computer Security @ VSU's Safari Book Online subscription](#)
  - <http://nob.cs.ucdavis.edu/book/book-intro/slides/>

# Overview

---

- ❑ Cryptography as mechanism to enforce security policies
- ❑ Concepts
  - Cryptography, cryptanalysis
- ❑ Basic Cryptography
  - Classical Cryptography
  - Public Key Cryptography
  - Cryptographic Checksums

# Overview

---

## ❑ Classical Cryptography

- Caesar cipher
- Vigenere cipher
- DES
- AES

## ❑ Public Key Cryptography

- Diffie-Hellman
- RSA

## ❑ Cryptographic Checksums

- HMAC

# Security Policy and Mechanism

---

## ❑ Security policy

- A statement of what is allowed and what is not allowed
- Example
  - ❑ A student may not copy another student's homework
- Can be informal or highly mathematical

## ❑ Security mechanism

- A method, tool, or procedure for enforcing security policy
- Technical and non-technical
  - ❑ A homework electronic submission system (e.g., Blackboard) enforces who may read a homework submission

# Security Mechanisms

---

- ❑ Cryptographic mechanisms
- ❑ Non-cryptographic mechanisms (system-dependent mechanisms)

# Cryptography

---

- ❑ Word Origin
  - Greek words
  - “secrete writing”
- ❑ Art & science of concealing meaning

# Cryptanalysis

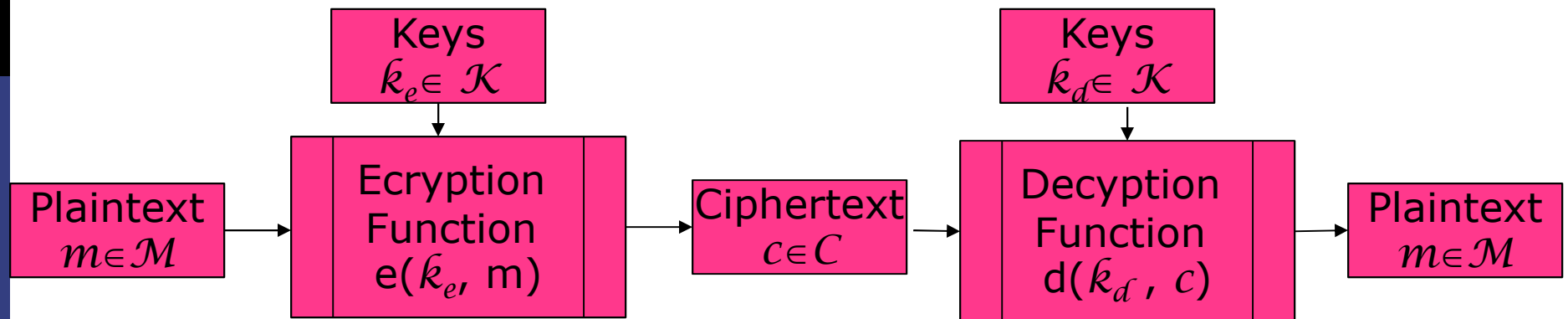
---

- ❑ Breaking of codes
- ❑ Application
  - World War II
- ❑ Further Reading
  - W. Diffie and M. Hellman. 2006. New directions in cryptography. *IEEE Trans. Inf. Theor.* 22, 6 (September 2006), 644-654. DOI=10.1109/TIT.1976.1055638  
<http://dx.doi.org/10.1109/TIT.1976.1055638>

