

## ETP Cover Sheet

Title of ETP	Intro to Cryptography
Name of IISME Fellow	Eric Ferrante
Fellow's year-round email	<a href="mailto:eric_ferrante@fuhisd.org">eric_ferrante@fuhisd.org</a>
Sponsor Company	Hewlett Packard
Name of Mentor	Tom Collins
National Bd Certificate Area	Mathematics 9-12

Category	<p><i>Curriculum</i></p> <p>Subject: <u>Math</u>    Science    Technology _____</p> <p>Level: Elem    Middle    <u>High</u>    Other</p> <p><i>Staff Development</i></p> <p>Describe _____</p> <p><i>Other</i></p> <p>Describe _____</p>
----------	--

Support for the IISME Programs

Objectives	<p>Students will</p> <ul style="list-style-type: none"> <li>- learn historic cryptographic algorithms</li> <li>- learn binary numbers and ASCII character conversion</li> <li>- learn steganography concepts</li> <li>- learn how modern cryptography works</li> <li>- use public-key cryptography to send/receive messages</li> <li>- learn the mathematics of public-key cryptography</li> <li>- learn modular arithmetic and advanced number theory</li> <li>- write JAVA code to encrypt data</li> </ul>
------------	--

Abstract (50 words or less)	<p>The workshops cover:</p> <ol style="list-style-type: none"> <li>1. History of cryptography (lecture/discussion)             <ol style="list-style-type: none"> <li>a. Simple ciphers of historic interest (activity)</li> </ol> </li> <li>2. Learn how computers represent and store data (lecture/activity)             <ol style="list-style-type: none"> <li>a. Learn how to hide messages in pictures</li> </ol> </li> <li>3. Learn how modern cryptography works (lecture)             <ol style="list-style-type: none"> <li>a. Use PKE to send and receive encoded messages (activity)</li> </ol> </li> <li>4. Mathematics of public-key cryptography (lecture)             <ol style="list-style-type: none"> <li>a. Learn how the RSA algorithm works (lecture)</li> </ol> </li> <li>5. Learn Boolean logic and the XOR algorithm (lecture/activity)             <ol style="list-style-type: none"> <li>a. Write a JAVA program for the XOR algorithm (lab)</li> </ol> </li> </ol>
-----------------------------	--

Handout Masters

<p><b>Describe how your ETP aligns with the National Board Standard stated in your proposal.</b></p>	<p>National Board Standard VII: (Learning Environment) Students will work collaboratively after school in a challenging, supportive, and interactive environment.</p>
<p>Resources Needed</p>	<p>Computers (PC), Internet accessibility via Web, Java SDK, PKE</p>
<p>Evaluation/Assessment Measures Used</p>	<p>Since this ETP is both extracurricular and enrichment for participants, no assessment will be given. However, students will demonstrate their knowledge by solving problems during each session and by writing a working Java program to encrypt data at the end of the workshop series.</p>
<p>Formatting specifications</p>	<p>PC <u>  X  </u> or Mac <u>      </u> (<b>Must be in Word or Text Format</b>) Software used. <u>  Power Point      </u></p>
<p>Mentor Signature and comments</p>	
<p>Administrator's Signature and comments</p>	

