Common English Digrams and Trigrams

<table>
<thead>
<tr>
<th>Digrams</th>
<th>Trigrams</th>
</tr>
</thead>
<tbody>
<tr>
<td>EN</td>
<td>ENT</td>
</tr>
<tr>
<td>RE</td>
<td>ION</td>
</tr>
<tr>
<td>ER</td>
<td>AND</td>
</tr>
<tr>
<td>NT</td>
<td>ING</td>
</tr>
<tr>
<td>TH</td>
<td>IVE</td>
</tr>
<tr>
<td>ON</td>
<td>TIO</td>
</tr>
<tr>
<td>IN</td>
<td>FOR</td>
</tr>
<tr>
<td>TF</td>
<td>OUR</td>
</tr>
<tr>
<td>AN</td>
<td>THI</td>
</tr>
<tr>
<td>OR</td>
<td>ONE</td>
</tr>
</tbody>
</table>

Character Frequencies (English)

monoalphabetic cryptanalysis

- See class example

Security in Computing

Chapter 2

Elementary Cryptography (part 2)
Vigenere Cipher

- construct a table (a Vigenere tableau)
- each row in table is a different shift (alphabet)
- sender and receiver agree on sequence of rows
- helps to disguise patterns

Vigenere 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 |
| 0 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 1 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 2 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 5 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 6 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 7 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 8 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 9 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 10 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

cryptographer's counter-move

- cryptanalysts use properties of plaintext
- what can be cryptographers' counter-moves?

Hiding Patterns

- polyalphabetic ciphers
  - use multiple alphabets
- homophonic ciphers
  - multiple possible output characters for an input character
- polygram ciphers
  - encipher groups of letters at once
### More on Vigenere Keys

- usually think of choice of rows as a keyword
- example: keyword “BASE”
- row order is b,a,s,e,b,a,s,e, ...

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

### Vigenere Example

- suppose we agree on the key \( \{1,5,9,16,21,22\} \)
- encrypt:
  - char 1 with row 1
  - '2' with row 5
  - '3' with row 9
  - ...
  - char 6 with row 22
  - char 7 with row 1
  - char 8 with row 5
  - etc.

### Don't Need to Construct the Table

- keyword = BRAKE
  - equivalent \( K = \{1, 17, 0, 10, 4\} \)
- plaintext “I am sick of school”
- convert to numeric vals
- add to \( K \) mod 26

<table>
<thead>
<tr>
<th>M</th>
<th>A</th>
<th>M</th>
<th>S</th>
<th>I</th>
<th>C</th>
<th>K</th>
<th>O</th>
<th>F</th>
<th>S</th>
<th>C</th>
<th>H</th>
<th>O</th>
<th>O</th>
<th>L</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>0</td>
<td>12</td>
<td>18</td>
<td>8</td>
<td>2</td>
<td>10</td>
<td>14</td>
<td>5</td>
<td>18</td>
<td>2</td>
<td>7</td>
<td>14</td>
<td>14</td>
<td>11</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>B</th>
<th>R</th>
<th>A</th>
<th>K</th>
<th>E</th>
<th>B</th>
<th>R</th>
<th>A</th>
<th>K</th>
<th>E</th>
<th>B</th>
<th>R</th>
<th>A</th>
<th>K</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>7</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>2</td>
</tr>
</tbody>
</table>

### Vigenere Example

- \( K = 1,5,9,16,21,22 \)
- \( E_k(M) = DMJFKAMQN \)

- note: our key is in ascending order. this isn't required

### Vigenere Example

- \( M = \text{“Chappelle”} \)
- \( k = \) DMJFKAMQN
### Homophonic Ciphers (cont'd)

- Are there disadvantages to this?

### Homophonic Example

<table>
<thead>
<tr>
<th>Plaintext</th>
<th>Homophones</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>624 18 329 19 4</td>
</tr>
<tr>
<td>B</td>
<td>5 333 511</td>
</tr>
<tr>
<td>C</td>
<td>919 14 67 83</td>
</tr>
<tr>
<td>D</td>
<td>414 309 238 71 15 6</td>
</tr>
<tr>
<td>E</td>
<td>8 13 12</td>
</tr>
<tr>
<td>F</td>
<td>61 422</td>
</tr>
<tr>
<td>G</td>
<td>413 2 16</td>
</tr>
</tbody>
</table>

... ... ... ... ... ...