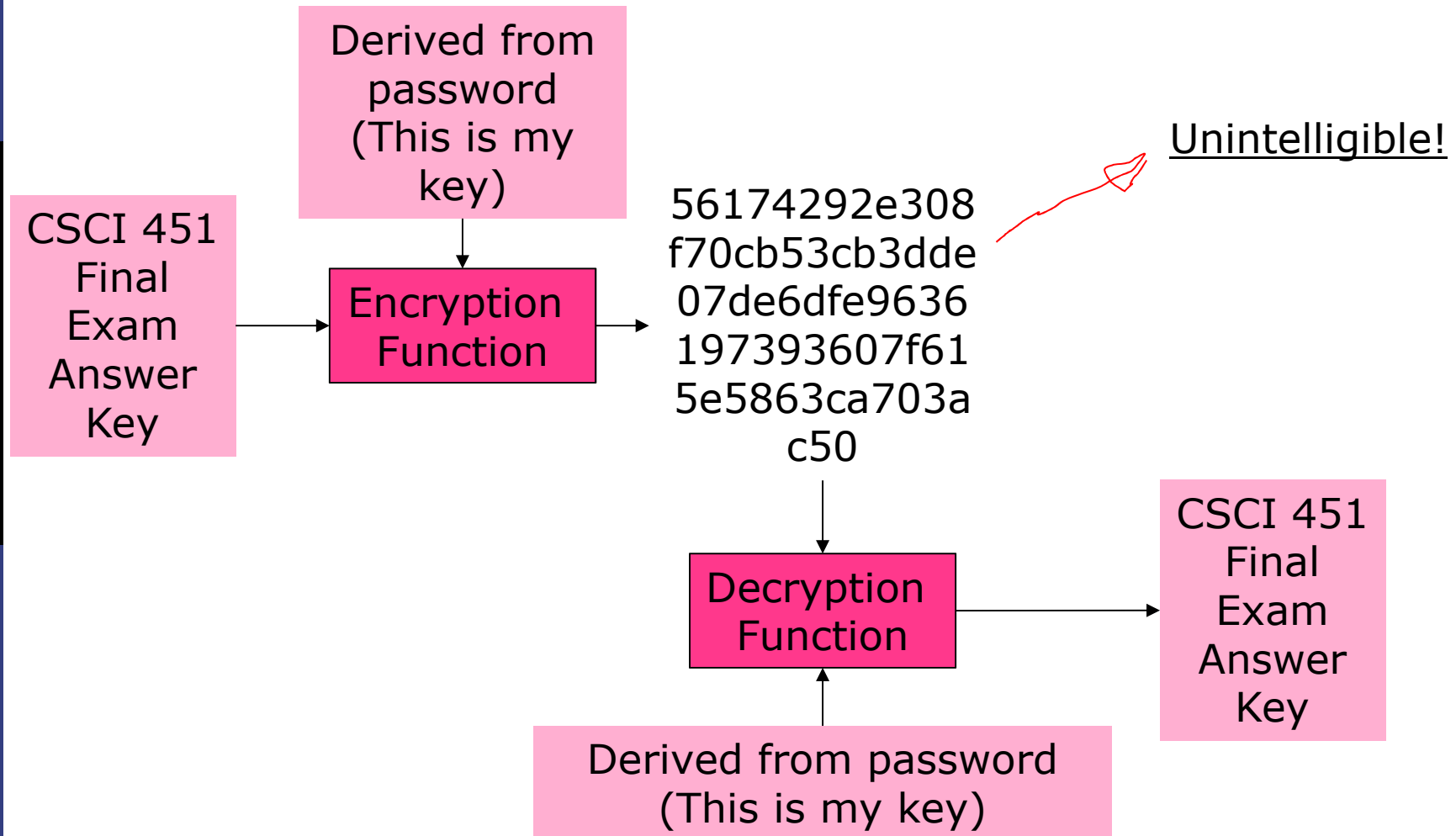


# Cryptosystem

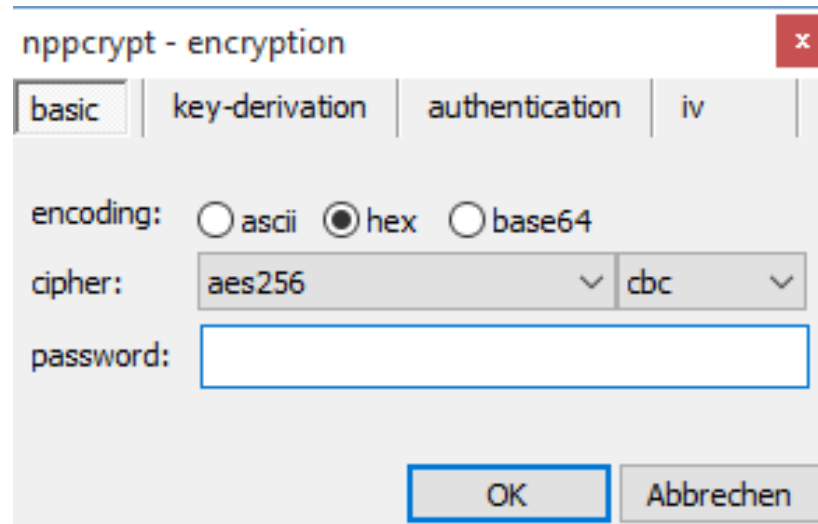
- Quintuple or 5-tuple  $(\mathcal{E}, \mathcal{D}, \mathcal{M}, \mathcal{K}, \mathcal{C})$ 
  - $\mathcal{M}$  set of plaintexts
  - $\mathcal{K}$  set of keys
  - $\mathcal{C}$  set of ciphertexts
  - $\mathcal{E}$  set of encryption functions  $e: \mathcal{M} \times \mathcal{K} \rightarrow \mathcal{C}$
  - $\mathcal{D}$  set of decryption functions  $d: \mathcal{C} \times \mathcal{K} \rightarrow \mathcal{M}$



# Example



# Example: NotePad++ NPPCrypt Plugin



```
<nppcrypt version="101">
<encryption cipher="aes256" mode="cbc" encoding="base16" />
<random iv="atuPGKigDnTy46fHBPM1vA==" salt="wOiEp1afVtXebE4kMSliFg==" />
<key algorithm="pbkdf2" hash="md5" iterations="1000" />
</nppcrypt>
56174292e308f70cb53cb3dde07de6dfe9636197393607f615e5863ca703ac50
```

# Classical Cryptography

---

- ❑ Sender, receiver share common key
  - Keys may be the same, or trivial to derive from one another
  - Sometimes called *symmetric cryptography*
- ❑ Two basic types
  - Transposition ciphers
    - ❑ Example: Rail Fence Cipher
  - Substitution ciphers
    - ❑ Example: Caesar Cipher
  - Combinations are called *product ciphers*

# Transposition Cipher

---

- ❑ Rearrange letters in plaintext to produce ciphertext
- ❑ Example
  - Rail-Fence Cipher
  - Example
    - ❑ HELLO WORLD becomes HLOOL ELWRD

# Rail-Fence Cipher

---

## □ Encryption

- Writing the plaintext in two rows, proceeding down, then across
- Reading the ciphertext across, then down.

# Rail-Fence Cipher

---

❑ Plaintext is HELLO WORLD

■ Rearrange as

HLOOL

ELWRD

■ Cipher-text is HLOOL ELWRD

❑ Mathematically, the key to a transposition cipher is a permutation function.

# Attacking Transposition Cipher

---

- ❑ Mathematically, the key to a transposition cipher is a permutation function.
- ❑ Observation: the permutation does not alter the frequency of plaintext characters
- ❑ Detecting the cipher by comparing character frequencies with a model of the language
  - Anagramming



# Anagramming Attack

---

- ❑ Language Model: tables of  $n$ -gram frequencies Input: Cipher-text
- ❑ Method:
  - If 1-gram frequencies match English frequencies, but other  $n$ -gram frequencies do not, probably transposition
  - Let  $n := 1$
  - Do
    - ❑  $n := n + 1$
    - ❑ Rearrange letters to form  $n$ -grams with highest frequencies
  - Until the transposition pattern is found

# Example

---

- ❑ Konheim's diagram table
- ❑ Cipher-text: HLOOLELWRD
- ❑ Frequencies of 2-grams beginning with H
  - HE 0.0305
  - HO 0.0043
  - HL, HW, HR, HD  $< 0.0010$
- ❑ Frequencies of 2-grams ending in H
  - WH 0.0026
  - EH, LH, OH, RH, DH  $\leq 0.0002$
- ❑ Implies E follows H

# Example

---

- Since “E” follows “H”, we arrange the letters so that each letter in the first block of five letters is adjacent to the corresponding letters in the 2<sup>nd</sup> block of five letters

- HLOOL ELWRD

- HE

- LL

- OW

- OR

- LD

# Substitution Ciphers

---

- ❑ Change characters in plaintext to produce ciphertext
- ❑ Example
  - Caesar cipher
    - ❑ Plaintext is HELLO WORLD
    - ❑ Change each letter to the third letter following it (X goes to A, Y to B, Z to C)
      - Key is 3, usually written as letter 'D'
    - ❑ Ciphertext is KHOOR ZRUOG
- ❑ More details follow

# Caesar Cipher

- Gaius Julius Caesar  
(July 100 BC - 15  
March 44 BC)
- *“If he had anything  
confidential to say, he  
wrote it in cipher...”*



# Did he invent this also?

---



# Caesar Cipher

---

- $\mathcal{M} = \{ \text{sequences of letters} \}$ 
  - The alphabet has  $N$  letters
- $\mathcal{K} = \{ i \mid i \text{ is an integer and } 0 \leq i \leq N - 1 \}$
- $\mathcal{E} = \{ E_k \mid k \in \mathcal{K} \text{ and for all letters } m, E_k(m) = (m + k) \bmod N \}$
- $\mathcal{D} = \{ D_k \mid k \in \mathcal{K} \text{ and for all letters } c, D_k(c) = (N + c - k) \bmod N \}$
- $C = \mathcal{M}$

# A Caesar Cipher

---

- $\mathcal{M} = \{0, 1, 2, \dots, 25\}$ 
  - Assume English alphabet. The alphabet has  $N = 26$  letters, representing each letter by its position in the alphabet
- Choose  $k = 3$
- $E_3(m) = (m + k) \bmod 26$
- $D_3(c) = (26 + c - k) \bmod 26$
- $C = \mathcal{M}$



# Example: Encryption

---

□ Plaintext = “HELLO”, i.e.,

■ 7 4 11 11 14

□  $k = 3$

□ Compute ciphertext

■  $7 + 3 \bmod 26 = 10$

■  $4 + 3 \bmod 26 = 7$

■  $11 + 3 \bmod 26 = 14$

■  $11 + 3 \bmod 26 = 14$

■  $14 + 3 \bmod 26 = 17$

■ 10 7 14 14 17

# Example

---

- ❑ Convert the integers back to letters
  - 10 7 14 14 17
- ❑ Ciphertext = “KHOOR”

# Example: Decryption

---

□ Ciphertext = “KHOOR”, i.e.,

■ 10 7 14 14 17

□  $k = 3$

□ Compute plaintext

■  $26 + 10 - 3 \bmod 26 = 7$

■  $26 + 7 - 3 \bmod 26 = 4$

■  $26 + 14 - 3 \bmod 26 = 11$

■  $26 + 14 - 3 \bmod 26 = 11$

■  $26 + 17 - 3 \bmod 26 = 14$

■ 7 4 11 11 14

# Example

---

- ❑ Convert the integers back to letters
  - 7 4 11 11 14
- ❑ Ciphertext = “HELLO”

# Attacking the Cipher

---

## ❑ Exhaustive search

- If the key space is small enough, try all possible keys until you find the right one
- Caesar cipher has only 26 possible keys (assuming English alphabet)
  - ❑ Exhaustive search is feasible

## ❑ Statistical analysis

- Compare to 1-gram model of English

# Exercise L2-1

---

- Use Caesar Cipher with  $k = 9$ , and compute ciphertext for the message below,
  - TROJAN

## Exercise L2-2

---

- ❑ Assume Caesar Cipher, use exhaustive search to find the key for the ciphertext below
  - XUW
- ❑ To determine if your key is correct, read the plaintext using the key guessed to see if it is intelligible.

