Basics of Data Encryption
CS 239
Computer Security
January 24, 2005

Outline
• What is data encryption?
• Basic encryption mechanisms
• Stream and block ciphers
• Characteristics of good ciphers

Data Encryption Concepts
• Introduction
• Terminology
• Basics of encryption algorithms
• Cryptanalysis

Introduction to Encryption
• Much of computer security is about keeping secrets
• One method is to make it hard for others to read
• While (usually) making it simple for authorized parties to read

Encryption
• Encryption is the process of hiding information in plain sight
• Transform the secret data into something else
• Even if the attacker can see the transformed data, he can’t understand the underlying secret

Encryption and Data Transformations
• Encryption is all about transforming the data
• One bit or byte pattern is transformed to another bit or byte pattern
• Usually in a reversible way
Encryption Terminology

- Encryption is typically described in terms of sending a message
  - Though it’s used for many other purposes
- The sender is S
- The receiver is R
- The transmission medium is T
- And the attacker is O

More Terminology

- Encryption is the process of making message unreadable/unalterable by O
- Decryption is the process of making the encrypted message readable by R
- A system performing these transformations is a cryptosystem
  - Rules for transformation sometimes called a cipher

Plaintext and Ciphertext

- Plaintext is the original form of the message (often referred to as P)
  - Transfer $100 to my savings account
- Ciphertext is the encrypted form of the message (often referred to as C)
  - Sqzmredq 099 sn lx rzuhmfr bbntms

Very Basics of Encryption Algorithms

- Most use a key to perform encryption and decryption
  - Referred to as K
- The key is a secret
- Without the key, decryption is hard
- With the key, decryption is easy

Terminology for Encryption Algorithms

- The encryption algorithm is referred to as E()
- C = E(K, P)
- The decryption algorithm is referred to as D()
- The decryption algorithm also has a key

Symmetric and Asymmetric Encryption Systems

- Symmetric systems use the same keys for E and D:
  \[ P = D(K, C) \]
  Expanding, \[ P = D(K, E(K, P)) \]
- Asymmetric systems use different keys for E and D:
  \[ C = E(K_P, P) \]
  \[ P = D(K_D, C) \]
Characteristics of Keyed Encryption Systems

- If you change only the key, a given plaintext encrypts to a different ciphertext
- Same applies to decryption
- Decryption should be hard without knowing the key

Cryptanalysis

- The process of trying to break a cryptosystem
- Finding the meaning of an encrypted message without being given the key

Forms of Cryptanalysis

- Analyze an encrypted message and deduce its contents
- Analyze one or more encrypted messages to find a common key
- Analyze a cryptosystem to find a fundamental flaw

Breaking Cryptosystems

- Most cryptosystems are breakable
- Some just cost more to break than others
- The job of the cryptosystem is to make the cost infeasible
  – Or incommensurate with the benefit extracted

Types of Attacks on Cryptosystems

- Ciphertext only
- Known plaintext
- Chosen plaintext
  – Differential cryptanalysis
- Algorithm and ciphertext
  – Timing attacks

Ciphertext Only

- No a priori knowledge of plaintext
- Or details of algorithm
- Must work with probability distributions, patterns of common characters, etc.
- Hardest type of attack
**Known Plaintext**

- Full or partial
- Cryptanalyst has matching sample of ciphertext and plaintext
- Or may know something about what ciphertext represents
  - E.g., an IP packet with its headers

**Chosen Plaintext**

- Cryptanalyst can submit chosen samples of plaintext to the cryptosystem
- And recover the resulting ciphertext
- Clever choices of plaintext may reveal many details

**Differential Cryptanalysis**

- Iteratively choose plaintexts that differ slightly in carefully chosen ways
- A good crypto algorithm should produce results that don’t help analysis
- But some crypto algorithms are vulnerable to this attack

**Algorithm and Ciphertext**

- Cryptanalyst knows the algorithm and has a sample of ciphertext
- But not the key, and may not get any more similar ciphertext
- Can use “exhaustive” runs of algorithm against guesses at plaintext
- Password guessers often work this way