- So “cabbage” could be encrypted as:
  - 14 329 511 5 624 2 8

### Don't Need to Construct the Table

- keyword = BRAKE
  - equivalent $K=\{1, 17, 0, 10, 4\}$
- plaintext “I am sick of school”
- convert to numeric vals
- add to $K$ mod 26

| I A M S I C K O F S C H O O L | 8 0 12 18 8 2 10 14 5 18 2 7 14 14 11 |
| B R A K E B R A K E | 1 17 0 10 4 1 17 0 10 4 1 17 0 10 4 |

Encrypts to: J RM CMDB OP WDYOYP

- set of possible ciphertext characters that map to a single plaintext character $m$ called **homophones**
Playfair Cipher

- 1850s. named after Playfair
- actually invented by his friend Wheatstone
- write keyword without dups. into 5x5 matrix
- treat I and J as the same character
- example:
  - keyword “MACARONI”
  -

<table>
<thead>
<tr>
<th>M</th>
<th>A</th>
<th>C</th>
<th>R</th>
<th>O</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>I/J</td>
<td>B</td>
<td>D</td>
<td>E</td>
</tr>
<tr>
<td>F</td>
<td>G</td>
<td>H</td>
<td>K</td>
<td>L</td>
</tr>
<tr>
<td>P</td>
<td>Q</td>
<td>S</td>
<td>T</td>
<td>U</td>
</tr>
<tr>
<td>V</td>
<td>W</td>
<td>X</td>
<td>Y</td>
<td>Z</td>
</tr>
</tbody>
</table>

Playfair Encryption

- divide plaintext into pairs
- double characters separated by dummy character (e.g. x)
  - mi ss is si pp i becomes
  - mi sx si sx si px pi
- if plaintext has odd number of chars, append dummy char.
- encrypt plaintext pairs
  - only 3 possibilities
    - same row
    - same column
    - different row and col.

Homophonic Ciphers (cont'd)

- Are there disadvantages to this?
  - ciphertext longer than the plaintext

- How many homophones per plaintext char?
  - fixed number
  - variable: more for frequent plaintext characters

Polygram Ciphers

- simple substitution ciphers, e.g. shift ciphers, keyword mixed alphabet, (even Vigenere tableau) ... substitute one character for another character
- polygram ciphers substitute a group of characters for another group of characters
- goal: make frequency analysis more difficult
- examples:
  - playfair cipher
  - hill cipher
One time pad

- Idea:
  - Take a stream of random data (keystream)
    - used to be physically on a pad.
    - rip out as many random pages as you need.
  - Combine it with plaintext to form ciphertext
- Message receiver uses same keystream to recover plaintext
- If the stream is truly random → perfect security
- Why don't we use this all the time?

One time pad

- Idea:
  - Take a stream of random data (keystream)
  - Combine it with plaintext to form ciphertext
- Message receiver uses same keystream to recover plaintext
- If the stream is truly random → perfect security
- Why don't we use this all the time?
  - How do we get unlimited truly random stream?
  - If we could get it, how do we distribute it?
  - What if sender and receiver aren't synchronized?

Playfair encryption (cont'd)

- same row
  - substitute with chars to right
    - examples: $MC \rightarrow AR$, $RM \rightarrow OA, SU \rightarrow TP$
  - same col
    - substitute with chars below
    - examples: $EU \rightarrow LZ$, $GW \rightarrow QA$
- different row and col → tricky

Playfair – different row and col

- substitute plaintext letter with letter that
  - is in its own row
  - and is in the column of the other plaintext letter
- example, AT becomes RQ

---

Pfleeger, Security in Computing, ch. 2
Quick Quiz

- I have:
  - DVD (∼ 5 GBytes) of random data
  - a 1.5 Mbps DSL
- If I copy the DVD and give it to a friend, how long can we use it as a one-time pad?

\[
\frac{(5 \text{ GBytes} \times 1024 \text{ MBytes/GByte} \times 8 \text{ bits/byte})}{1.5 \text{ Mbps}} = 27,306 \text{ secs.} = \text{about 7.5 hours}
\]
XOR (cont'd)

\[ a \oplus a = ? \]

\[ a \oplus b \oplus b = ? \]

\[ a \oplus a = 0 \]

\[ a \oplus b \oplus b = a \]