## Exercise L2-3

---

- ❑ Write a program that computes ciphertext letter from a plaintext letter using Caesar cipher with a given key  $k$
- ❑ Write a program that computes plaintext letter from a given ciphertext letter using Caesar cipher with a given key  $k$ .
- ❑ Submit in Blackboard by 10AM, Monday, August 22.



# Statistical Attack

---

## □ Example:

- Breaking the ciphertext “KHOOR ZRUOG”

## □ Procedure:

1. Compute  $f(c)$ , frequency of each letter in ciphertext
2. Obtain unigram model of English
3. Compute  $\varphi(i)$ ,  $0 \leq i \leq 25$ , the correlation of the frequency of each letter in the ciphertext with the character frequencies in English,  $\varphi(i)$ ,  $0 \leq i \leq 25$
4. Sort  $\varphi(i)$ , compute plaintext for  $i$  when  $\varphi(i)$  from the greatest to the smallest, until the correct key is found

# Step 1: Compute $f(c)$

- ❑ Breaking the ciphertext KHOOR ZRUOG
- ❑ *Step 1: compute  $f(c)$*  frequency of character  $c$  in the ciphertext

$c$	$f(c)$	$c$	$f(c)$	$c$	$f(c)$	$c$	$f(c)$
0	0	7	0.1	13	0	19	0
1	0	8	0	14	0.3	20	0.1
2	0	9	0	15	0	21	0
3	0	10	0.1	16	0	22	0
4	0	11	0	17	0.2	23	0
5	0	12	0	18	0	24	0
6	0.1					25	0

## Step 2: Obtain Unigram Model of English (Frequencies of Letters)

Letter	Frequency	Letter	Frequency	Letter	Frequency	Letter	Frequency
a	0.080	h	0.060	n	0.070	t	0.090
b	0.015	i	0.065	o	0.080	u	0.030
c	0.030	j	0.005	p	0.020	v	0.010
d	0.040	k	0.005	q	0.002	w	0.015
e	0.130	l	0.035	r	0.065	x	0.005
f	0.020	m	0.030	s	0.060	y	0.020
g	0.015					z	0.002

## Step 3: Compute Correlation $\phi(i)$ for Caesar Cipher

---

- ❑  $f(c)$ : frequency of character  $c$  in ciphertext
- ❑  $p(26 + c - i \bmod 26)$ : frequency of character  $26 + c - i \bmod 26$  in the unigram model given  $i$ 
  - Since the decryption function of Caesar Cipher is
  - $d_i(c) = 26 + c - i \bmod 26$
- ❑ Correlation of frequency of letters in ciphertext with corresponding letters in English becomes
  - $\phi(i) = \sum_{0 \leq c \leq 25} f(c)p(26 + c - i \bmod 26)$
  - $p(x)$  is frequency of character  $x$  in English
- ❑ See next slide for example

# Example: Correlation: $\phi(4)$

$c$	$f(c)$	$c$	$f(c)$	$c$	$f(c)$	$c$	$f(c)$
0	0	7	0.1	13	0	19	0
1	0	8	0	14	0.3	20	0.1
2	0	9	0	15	0	21	0
3	0	10	0.1	16	0	22	0
4	0	11	0	17	0.2	23	0
5	0	12	0	18	0	24	0
6	0.1					25	0

$$\begin{aligned}
 \square \quad \phi(4) &= \sum_{0 \leq c \leq 25} f(c)p(26 + c - 4 \bmod 26) \\
 &= f(6)p(26 + 6 - 4 \bmod 26) + f(7)p(26 + 7 - 4 \bmod 26) + \\
 &\quad f(10)p(26 + 10 - 4 \bmod 26) + f(14)p(26 + 14 - 4 \bmod 26) + \\
 &\quad f(17)p(26 + 17 - 4 \bmod 26) + f(20)p(26 + 20 - 4 \bmod 26) + \\
 &= 0.1 p(26 + 6 - 4 \bmod 26) + 0.1 p(26 + 7 - 4 \bmod 26) + \dots
 \end{aligned}$$

# Example: Correlation: $\phi(4)$ (continued)

Letter	Frequency	Letter	Frequency	Letter	Frequency	Letter	Frequency
a	0.080	h	0.060	n or 13	0.070	t	0.090
b	0.015	i	0.065	o	0.080	u	0.030
c or 2	0.030	j	0.005	p	0.020	v	0.010
d or 3	0.040	k or 10	0.005	q or 16	0.002	w	0.015
e	0.130	l	0.035	r	0.065	x	0.005
f	0.020	m	0.030	s	0.060	y	0.020
g or 6	0.015					z	0.002

$$\begin{aligned}
 \square \quad \phi(4) &= \sum_{0 \leq c \leq 25} f(c)p(26 + c - 4 \bmod 26) \\
 &= f(6)p(2) + f(7)p(3) + f(10)p(6) + f(14)p(10) + f(17)p(13) + f(20)p(16) \\
 &= 0.1*0.030 + 0.1*0.040 + 0.1*0.015 + 0.3*0.005 + 0.2*0.070 + 0.1*0.002 \\
 &= 0.0252
 \end{aligned}$$

## Correlation: $\varphi(i)$ for $0 \leq i \leq 25$

---

$i$	$\varphi(i)$	$i$	$\varphi(i)$	$i$	$\varphi(i)$	$i$	$\varphi(i)$
0	0.0482	7	0.0442	13	0.0520	19	0.0315
1	0.0364	8	0.0202	14	0.0535	20	0.0302
2	0.0410	9	0.0267	15	0.0226	21	0.0517
3	0.0575	10	0.0635	16	0.0322	22	0.0380
4	0.0252	11	0.0262	17	0.0392	23	0.0370
5	0.0190	12	0.0325	18	0.0299	24	0.0316
6	0.0660					25	0.0430

## Step 4: Test Results

---

### ❑ Most probable keys, based on $\varphi$ :

- $i = 6, \varphi(i) = 0.0660$ 
  - ❑ plaintext EBIIL TLOLA
- $i = 10, \varphi(i) = 0.0635$ 
  - ❑ plaintext AXEEH PHKEW
- $i = 3, \varphi(i) = 0.0575$ 
  - ❑ plaintext HELLO WORLD
- $i = 14, \varphi(i) = 0.0535$ 
  - ❑ plaintext WTAAD LDGAS

### ❑ Only English phrase is for $i = 3$

- That's the key (3 or 'D')