**Timing Attacks**

- Usually assume knowledge of algorithm
- And ability to watch algorithm encrypting/decrypting
- Some algorithms perform different operations based on key values
- Watch timing to try to deduce keys
- Has been successful against crypto in some smart cards

**Basic Encryption Methods**

- Substitutions
  - Monoalphabetic
  - Polyalphabetic
- Permutations
Substitution Ciphers

- Substitute one or more characters in a message with one or more different characters
- Using some set of rules
- Decryption is performed by reversing the substitutions

Example of a Simple Substitution Cipher

How did this transformation happen?

Transfer $100 to my savings account

Square red #099 sn lx rzuhmfr zbbntms

Sransfer $100 to my savings account

Square red #099 sn lx rzuhmfr zbbntms

Stransfer $100 to my savings account

Square red #099 sn lx rzuhmfr zbbntms

Caesar Ciphers

- A simple substitution cipher like the previous example
  – Supposedly invented by Julius Caesar
- Translate each letter a fixed number of positions in the alphabet
- Reverse by translating in opposite direction

Is the Caesar Cipher a Good Cipher?

- Well, it worked great 2000 years ago
- It’s simple, but
  - Fails to conceal many important characteristics of the message
  - Which makes cryptanalysis easier
- Limited number of useful keys

How Would Cryptanalysis Attack a Caesar Cipher?

- Letter frequencies
- In English (and other alphabetic languages), some letters occur more frequently than others
- Caesar ciphers translate all occurrences of a given letter into the same cipher letter
- All you need is the offset

More On Frequency Distributions

- In most languages, some letters used more than others
  – In English, “e,” “t,” and “s” common
- True even in non-natural languages
  – Certain characters appear frequently in C code
  – Zero appears often in much numeric data
Cryptanalysis and Frequency Distribution

• If you know what kind of data was encrypted, you can (often) use frequency distributions to break it
• Especially for Caesar ciphers
  – And other simple encryption algorithms

Breaking Monoalphabetic Ciphers

• Identify (or guess) kind of data
• Count frequency of each encrypted symbol
• Match to observed frequencies of other symbols in other kinds of data
• Provides probable mapping of cipher
• The more ciphertext available, the more reliable this technique

Example

• With ciphertext “Sqzmrdez #099 sn lx rzuhmfr zbbntms”
• Frequencies -

<table>
<thead>
<tr>
<th>Letter</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>0</td>
</tr>
<tr>
<td>b</td>
<td>2</td>
</tr>
<tr>
<td>c</td>
<td>0</td>
</tr>
<tr>
<td>d</td>
<td>1</td>
</tr>
<tr>
<td>e</td>
<td>1</td>
</tr>
<tr>
<td>f</td>
<td>1</td>
</tr>
<tr>
<td>g</td>
<td>0</td>
</tr>
<tr>
<td>h</td>
<td>1</td>
</tr>
<tr>
<td>i</td>
<td>0</td>
</tr>
<tr>
<td>j</td>
<td>0</td>
</tr>
<tr>
<td>k</td>
<td>0</td>
</tr>
<tr>
<td>l</td>
<td>1</td>
</tr>
<tr>
<td>m</td>
<td>3</td>
</tr>
<tr>
<td>n</td>
<td>2</td>
</tr>
<tr>
<td>o</td>
<td>0</td>
</tr>
<tr>
<td>p</td>
<td>0</td>
</tr>
<tr>
<td>q</td>
<td>2</td>
</tr>
<tr>
<td>r</td>
<td>3</td>
</tr>
<tr>
<td>s</td>
<td>3</td>
</tr>
<tr>
<td>t</td>
<td>1</td>
</tr>
<tr>
<td>u</td>
<td>1</td>
</tr>
<tr>
<td>v</td>
<td>0</td>
</tr>
<tr>
<td>w</td>
<td>0</td>
</tr>
<tr>
<td>x</td>
<td>1</td>
</tr>
<tr>
<td>y</td>
<td>0</td>
</tr>
<tr>
<td>z</td>
<td>3</td>
</tr>
</tbody>
</table>

