CIS 551 / TCOM 401

Computer and Network Security

Spring 2009

Lecture 12

Announcements

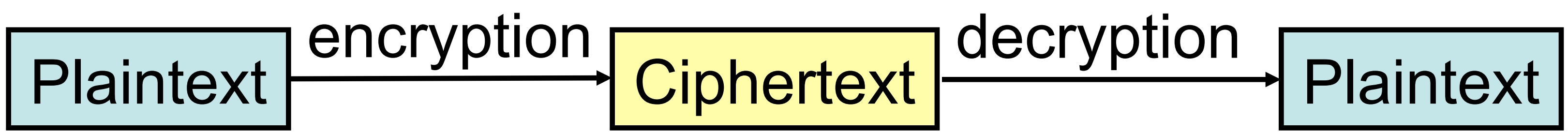
- Plan for Today:
  – Introduction to Cryptography
- Project 2 reminder
  – Due: Friday, 11:59 pm
- Project 3 will be up soon

Κρυϟτογραφία (Cryptography)

- From the Greek "kryptos" and "graphia" for “secret writing”

- Confidentiality
  - Obscure a message from eaves-droppers
- Integrity
  - Assure recipient that the message was not altered
- Authentication
  - Verify the identity of the source of a message
- Non-repudiation
  - Convince a 3rd party that what was said is accurate

Terminology



- Cryptographer
  - Invents cryptosystems
- Cryptanalyst
  - Breaks cryptosystems
- Cryptology
  - Study of crypto systems
- Cipher
  - Mechanical way of encrypting text or data
- Code
  - Semantic translation: “eat breakfast tomorrow” = “attack on Thursday” (or use Navajo!)
- Key
  - – a parameter of the cipher algorithm

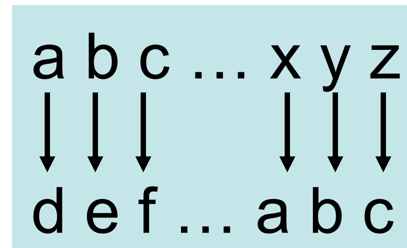
Kinds of Cryptographic Analysis

- Goal is to recover the key (& algorithm)
  - And hence recover the plaintext

- *Ciphertext Only* attacks
  - No information about content or algorithm
  - Very hard
- *Algorithm & Ciphertext* attacks
  - Known algorithm, known ciphertext, recover key
  - Common in practice
- *Known Plaintext* attacks
  - Full or partial plaintext available in addition to ciphertext
- *Chosen Plaintext* attacks
  - Attacker can choose which plaintext is encrypted, tries to reverse engineer the key. May be able to choose multiple plaintexts.

The Caesar Cipher

- Purportedly used by Julius Caesar (c. 75 B.C.)
  - Add 3 mod 26

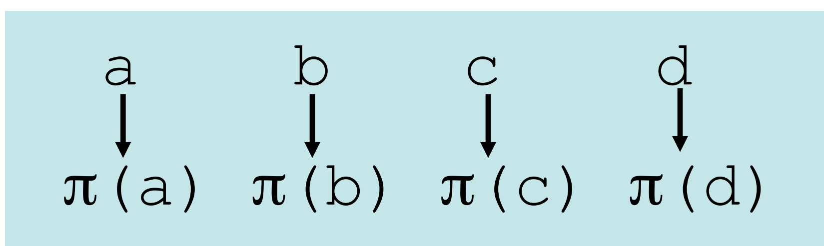


- Advantages
  - Simple
  - Intended to be performed in the field
  - Most people couldn’t read anyway
- Disadvantages
  - Violates “no security through obscurity”
  - Easy to break (why?)

Monoalphabetic Ciphers

- Also called *substitution* ciphers
- Separate *algorithm* from the *key*
  - Add N mod 26
  - rot13 = Add 13 mod 26

- General monoalphabetic cipher
  - Arbitrary permutation π of the alphabet
  - Key is the permutation



Example Cipher

	a	**b**	**c**	**d**	**e**	**f**	**g**	**h**	**i**	**j**	**k**	**l**	**...**
π	**z**	**d**	**a**	**n**	**c**	**e**	**w**	**i**	**b**	**f**	**g**	**h**	**...**