# Statistical Attack on Caesar Cipher

---

- $f(c)$ : frequency of character  $c$  in ciphertext
- $d(\tilde{k}_d, c)$ : decryption function on ciphertext character  $c$  with key  $\tilde{k}_d$
- $\varphi(\tilde{k}_d) = \sum_{0 \leq c \leq 25} f(c)p(d(\tilde{k}_d, c))$ : correlation of frequency of letters in ciphertext with corresponding letters in English
  - key is  $\tilde{k}_d$
  - $p(x)$  is frequency of character  $x$  in the language
- This correlation should be a maximum when the key  $k$  translates to the ciphertext into English, i.e.,
  - $\operatorname{argmax}_{\tilde{k}_d} \varphi(\tilde{k}_d)$

# Problem with Caesar Cipher

---

- ❑ Key is too short
  - Can be found by exhaustive search
  - Statistical frequencies not concealed well
    - ❑ They look too much like regular English letters
- ❑ So make it longer: long key may obscure the statistics
  - Multiple letters in key
  - Idea is to smooth the statistical frequencies to make cryptanalysis harder

# Vigenère Cipher

---

- ❑ Giovan Battista Bellaso, 1553
- ❑ Use phrase as the key
- ❑ Similar to Caesar cipher, but use each letter from the key to encipher
- ❑ Example

- Message: THE BOY HAS THE BALL

- Key: VIG

- Encipher using Caesar cipher for each letter:

key	VIGVIGVIGVIGVIGV
-----	------------------

plain	THEBOYHASTHEBALL
-------	------------------

cipher	OPKWWECIYOPKWIRG
--------	------------------

# Table-Lookup Approach

---

- ❑ Trade memory for efficiency
- ❑ Store pre-calculated ciphertext for each letter using each possible key letter
  - 26 letters
  - 26 possible keys
  - Table of  $26 \times 26$

# Vigenère Tableau

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

## Generate the table

In C++

```
for (int key = 0; key < KEY_SPACE_SIZE; key ++)  
{  
    cout << char(key + 'A') << ' ';  
    for (int letter = 0; letter < ALPHABET_SIZE; letter ++)  
    {  
        int ciphertext = (letter + key) % ALPHABET_SIZE;  
        cout << char(ciphertext + 'A') << ' ';  
    }  
    cout << endl;  
}
```

# Relevant Parts of Tableau

---

	<i>G</i>	<i>I</i>	<i>V</i>
<i>A</i>	G	I	V
<i>B</i>	H	J	W
<i>E</i>	L	M	Z
<i>H</i>	N	P	C
<i>L</i>	R	T	G
<i>O</i>	U	W	J
<i>S</i>	Y	A	N
<i>T</i>	Z	B	O
<i>Y</i>	E	H	T

▣ Tableau shown has relevant rows, columns only

▣ Example encipherments:

- key V, letter T: follow V column down to T row (giving "O")
- Key I, letter H: follow I column down to H row (giving "P")

# Useful Terms

---

- ❑ *period*: length of key
  - In earlier example, period is 3
- ❑ *tableau*: table used to encipher and decipher
  - Vigenère cipher has key letters on top, plaintext letters on the left
- ❑ *polyalphabetic*: the key has several different letters
  - Caesar cipher is monoalphabetic

# Attacking Vigenère Cipher

---

## □ Approach

- Establish period; call it  $n$
- Break message into  $n$  parts, each part being enciphered using the same key letter
- Solve each part
  - You can leverage one part from another

## □ We will show each step



# Target Ciphertext

---

- We want to break the Vigenère cipher using the ciphertext:

```
ADQYS MIUSB OXKKT MIBHK IZOOO
EQOOG IFBAG KAUMF VVTAA CIDTW
MOCIO EQOOG BMBFV ZGGWP CIEKQ
HSNEW VECNE DLAAY RWKXS VNSVP
HCEUT QOIOF MEGJS WTPCH AJMOC
HIUIX
```

# Establish Period

---

- ❑ The key is to establish the period
- ❑ Method
  - Using Kasiski method establish initial guesses
  - Using index of coincidence to confirm the guesses

# Establish Period: Kasiski

---

- ❑ Friedrich W. Kasiski: a Prussian cavalry officer
  - *repetitions in the ciphertext occur when characters of the key appear over the same characters in the plaintext (Kasiski, 1863)*

- ❑ Example:

key	<b>VIGV</b> IGVIG <b>VIGV</b> IGV
plain	<b>THEB</b> OYHAS <b>THEB</b> ALL
cipher	<u><b>OPKW</b></u> WECIY <u><b>OPKW</b></u> IRG

Counting distance 0123456789

Note the key and plaintext line up over the repetitions (underlined). As distance between repetitions is 9, the period is a factor of 9 (that is, 1, 3, or 9)

# Repetitions in Example

<i>Letters</i>	<i>Start</i>	<i>End</i>	<i>Distance</i>	<i>Factors</i>
MI	5	15	10	2, 5
OO	22	27	5	5
OEQOOG	24	54	30	2, 3, 5
FV	39	63	24	2, 2, 2, 3
AA	43	87	44	2, 2, 11
MOC	50	122	72	2, 2, 2, 3, 3
QO	56	105	49	7, 7
PC	69	117	48	2, 2, 2, 2, 3
NE	77	83	6	2, 3
SV	94	97	3	3
CH	118	124	6	2, 3

- 
- Note that the program counts from 1 and we count from 0 in previous example

# Looking For Repetition using Provided Program

- ❑ Note that the program counts from 1; however, we count from 0 in previous example

```
octave>
findcommonsubstrings('ADQYSMIUSBOXKKTMIBHKIZOOOEQOOGIFBAGKAUMFVVTAACIDTWMOCIOEQOOGBMBFVZGGWPCIEKQHSNEWVEC
NEDLA AVRWKXSVNSVP HCEUTQOIOFMEGJSWTPCHAJMOCHIUIX', 'v');
```

Start	End	Len	Gap	Letters
6	16	2	10	MI
7	127	2	120	IU
23	28	2	5	OO
23	58	2	35	OO
24	28	2	4	OO
24	58	2	34	OO
27	106	2	79	QO
25	55	6	30	OEQOOG
40	64	2	24	FV
44	88	2	44	AA
46	53	2	7	CI
46	71	2	25	CI
51	123	3	72	MOC
53	71	2	18	CI
54	108	2	54	IO
57	106	2	49	QO
70	118	2	48	PC
78	84	2	6	NE
95	98	2	3	SV
119	125	2	6	CH

```
octave>
8/17/2016
```

# Estimate of Period

---

- ❑ OEQOOG is probably not a coincidence
  - It is too long for that
  - Period may be 1, 2, 3, 5, 6, 10, 15, or 30
- ❑ Most others (7/10) have 2 in their factors
- ❑ Almost as many (6/10) have 3 in their factors
- ❑ Begin with period of  $2 \times 3 = 6$