Applying Frequencies To Our Example

• The most common English letters are typically “e,” “t,” “a,” “o,” and “s”
• Four out of five of the common English letters in the plaintext map to these letters

Cracking the Caesar Cipher

• Since all substitutions are offset by the same amount, just need to figure out how much
• How about +1?
  – That would only work for a=>b
• How about -1?
  – That would work for t=>s, a=>z, o=>n, and s=>r
  – Try it on the whole message and see if it looks good

More Complex Substitutions

• Monoalphabetic substitutions - Each plaintext letter maps to a single, unique ciphertext letter
• Any mapping is permitted
• Key can provide method of determining the mapping
  – Key could be the mapping
Are These Monoalphabetic Ciphers Better?

- Only a little
- Finding the mapping for one character doesn’t give you all mappings
- But the same simple techniques can be used to find the other mappings
- Generally insufficient for anything serious

Codes and Monoalphabetic Ciphers

- Codes are sometimes considered different than ciphers
- A series of important words or phrases are replaced with meaningless words or phrases
- E.g., “Transfer $100 to my savings account” becomes
  – “The hawk flies at midnight”

Are Codes More Secure?

- Depends
- Frequency attacks based on letters don’t work
- But frequency attacks based on phrases may
- And other tricks may cause problems
- In some ways, just a limited form of substitution cipher
- Weakness based on need for codebook
  – Can your codebook contain all message components?

Superencipherment

- First translate message using a code book
- Then encipher the result
- If opponent can’t break cipher, great
- If he can, he still has to break the code
- Depending on several factors, may (or may not) be better than just a cipher
- Popular during WWII (but the Allies still read Japan’s and Germany’s messages)

Polyalphabetic Ciphers

- Ciphers that don’t always translate a given plaintext character into the same ciphertext character
- For example, use different substitutions for odd and even positions

Example of Simple Polyalphabetic Cipher

- Move one character “up” in even positions, one character “down” in odd positions
- Note that same character translates to different characters in some cases

Transfer $100 to my savings account
Sszorgds V019 sp nx tbujmhr zdbptos
Are Polyalphabetic Ciphers Better?

- Depends
- On how easy it is to determine the pattern of substitutions
- If it’s easy, then you’ve gained little

Cryptanalysis of Our Example

- Consider all even characters as one set
- And all odd characters as another set
- Apply basic cryptanalysis to each set
- The transformations fall out easily

How About For More Complex Patterns?

- Good if the attacker doesn’t know the choices of which characters get transformed which way
- Attempt to hide patterns well
- But known methods still exist for breaking them

Methods of Attacking Polyalphabetic Ciphers

- Kasiski method tries to find repetitions of the encryption pattern
- Index of coincidence predicts the number of alphabets used to perform the encryption
- Both require lots of ciphertext

How Does the Cryptanalyst “Know” When He’s Succeeded?

- Every key translates a message into something
- If a cryptanalyst thinks he’s got the right key, how can he be sure?
- Usually because he doesn’t get garbage when he tries it
- Chances are he will get garbage from any other key
- Why?

The Unbreakable Cipher

- There is a “perfect” substitution cipher
- One that is theoretically (and practically) unbreakable without the key
One-Time Pads

- Essentially, use a new substitution alphabet for every character
- Substitution alphabets chosen purely at random
  - These constitute the key
- Provably unbreakable without knowing this key

Example of One Time Pads

- Usually explained with bits, not characters
- We shall use a highly complex cryptographic transformation:
  - XOR
- And a three bit message
  - 010

One Time Pads at Work

Flip some coins to get random numbers

Apply our sophisticated cryptographic algorithm

Any key was equally likely

What’s so secure about that?