**XOR is an involution**

XOR

- **Review:** XOR logic operator

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>A XOR B</th>
</tr>
</thead>
<tbody>
<tr>
<td>FALSE</td>
<td>FALSE</td>
<td>FALSE</td>
</tr>
<tr>
<td>FALSE</td>
<td>TRUE</td>
<td>TRUE</td>
</tr>
<tr>
<td>TRUE</td>
<td>FALSE</td>
<td>TRUE</td>
</tr>
<tr>
<td>TRUE</td>
<td>TRUE</td>
<td>FALSE</td>
</tr>
</tbody>
</table>

XOR (cont'd)

- XOR often denoted \( \oplus \)
- Don't have to write the words TRUE, FALSE
- The following are equivalent:

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>A ( \oplus ) B</th>
</tr>
</thead>
<tbody>
<tr>
<td>FALSE</td>
<td>FALSE</td>
<td>FALSE</td>
</tr>
<tr>
<td>FALSE</td>
<td>TRUE</td>
<td>TRUE</td>
</tr>
<tr>
<td>TRUE</td>
<td>FALSE</td>
<td>TRUE</td>
</tr>
<tr>
<td>TRUE</td>
<td>TRUE</td>
<td>FALSE</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>A ( \oplus ) B</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Aside: Involutions

Let:

- $S$ be a finite set
- $f$ a bijection ($1 \to 1$, onto) from $S$ to $S$ (i.e. $f: S \to S$)
- $f$ is an involution if $f = f^{-1}$
  - i.e. $f(f(x)) = x$
- So XOR is an involution

XOR Example

- can encrypt by XORing plaintext with keystream
- Example: plaintext = “Chappelle”

XOR in Java

- a XOR b in Java is $a \oplus b$
  - same in C, C++, Perl
- Code that produced the example:

  ```java
  System.out.println("c\t\tc(bin)\t\tkey\tkey(bin)\tc XOR k");
  for (int i=0; i<plaintext.length(); i++)
  {
      System.out.println(plaintext.charAt(i) + "\t" +
                         Integer.toBinaryString((int)plaintext.charAt(i)) + "\t" +
                         key[i] + "\t" + Integer.toBinaryString(key[i]) + "\t" +
                         Integer.toBinaryString((int)plaintext.charAt(i) ^ key[i]));
  }
  ```

XOR Example

- can encrypt by XORing plaintext with keystream
- Example: plaintext = “Chappelle”

- Question: if I have the keystream, how do I decrypt?
  - XOR it the keystream with the ciphertext.
Combining Plaintext with Keystream

- Besides XOR, for text, you can add key to data mod 26
- Example:

  | Plaintext | D A V E A T T E L |
  | Key       | J O M P K R L Q E |
  | P + K (mod 26) | 12 14 7 19 10 10 4 20 15 |

  | ciphertext | M O H T K K E U P |

Vernam Cipher

- type of one-time pad
- combine an arbitrarily long nonrepeating series of numbers with the plaintext to form ciphertext
- originally implemented as a paper tape attached to a teletype machine

Key Reuse

- What happens if you use the same key twice?

  \[ C_1 = P_1 \oplus K \quad C_2 = P_2 \oplus K \]

“Book” Ciphers

- construct a poor man's one-time pad
- get “randomness” from:
  - novels
  - newspapers
  - telephone books
  - pieces of music
  - decks of cards
5 rotor machine

- For a 5 rotor machine, $26^5$ substitution alphabets before machine repeats
- For a practical break based on letter frequency:
  - “The ciphertext would have to be as long as all the speeches made on the floor of the Senate and House of Representatives in three successive sessions of Congress” -- Kahn, *The Codebreakers*

Enigma Exhibit at NSA

- enigma exhibit at the NSA
  - [http://www.nsa.gov/museum/museu00007.cfm](http://www.nsa.gov/museum/museu00007.cfm)
- java enigma simulator
  - [http://russells.freeshell.org/enigma/](http://russells.freeshell.org/enigma/)

Key Reuse

- What happens if you use the same key twice?