# Checking on Period

---

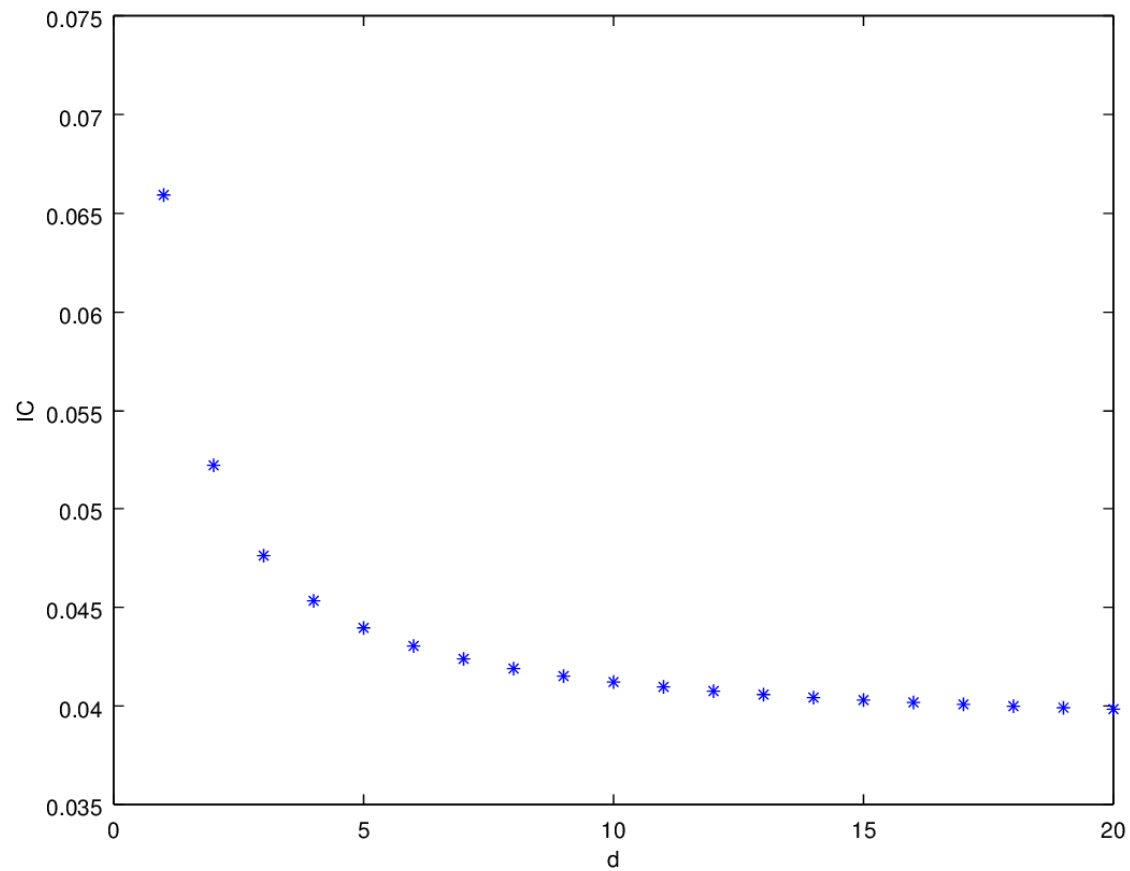
- ❑ Index of coincidence is probability that two randomly chosen letters from ciphertext match
- ❑ Tabulated for different periods for English ciphertexts at different periods (d):  
$$IC = 0.065933 / d + 0.038462 (d - 1) / d$$

Period	IC	Period	IC	Period	IC
1	0.066	3	0.047	5	0.044
2	0.052	4	0.045	10	0.041
Large	0.038				



# Index of Coincidence for English Ciphertext

---



# Computing IC

---

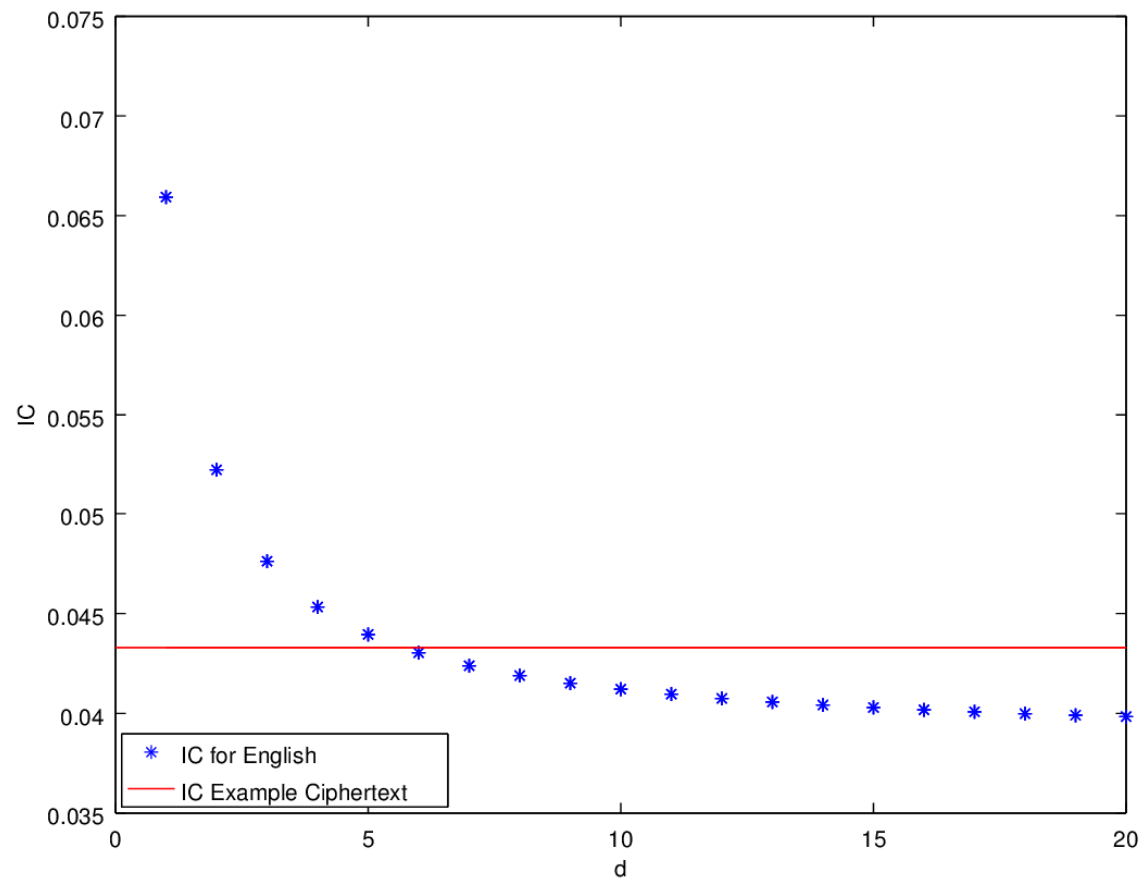
- $IC = [n (n - 1)]^{-1} \sum_{0 \leq i \leq 25} [F_i (F_i - 1)]$ 
  - where  $n$  is length of ciphertext and  $F_i$  the number of times character  $i$  occurs in ciphertext
- Here,  $IC = 0.043$ 
  - Indicates a key of slightly more than 5
  - A statistical measure, so it can be in error, but it agrees with the previous estimate (which was 6)

# Computing IC using Provided Program

---

```
octave> ciphertext =  
'ADQYSMIUSBOXKKTMIBHKIZOOOEQOOGIFBAGK  
AUMFVVTAACIDTWMOCIOEQOOGBMBFVZGGWPCIE  
KQHSNEWVECNEDLAAVRWKXSVNSVPHCEUTQOIOF  
MEGJSWTPCHAJMOCHIUIX';  
octave> computeic(ciphertext)  
ans = 0.043292  
octave>
```