# Cryptography Workshop

*By Mr. Ferrante*

# Session 1

- Learn what cryptography is
- Learn simple techniques

# What is Cryptography?

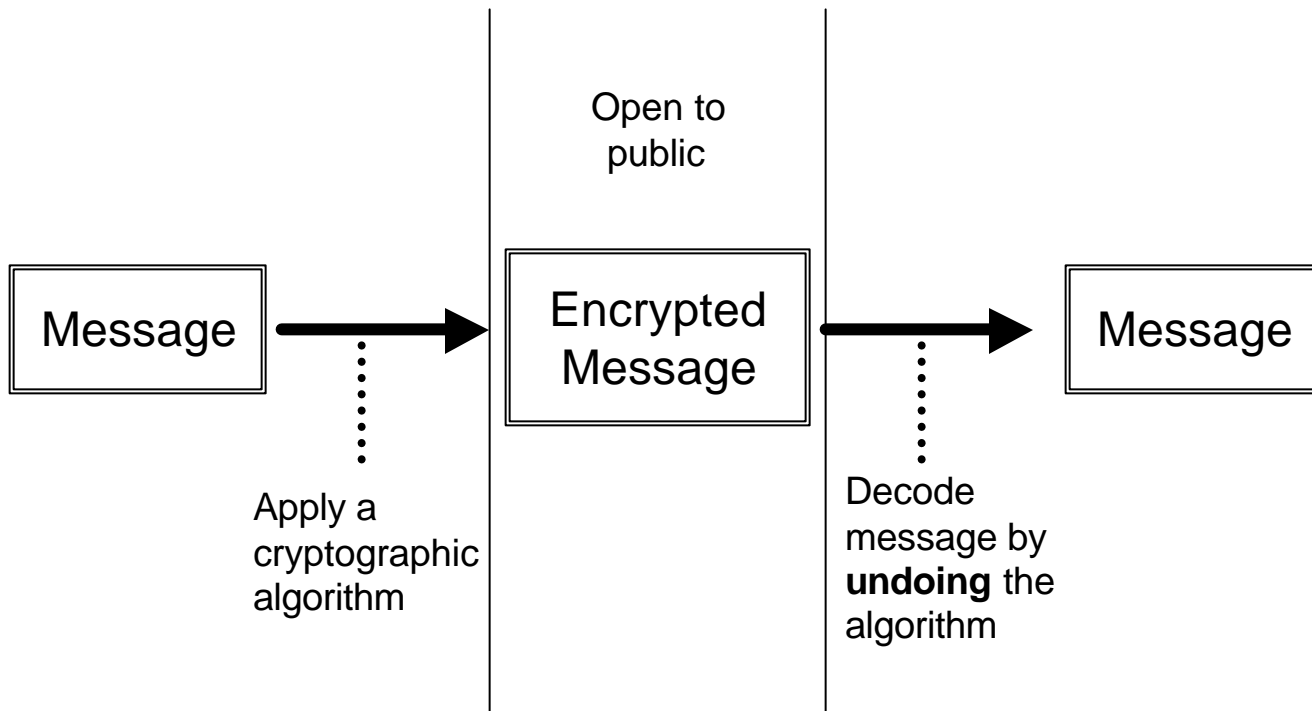
Cryptography is the “art and science of keeping messages secure.” [Schneier 1]

# Terminology

<b>Cryptography</b>	the art and science of keeping messages secure
<b>Cryptographer</b>	a person who uses cryptography to keep data secret
<b>Cryptanalyst</b>	a person who tries to break encoded data and retrieve its contents
<b>Cryptology</b>	the branch of mathematics dealing with cryptographic and cryptanalytic algorithms
<b>Cipher</b>	A cryptographic algorithm (the procedure used)

# How Cryptography Works

(Restricted algorithm)



# Challenge

(Restricted algorithm)

Break the encrypted secret!

**L OLNH LFH FUHDP**

## Algorithm

**Encrypt:** rotate alphabet 3 letters to the right

**Decrypt:** rotate alphabet 3 letters to the left

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  
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

**I LIKE ICE CREAM**

**L OLNH LFH FUHDP**



# Substitution Ciphers

..... What's a "Cipher?"

<b>Monoalphabetic / Simple</b>	Each letter replaced by another character
<b>Homophonic</b>	Each letter can map to one of several characters. Ex: A ↦ {8, 12, 57} B ↦ {128, 2, 5}
<b>Polygram</b>	Characters are encrypted in groups. Ex: "ABA" ↦ "XPO"
<b>Polyalphabetic</b>	Multiple simple substitution ciphers First letter encrypted by first key Second letter encrypted by second key ... Cycle through the keys

# Transposition Ciphers

- A transposition is a permutation, or “shuffling” of objects.
- Transpose ciphers keep the original message intact, but transpose its order.