Plaintext: **he lied**
Ciphertext: **ic hbcn**

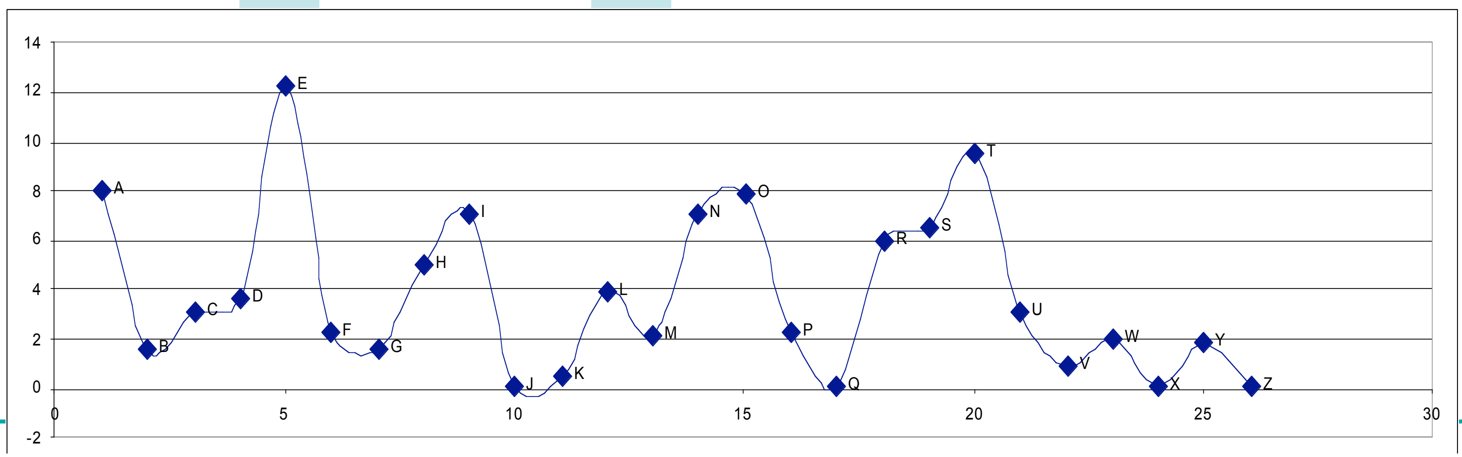
Cryptanalysis of Monoalphabetic Ciphers

- Brute force attack: try every key
  - N! Possible keys for N-letter alphabet
  - 26! $\approx 4 \times 10^{26}$ possible keys
  - Try 1 key per μsec … 10 trillion years
- …but (!) monoalphabetic ciphers are *easy* to solve

- One-to-one mapping of letters is bad
- Frequency distributions of common letters

Order & Frequency of Single Letters

E	12.31%	L	4.03%	B	1.62%
T	9.59	D	3.65	G	1.61
A	8.05	C	3.20	V	0.93
O	7.94	U	3.10	K	0.52
N	7.19	P	2.29	Q	0.20
I	7.18	F	2.28	X	0.20
S	6.59	M	2.25	J	0.10
R	6.03	W	2.03	Z	0.09
H	5.14	Y	1.88		



Monoalphabetic Cryptanalysis

- Count the occurrences of each letter in the cipher text
- Match against the statistics of English

- Most frequent letter likely to be “e”
- 2^{nd} most frequent likely to be “t”
- etc.

- Longer ciphertext makes statistical analysis more likely to work…

Digrams and Trigrams

- Diagrams in frequency order (for English)

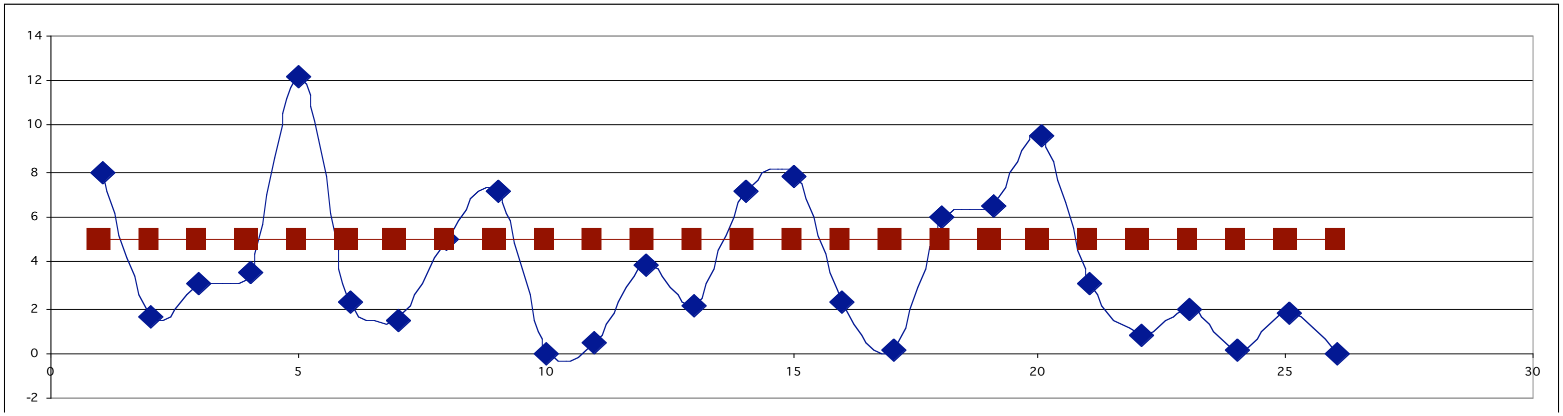
  TH HE AN IN ER RE ES ON EA TI AT ST EN
  ND OR