# Confirming Key Length



# Splitting Into Alphabets using Estimated Period (Period = 6)

---

## Ciphertext

ADQYS MIUSB OXKKT MIBHK IZOOO EQOOG IFBAG  
KAUMF VVTAA CIDTW MOCIO EQOOG BMBFV ZGGWP  
CIEKQ HSNEW VECNE DLAAY RWKXS VNSVP HCEUT  
QOIOF MEGJS WTPCH AJMOC HIUIX

alphabet 1: AIKHOIATTOBGEEERNEOSAI

alphabet 2: DUKKEFUAWEMGKWDWSUFWJU

alphabet 3: QSTIQBMAMQBWQVLKVTMTMI

alphabet 4: YBMZOAFCOOFPHEAXPQEPOX

alphabet 5: SOIOOGVICOVCSVASHOGCC

alphabet 6: MXBOGKVDIGZINNVVCIJHH

# Checking on IC

---

alphabet 1: AIKHOIATTOBGEEERNEOSAI

alphabet 2: DUKKEFUAWEMGKWDWSUFWJU

alphabet 3: QSTIQBMAMQBWQVLKVTMTMI

alphabet 4: YBMZOAFCOOFPHEAXPQEPOX

alphabet 5: SOIOOGVICOVCSVASHOGCC

alphabet 6: MXBOGKVDIGZINNVVCIJHH

## □ ICs

- #1, 0.069; #2, 0.078; #3, 0.078; #4, 0.056; #5, 0.124; #6, 0.043
- Indicate all alphabets have period 1, except #4 and #6; assume statistics off

# Computing IC using Provided Octave/Matlab Program

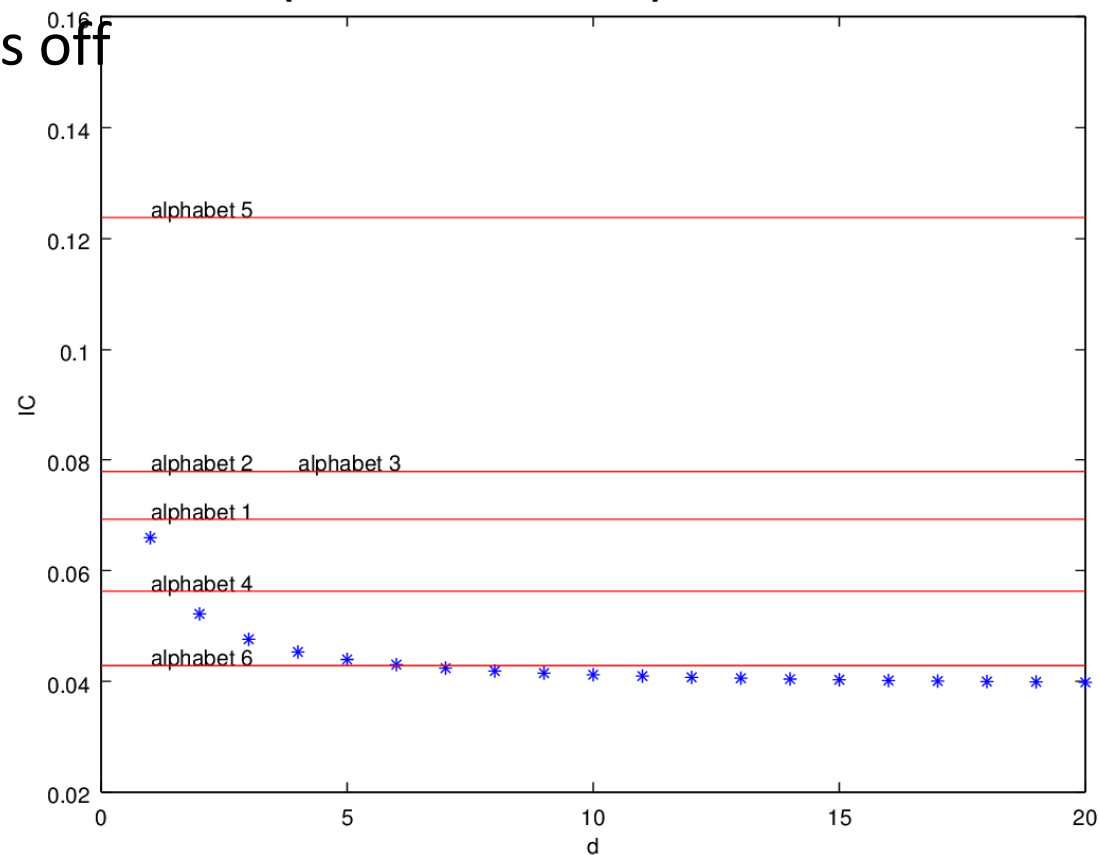
---

```
octave> alphabet1 = ciphertext(1:6:length(ciphertext))
alphabet1 = AIKHOIATTOBGEEERNEOSAI
octave> computeic(alphabet1)
ans = 0.069264
octave> alphabet2 = ciphertext(2:6:length(ciphertext))
alphabet2 = DUKKEFUAWEMGKWDWSUFWJU
octave> computeic(alphabet2)
ans = 0.077922
octave> alphabet3 = ciphertext(3:6:length(ciphertext))
alphabet3 = QSTIQBMAMQBWQVLKVTMTMI
octave> computeic(alphabet3)
ans = 0.077922
octave>
```

.....

# Checking on IC

- all alphabets have period 1, except #4 and #6; assume statistics off





# Frequency Examination

---

	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
1	3	1	0	0	4	0	1	1	3	0	1	0	0	1	3	0	0	1	1	2	0	0	0	0	0	0
2	1	0	0	2	2	2	1	0	0	1	3	0	1	0	0	0	0	0	1	0	4	0	4	0	0	0
3	1	2	0	0	0	0	0	2	0	1	1	4	0	0	0	4	0	1	3	0	2	1	0	0	0	0
4	2	1	1	0	2	2	0	1	0	0	0	0	1	0	4	3	1	0	0	0	0	0	0	2	1	1
5	1	0	5	0	0	0	2	1	2	0	0	0	0	0	5	0	0	0	3	0	0	2	0	0	0	0
6	0	1	1	1	0	0	2	2	3	1	1	0	1	2	1	0	0	0	0	0	0	3	0	1	0	1