Any plaintext could have produced this message with one of those keys

Security of One-Time Pads

- If the key is truly random, provable that it can’t be broken without the key
- But there are problems
- Need one bit of key per bit of message
- Key distribution is painful
- Synchronization of keys is vital
- A good random number generator is hard to find

Attacking One-Time Pads

- Not much fun
  - But then, neither is using them
- Essentially, you attack their random number generator
- Hope that it isn’t really random, and try to find its non-random characteristics

One-Time Pads and Cryptographic Snake Oil

- Companies regularly claim they have “unbreakable” cryptography
- Usually based on one-time pads
- But typically misused
  - Pads distributed with some other crypto mechanism
  - Pads generated with non-random process
  - Pads reused
Permutation Ciphers
• Instead of substituting different characters, scramble up the existing characters
• Use algorithm based on the key to control how they’re scrambled
• Decryption uses key to unscramble

Characteristics of Permutation Ciphers
• Doesn’t change the characters in the message
  – Just where they occur
• Thus, character frequency analysis doesn’t help cryptanalyst

Columnar Transpositions
• Write the message characters in a series of columns
• Copy from top to bottom of first column, then second, etc.

Example of Columnar Substitution
How did this transformation happen?

\[
\text{Transfer} \ \$100 \ \text{to} \ \text{money savings account} \ \rightarrow \ \text{Te0yncrr goa tssun$n$oa ns1 vat0mic}
\]

Attacking Columnar Transformations
• The trick is figuring out how many columns were used
• Use information about digrams, trigrams, and other patterns
• Digrams are letters that frequently occur together (re, th, en, for example)
• For each possibility, check digram frequency

For Example,
• In our case, the presence of numerals in the text is suspicious
  – One might guess the numerals belong together
  – And maybe the dollar sign with them
• Most of this analysis is more complicated
Double Transpositions

• Do it twice
• Using different numbers of columns each time
• Find pairs of letters that probably appeared together in the plaintext
• Figure out what transformations would put them in their positions in the ciphertext

Generalized Transpositions

• Any algorithm can be used to scramble the text
• Usually somehow controlled by a key
• Generality of possible transpositions makes cryptanalysis harder

Which Is Better, Transposition or Substitution?

• Well, neither, really
• Strong modern ciphers tend to use both
• Transposition scrambles text patterns
• Substitution hides underlying text characters/bits
• Combining them can achieve both effects
  – If you do it right . . .

Stream and Block Ciphers

• Stream ciphers convert one symbol of plaintext immediately into one symbol of ciphertext
• Block ciphers work on a given sized chunk of data at a time

Stream Ciphers

Key

Plaintext → Encryption → Ciphertext

Advantages of Stream Ciphers

+ Speed of encryption and decryption
  • Each symbol encrypted as soon as it’s available
+ Low error propagation
  • Errors affect only the symbol where the error occurred
Disadvantages of Stream Ciphers

- Low diffusion
  - Each symbol separately encrypted
  - Each ciphertext symbol only contains information about one plaintext symbol
- Susceptible to insertions and modifications
- Not good match for many common uses of cryptography

Block Ciphers

Advantages of Block Ciphers

+ Diffusion
  - Easier to make a set of encrypted characters depend on each other
+ Immunity to insertions
  - Encrypted text arrives in known lengths

Disadvantages of Block Ciphers

- Slower
  - Need to wait for block of data before encryption/decryption starts
- Worse error propagation
  - Errors affect entire blocks

Characteristics of Good Ciphers

- Well matched to requirements of application
  - Amount of secrecy required should match labor to achieve it
- Freedom from complexity
  - The more complex algorithms or key choices are, the worse

More Characteristics

- Simplicity of implementation
  - Seemingly more important for hand ciphering
  - But relates to probability of errors in computer implementations
- Errors should not propagate
Yet More Characteristics

- Ciphertext size should be same as plaintext size
- Encryption should maximize confusion
  - Relation between plaintext and ciphertext should be complex
- Encryption should maximize diffusion
  - Plaintext information should be distributed throughout ciphertext