- Trigrams in frequency order (for English)

  THE AND THA ENT ION TIO FOR NDE
  HAS NCE EDT TIS OFT STH MEN

Desired Statistics

- Problems with monoalphabetic ciphers
  - Frequency of letters in ciphertext reflects frequency of plaintext
- Want a single plaintext letter to map to multiple ciphertext letters
  - “e” ⟶ “x”, “c”, “w”
- Ideally, ciphertext frequencies should be flat



Vigenère Tableau

- Multiple substitutions
  - Can choose “complimentary” ciphers so that the frequency distribution flattens out
  - More generally: more substitutions means flatter distribution
- Vigenère Tableau
  - Invented by Blaise de Vigenère for the court of Henry III of France (c. 1500’s)
  - Collection of 26 permutations
  - Usually thought of as a 26 x 26 grid
  - Key is a word

Vigenère Tableau

	a	b	c	d	e	f	g	.	.	.
A	a	b	c	d	e	f	g	.	.	.
B	b	c	d	e	f	g	h	.	.	.
C	c	d	e	f	g	h	i	.	.	.
D	d	e	f	g	h	i	j	.	.	.
E	e	f	g	h	i	j	k	.	.	.
.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.

```
Plaintext:     a bad deed
Key "bed":     B EDB EDBE
Ciphertext:    b fde hgfh
```

Polyalphabetic Substitutions

- Pick k substitution ciphers
  - $\pi_1\ \pi_2\ \pi_3\ \ldots\ \pi_k$
  - Encrypt the message by rotating through the k substitutions

m	e	s	s	a	g	e
$\pi_1(\mathbf{m})$	$\pi_2(\mathbf{e})$	$\pi_3(\mathbf{s})$	$\pi_4(\mathbf{s})$	$\pi_1(\mathbf{a})$	$\pi_2(\mathbf{g})$	$\pi_3(\mathbf{e})$
q	a	x	o	a	u	v

- Same letter can be mapped to multiple different ciphertexts
  - Helps smooth out the frequency distributions
  - *Diffusion*

Cracking Polyalphabetic Substitutions

- Step 1:
  - Try to identify the number of substitutions used
  - For example, guess the length of the word used as a key in the Vigenère tableau.

- Step 2:
  - Use frequency information to crack each of the subsitutions as though it was a monoalphabetic cipher.

Kasiski Method

- Identify key length of polyalphabetic ciphers
  - If pattern appears k times and key length is n then it will be encoded k/n times by the same key

- 1. Identify repeated patterns of ≥ 3 chars.
- 2. For each pattern
  - Compute the differences between starting points of successive instances
  - Determine the factors of those differences
- 3. Key length is likely to be one of the frequently occurring factors