**Letter frequencies are (H high, M medium, L low):**

HMMM HMMH HMMMM H HML HHH MLLLLL

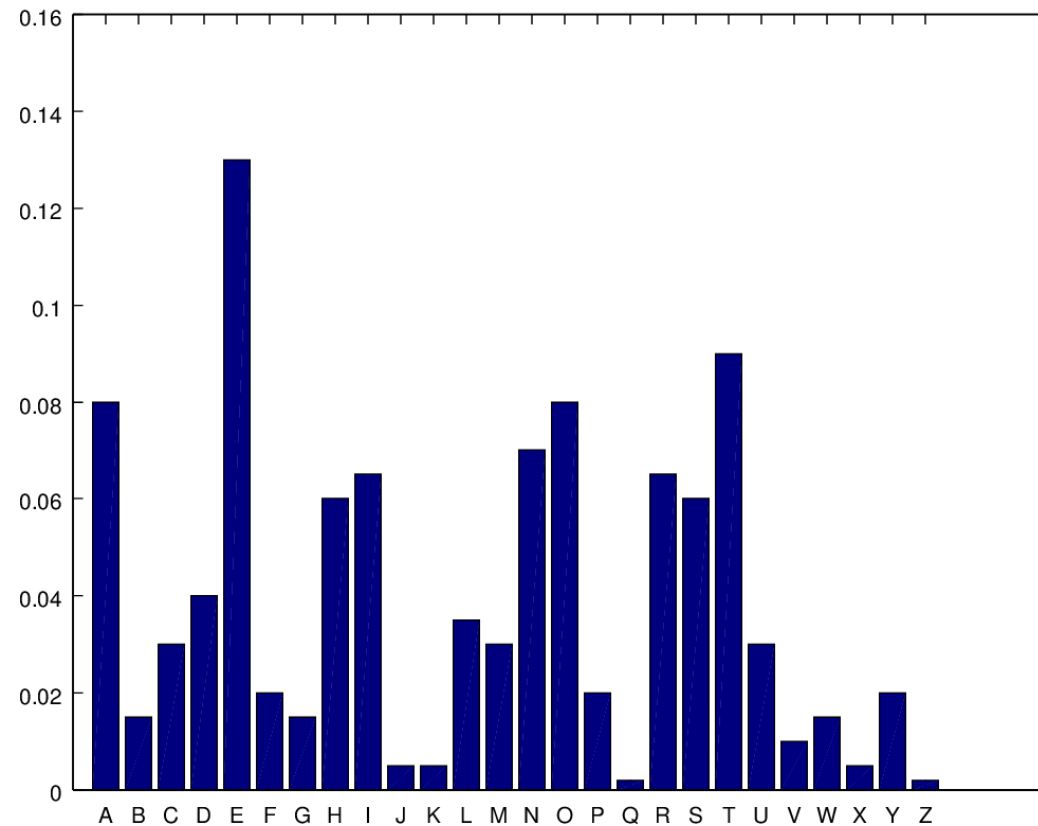
# English Letter Frequencies

---

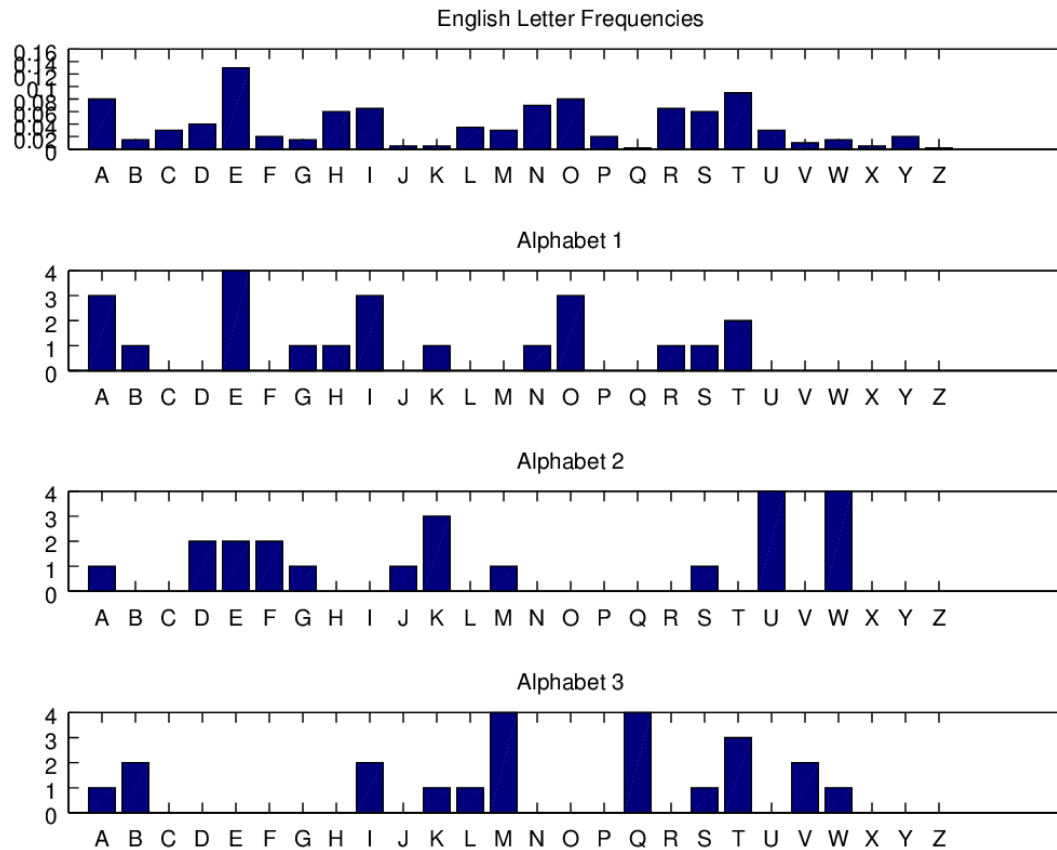
Letter	Frequency	Letter	Frequency	Letter	Frequency	Letter	Frequency
a	0.080	h	0.060	n	0.070	t	0.090
b	0.015	i	0.065	o	0.080	u	0.030
c	0.030	j	0.005	p	0.020	v	0.010
d	0.040	k	0.005	q	0.002	w	0.015
e	0.130	l	0.035	r	0.065	x	0.005
f	0.020	m	0.030	s	0.060	y	0.020
g	0.015					z	0.002

# English Letter Frequencies

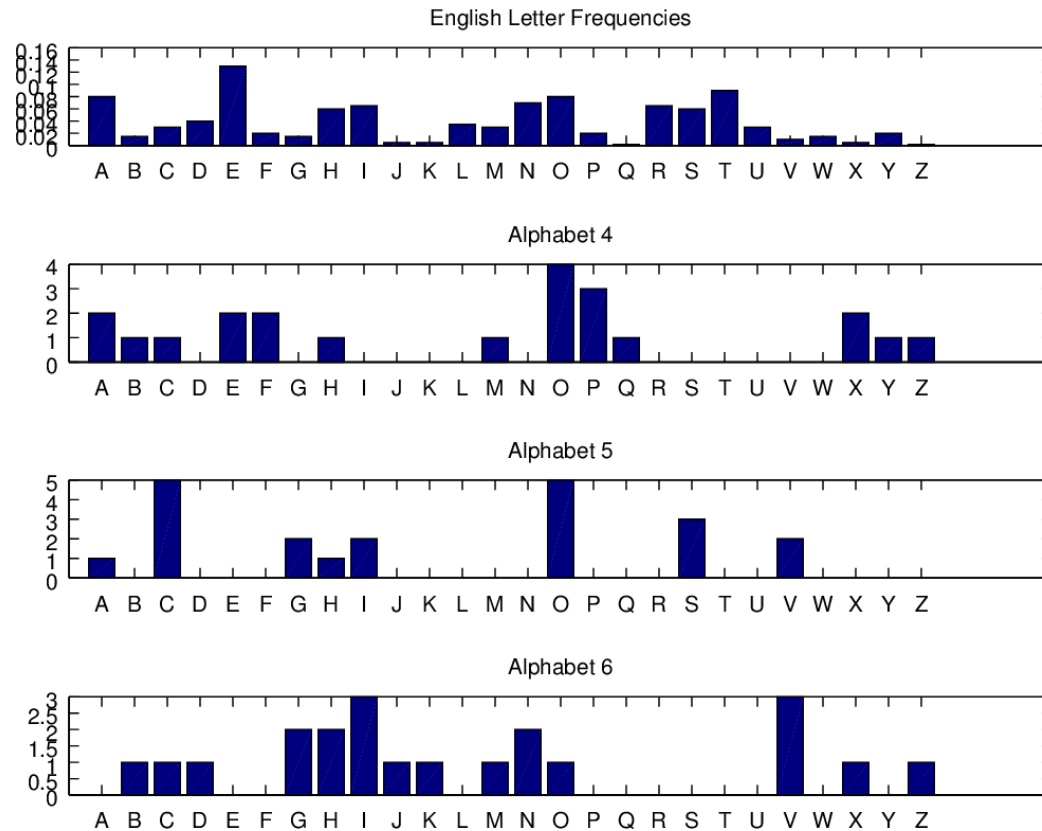
---



# Guessing Key



# Guessing Key



# Begin Decryption

---

- ❑ First matches characteristics of unshifted alphabet
- ❑ Third matches if I shifted to A
- ❑ Sixth matches if V shifted to A
- ❑ Substitute into ciphertext (bold are substitutions)