\[
C_1 = P_1 \oplus K \\
C_2 = P_2 \oplus K \\
C_1 \oplus C_2 = P_1 \oplus K \oplus P_2 \oplus K \\
... = P_1 \oplus P_2
\]

much easier to solve

Rotor Machines

- Implements a kind of Vigenere tableau
- Physically:
  - keypad
  - several rotors
  - keypad wired to a rotor, and rotors wired to each other
- After each key is pressed, at least one rotor spins
- rotors positions don't repeat until $26^{\#rotors}$ keys have been pressed
- effect: $26^{\#rotors}$ substitution alphabets
- WWII examples:
  - Enigma
  - Purple
2.3 Transposition Ciphers

- Rearrange P to get C
- Example:
  - P = BOREDOM
  - C = MOODERB

Rotor Machines. Why?

- Why mention rotor machines?
  - They're not used but lead to DES

Columnar Transposition

- Use a two-dimensional array (matrix)
- P = “NARCOLEPTIC”
- C formed by reading down columns
  - “TOTALIRECCP”
- Can also reorder columns
  - 2, 1, 3, 4 becomes “ALINOTRECCP”

Chapter Outline

- 2.1 Terminology and Background
- 2.2 Substitution Ciphers
- 2.3 Transpositions (Permutations)
- 2.4 Making “Good” Encryption Algorithms
- 2.5 The Data Encryption Standard (DES)
- 2.6 The AES Algorithm
- 2.7 Public Key Encryption
- 2.8 Uses of Encryption
- 2.9 Summary
Breaking a Transposition Cipher

1) Figure out that it's a transposition cipher
   - ciphertext chars will have same frequency as plaintext

2) Break the transposition
   - use common letter pairs (digrams), triples (trigrams) to figure out \( d \)

More General Transposition

- many transpositions use fixed period \( d \)
- Let \( Z_d \) be the integers from 1 to \( d \)
- Let \( f:Z_d \rightarrow Z_d \) be a permutation over \( Z_d \)
- Key for the cipher is \( K=(d,f) \)
- message \( M=m_1,m_2,\ldots,m_d,m_{d+1},\ldots,m_{2d},\ldots \)
- ciphertext \( C=m_{f(1)},m_{f(2)},\ldots,m_{f(d)},m_{d+f(1)},\ldots,m_{d+f(d)},\ldots \)
- this is easier to see with an example

Common English Digrams and Trigrams

<table>
<thead>
<tr>
<th>Digrams</th>
<th>Trigrams</th>
</tr>
</thead>
<tbody>
<tr>
<td>EN</td>
<td>ENT</td>
</tr>
<tr>
<td>RE</td>
<td>ION</td>
</tr>
<tr>
<td>ER</td>
<td>AND</td>
</tr>
<tr>
<td>NT</td>
<td>ING</td>
</tr>
<tr>
<td>TH</td>
<td>IVE</td>
</tr>
<tr>
<td>ON</td>
<td>TIO</td>
</tr>
<tr>
<td>IN</td>
<td>FOR</td>
</tr>
<tr>
<td>TF</td>
<td>OUR</td>
</tr>
<tr>
<td>AN</td>
<td>THI</td>
</tr>
<tr>
<td>OR</td>
<td>ONE</td>
</tr>
</tbody>
</table>

General Transposition Example

- suppose that the period \( d = 4 \)
- suppose that \( f \) is:

\[
\begin{array}{|c|c|c|c|}
\hline
i & 1 & 2 & 3 & 4 \\
\hline
f(i) & 2 & 4 & 1 & 3 \\
\hline
\end{array}
\]

- \( M = \text{"AGGRAVATION"} \)

\[
\begin{array}{|c|c|c|c|c|}
\hline
M & A & G & G & R \\
\hline
E(M) & G & R & A & G \\
\hline
\end{array}
\]

- short block at the end:
  - chars in \( C \) in relative position in permutation
  - e.g. 2 is before 3 in \( f(i) \) so I is before N
Combinations of Approaches

- If it's not too difficult to break:
  - basic substitutions
  - basic permutations
- Use a combination of the two → **product cipher**
  - composition of functions
  - stronger than the separate parts
- substitution adds **confusion**
- transposition adds **diffusion**

Diffusion

- Other than simple permutations, is there anything else that we can do to provide diffusion?
  - Anything else that we can do to spread the information around, *e.g.*
    - add redundant information
    - steganography