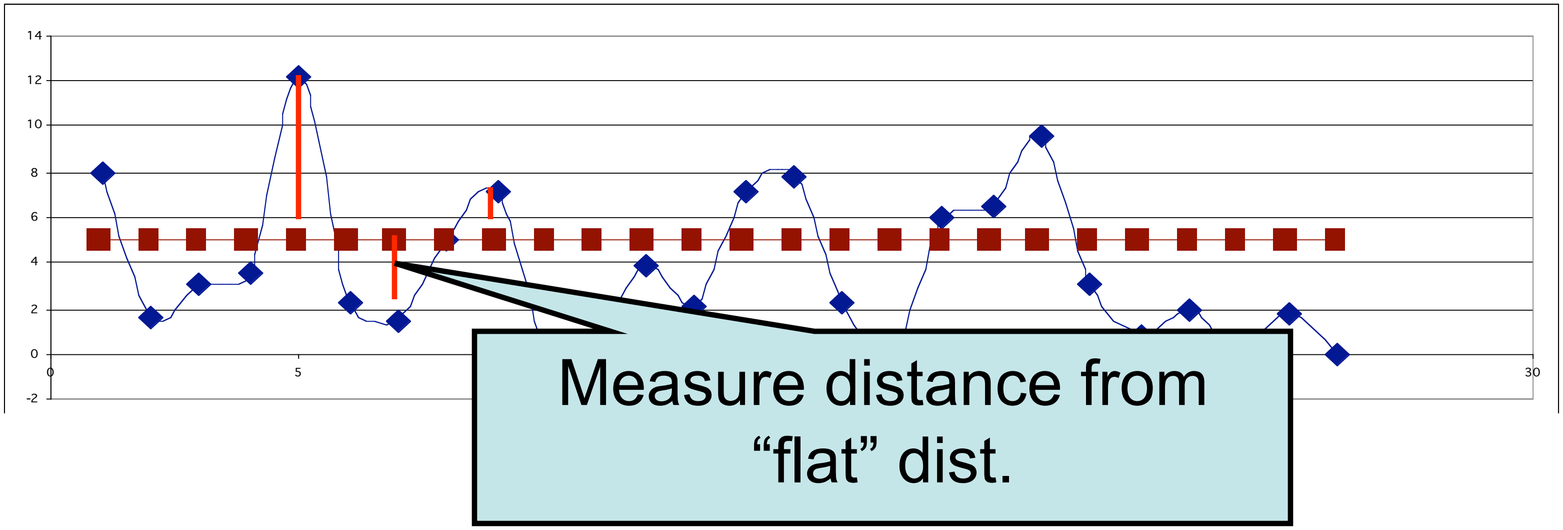
Cryptanalysis Continued

- Once key length is guessed to be k…
- Split ciphertext into k slices
  - Single letter frequency distribution for each slice should resemble English distribution

- How do we tell whether a particular distribution is a good match for another?
  - Let prob(α) be the probability for letter α
  - In a perfectly flat distribution

$$\text{prob}(\alpha) = 1/26 \approx 0.0384$$

Variance: Measure of “roughness”



$$\mathrm{Var} = \sum_{\alpha = a}^{\alpha = z} (\mathrm{prob}(\alpha) - 1/26)^2$$

$$= \ldots$$

$$= \left(\sum_{\alpha = a}^{\alpha = z} \mathrm{prob}(\alpha)^2 \right) - 1/26$$

Estimate Variance From Frequency

- $prob(\alpha)^2$ is probability that any two characters drawn from the text will be α
- Suppose there are n ciphertext letters total
- Suppose $freq(\alpha)$ is the frequency of α
- What is likelihood of picking α twice at random?
  - $freq(\alpha)$ ways of picking the first α
  - $(freq(\alpha) - 1)$ ways of picking the second α
  - But this counts twice because $(\alpha,\beta) = (\beta,\alpha)$
  - So

$$\frac{freq(\alpha) \times (freq(\alpha) - 1)}{2}$$

Index of Coincidence

- But there are $\frac{n \times (n-1)}{2}$ pairs of letters

- …so prob(α) is roughly $\frac{freq(\alpha) \times (freq(\alpha) - 1)}{n \times (n-1)}$

- Index of coincidence: approximates variance from frequencies

$$IC = \sum_{\alpha = a}^{\alpha = z} \frac{freq(\alpha) \times (freq(\alpha) - 1)}{n \times (n-1)}$$

What's it good for?

- If the distribution is flat, then IC ≈ 0.0384
- If the distribution is like English, then IC ≈ 0.068
- Can verify key length:

keylen	1	2	3	4	5	many
IC	0.068	0.052	0.047	0.044	0.044	… 0.038

Summary: Cracking Polyalphabetics

- Use Kasiski method to guess likely key lengths
- Compute the Index of Coincidence to verify key length k
- k-Slices should have IC similar to English

- Note: digram information harder to use for polyalphabetic ciphers…
  - May want to consider “split digrams”
  - Example: if tion is a common sequence k=2 then “t?o” and “i?n” are likely “split digrams”

Perfect Substitution Ciphers

$$
\begin{array}{r}
p_1\ p_2\ p_3\ \dots\ p_n \\
\oplus\ \underline{b_1\ b_2\ b_3\ \dots\ b_n} \\
c_1\ c_2\ c_3\ \dots\ c_n
\end{array}
$$

- Choose a string of random bits the same length as the plaintext, XOR them to obtain the ciphertext.
- Perfect Secrecy
  - Probability that a given message is encoded in the ciphertext is unaltered by knowledge of the ciphertext
  - Proof: Give me any plaintext message and any ciphertext and I can construct a key that will produce the ciphertext from the plaintext.