**A**DIYS **RIUK**B O**CKKL** MI**GHK** **AZOTO** EI**OO**L  
**I**F**T**AG **PAUE**F V**ATAS** CI**ITW** **EOCNO** EI**OO**L  
**BMT**FV **EGGO**P C**NEKI** HS**SEW** **NECSE** DD**AAA**  
**RWC**XS **ANSNP** H**HEUL** QO**NOF** **EEGOS** WL**PCM**  
**AJE**OC **MIUA**X

# Look For Clues

---

- **AJE** in last line suggests “are”, meaning second alphabet maps A into S:

**ALIYS RICKB OCKSL MIGHS AZOTO**  
**MI00L INTAG PACEF VATIS CIITE**  
**EOCNO MI00L BUTFV EGOOP CNESI**  
**HSSEE NECSE LDAAA RECXS ANANP**  
**HHECL QONON EEGOS ELPCM AREOC**  
**MICAX**

# Next Alphabet

---

- **MICAX** in last line suggests “mical” (a common ending for an adjective), meaning fourth alphabet maps O into A:

<b>ALIMS</b>	<b>RICKP</b>	<b>OCKSL</b>	<b>AIGHS</b>	<b>ANOTO</b>	<b>MICOL</b>
<b>INTOG</b>	<b>PACET</b>	<b>VATIS</b>	<b>QIITE</b>	<b>ECCNO</b>	<b>MICOL</b>
<b>BUTTV</b>	<b>EGOOD</b>	<b>CNESI</b>	<b>VSSEE</b>	<b>NSCSE</b>	<b>LDOAA</b>
<b>RECLS</b>	<b>ANAND</b>	<b>HHECL</b>	<b>EONON</b>	<b>ESGOS</b>	<b>ELDCM</b>
<b>ARECC</b>	<b>MICAL</b>				



# Got It!

---

- QI means that U maps into I, as Q is always followed by U:

ALIME	RICKP	ACKSL	AUGHS	ANATO	MICAL
INTOS	PACET	HATIS	QUITE	ECONO	MICAL
BUTTH	EGOOD	ONESI	VESEE	NSOSE	LDOMA
RECLE	ANAND	THECL	EANON	ESSOS	ELDOM
ARECO	MICAL				

# With Proper Spacing and Punctuation

---

- ❑ A LIMERICK PACKS LAUGHS ANATOMICAL INTO SPACE THAT IS QUITE ECONOMICAL. BUT THE GOOD ONES I'VE SEEN SO SELDOM ARE CLEAN, AND THE CLEAN ONES SO SELDOM ARE COMICAL.

# Lessons Learned

---

- ❑ Vigenère cipher was once considered unbreakable
- ❑ It is easy to break by hand!
- ❑ Principles of attacks hold for more complex ciphers
  - WordPerfect: encipher a file with a password
    - ❑ Certain fields in the enciphered file contained information internal to WordPerfect
    - ❑ These fields could be predicted
- ❑ Cycles of Attack → Fix → Attack → Fix
- ❑ Stronger ciphers

# One-Time Pad

---

- A variant of Vigenère Cipher
  - The key string is chosen at random
  - The key string is at least as long as the message

# Discussion on Attacks

---

- ❑ Opponent whose goal is to break cryptosystem is the *adversary*
  - Assume adversary knows algorithm used, but not key
- ❑ Three types of attacks:
  - *ciphertext only*: adversary has only ciphertext; goal is to find plaintext, possibly key
  - *known plaintext*: adversary has ciphertext, corresponding plaintext; goal is to find key
  - *chosen plaintext*: adversary may supply plaintexts and obtain corresponding ciphertext; goal is to find key
- ❑ Good cryptosystems protects against all 3 types of attacks

# Discussion on Attacks

---

## ❑ Mathematical attacks

- Based on analysis of underlying mathematics

## ❑ Statistical attacks

- Make assumptions about the distribution of letters, pairs of letters (digrams), triplets of letters (trigrams), *etc.*
  - ❑ Called *models of the language*
- Examine ciphertext, correlate properties with the assumptions.

## Exercise L2-4

---

- ❑ Textbook exercise: Question 2 of Chapter 8 in the textbook
- ❑ You may use the provided program `attackcaesar.m`, but must explain your result

# Exercise L2-5

---

## ❑ Breaking two Vigenère ciphers

- The ciphertext is in *pg.txt* and *tc.txt*
- Use the programs (the example that breaks *pg.txt* follows and you will break *tc.txt* on your own)

## ❑ Disclaimer

- All programs were tested in Octave, but not in Matlab although they should be mostly fine in Matlab



# Attacking Vigenère in Programs (1)

---

## 1. Read the ciphertext and find repeating substrings

```
octave> ciphertext = readline('pg.txt');
octave> computeletterfreq(ciphertext);
octave> [idx1st, idx2nd, lensubstr, gaps] =
findcommonsubstrings(ciphertext(1:1000), 'v');
octave> gaps(lensubstr > 6)
ans =
    216    48    78   138    60    12
```

## 2. Let us now guess the period (the key length): 6

## 3. Confirm with index of coincidence

```
octave> computeic(ciphertext)
ans = 0.041854
```

# Attacking Vigenère in Programs (2)

---

## 4. Now guess the letters in the key

```
octave> guesskey(ciphertext(1:6:end), 'v');  
octave> guesskey(ciphertext(2:6:end), 'v');  
octave> guesskey(ciphertext(1:6:end), 'v');  
octave> guesskey(ciphertext(2:6:end), 'v');  
octave> guesskey(ciphertext(3:6:end), 'v');  
octave> guesskey(ciphertext(4:6:end), 'v');  
octave> guesskey(ciphertext(5:6:end), 'v');  
octave> guesskey(ciphertext(6:6:end), 'v');
```

The key appears to be ASIMOV.

# Attacking Vigenère in Programs (3)

---

## 5. Decipher the ciphertext

```
octave:34> char(vigenere(ciphertext, 'ASIMOV', 'd'))  
ans =  
THEPROJECTGUTENBERGEBOOKOFMOBYDICKORTHEWHALEBYHERMANM.....
```

What if the result is not intelligible?

# Homework L2-1

---

- ❑ Breaking a Vigenère cipher. The ciphertext is in Exercise 8 of Chapter 8 in the textbook.
- ❑ Show steps, intermediate and final results

# Summary

---

- ❑ Classical Cryptography
  - Caesar cipher
  - Vigenere cipher
- ❑ Attack on Caesar cipher and Vigenere cipher
- ❑ Concepts of cryptanalysis
  - Simple cryptanalysis