One-time Pads

- Another name for Perfect Substitution
- Actually used by US agents in Russia
  - Physical pad of paper
  - List of random numbers
  - Pages were torn out and destroyed after use
  - “Numbers Stations”?
- Vernam Cipher
  - Used by AT&T
  - Random sequence stored on punch tape
- Not practical for general purpose crypography
  - But useful as component in other protocols.

Problems with "Perfect" Substitution

- Key is the same length as the plaintext
  - Sender and receiver must agree on the same random sequence
  - Not any easier to transmit key securely than to transmit plaintext securely
- Need to be able to generate many truly random bits
  - Pseudorandom numbers generated by an algorithm aren't good enough for long messages
  - Must be careful: Remember the RC4 algorithm from WEP.
- Can't reuse the key
  - Not enough confusion

Diffusion and Confusion

- Diffusion
  - Ciphertext should look random
  - Protection against statistical attacks
  - Monoalphabetic -> Polyalphabetic substitution; diffusion increases
- Confusion
  - Make the relation between the key, plaintext and ciphertext complex
  - Lots off confusion -> hard to calculate key in a known plaintext attack
  - Polyalphabetic substitution: little confusion

Computational Security

- Perfect Ciphers are *unconditionally secure*
  - No amount of computation will help crack the cipher (i.e. the *only* strategy is brute force)
- In practice, strive for *computationally security*
  - Given enough power, the attacker could crack the cipher (example: brute force attack)
  - But, an attacker with only *bounded resources* is extremely unlikely to crack it
  - Example: Assume attacker has only polynomial time, then encryption algorithm that can't be inverted in less than exponential time is secure.
  - Results are usually stated *probabilistically*

Kinds of Industrial Strength Crypto

- Shared Key Cryptography
- Public Key Cryptography
- Cryptographic Hashes

- All of these aim for computational security
  - Not all methods have been proved to be intractable to crack.

Shared Key Cryptography

- Sender & receiver use the same key
- Key must remain private
- Also called *symmetric* or *secret key* cryptography
- Often are *block-ciphers*
  - Process plaintext data in blocks
- Examples: DES, Triple-DES, Blowfish, Twofish, AES, Rijndael, …

Shared Key Notation

- Encryption algorithm

  E : key x plain → cipher

  Notation: K{msg} = E(K, msg)

- Decryption algorithm

  D : key x cipher → plain

- D inverts E

  D(K, E(K, msg)) = msg

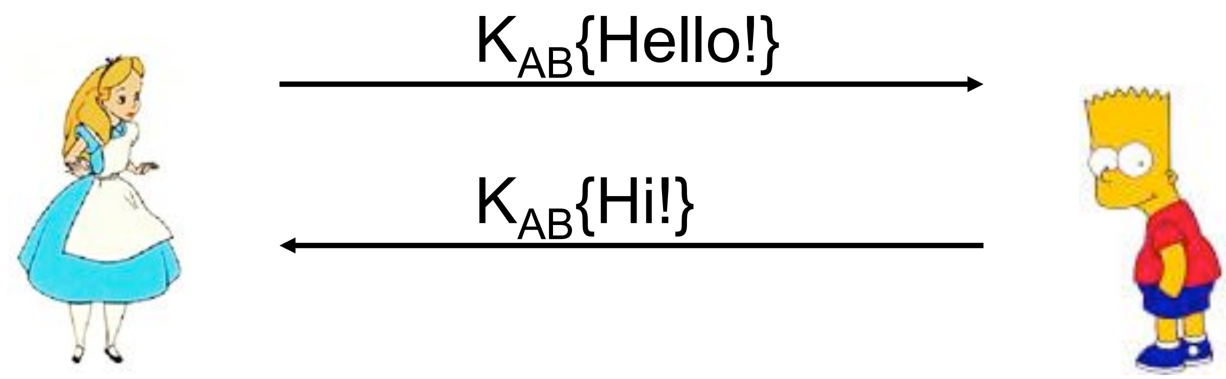
- Use capital "K" for shared (secret) keys

- Sometimes E is the same algorithm as D

Secure Channel: Shared Keys

Alice

Bart



K_{AB}

K_{AB}