## Example 1:

**M:** Hello there, my name is Bob.

**E:** H e l l o  
t h e r e  
m y n a m  
e i s B o  
b

5x5

HtmebehylenslraBoemo

# Transposition Ciphers

To break a simple transposition cipher, you need:

- To guess the dimensions of the original matrix
- Allowing spaces makes it harder (Why?)

Example 2:

M: Hello there, my name is Bob.

E: H e l l o t h e  
r e M y n a m e  
I s B o b



3x8

HrieeslmBlyoonbtahmee

# Challenge !

Break the ciphertext

E: Turtpoerritoleeeehodabdeksuasyr

M: ?

Hint : There are 3 spaces

## Solution

M: Treasure buried at the Spooky Old Tree  
(7x5 matrix)

# German ADFGVX Cipher

- Historical cipher used by the German army during World War I
- Uses a combination of simple substitution and transposition
- Read about it at <http://www.und.edu/org/crypto/crypto/lanaki.crypt.class/programs/adfgvx/grc.txt>  
Or search the Web for "German ADFGVX Cipher"

# Session 2

- Learn how computers store data
- Learn how to hide messages in pictures!

# To understand Cryptography...

- You need to know how computers store data

- To motivate the topic, we'll look at **Steganography**

Good stuff!

# Steganography

- Is the practice of protecting information by hiding its existence

## Examples

- Writing a letter that has a "decoy" message on one side and the real message written on the back using invisible ink
- Writing a message where the first letter of each word is the real message
- Hiding a message in a bitmap or sound file by replacing the least significant bit with the message bits.

# Steganography

Which image has the hidden message?



**Fact:** You can fit a 120 word paragraph in one of these images. (1 word = 5 characters)



..... Actual size!

# Prerequisite knowledge

- Know how to convert between base 10 and binary (base 2)
- Know what a least significant bit is
- Know how computers store characters (ASCII codes)
- Know how computers store pictures

# How to convert base 10 to binary (base 2)

- You use base 10 numbers every day
  - **Ex:  $325 + 4 = 329$**

Why is this system called “base 10” ?

- because the system uses 10 digits to represent numbers:  
0 1 2 3 4 5 6 7 8 9
  - **Ex: 578 is made up of these 10 digits**

And...

There's another reason...

# Base 10

- You can write any number as a **sum of powers of 10**

## Example 1:

$$\begin{aligned} 329 &= 300 + 20 + 9 \\ &= 3(100) + 2(10) + 9(1) \\ &= 3?10^2 + 2?10^1 + 9?10^0 \end{aligned}$$

## Example 2:

$$\begin{aligned} 5608 &= 5?10^3 + 6?10^2 + 0?10^1 + 8?10^0 \\ &= 5000 + 600 + 0 + 8 \end{aligned}$$

## Try it!

- Write each number as a sum of powers of 10
  - 75
  - 4286
  - 50913

$$75 = 7 \cdot 10^1 + 5 \cdot 10^0$$

$$4286 = 4 \cdot 10^3 + 2 \cdot 10^2 + 8 \cdot 10^1 + 6 \cdot 10^0$$

$$50913 = 5 \cdot 10^4 + 0 \cdot 10^3 + 9 \cdot 10^2 + 1 \cdot 10^1 + 3 \cdot 10^0$$

# Binary numbers (base 2)

- Works the same way as base 10, but uses 2 digits to represent numbers: 0 1 (that's why it's called base 2)
  - **Ex: 101** (in base 2)

- To convert binary numbers to base 10, write the number as a sum of powers of 2

$$\begin{aligned}101 &= 1?2^2 + 0?2^1 + 1?2^0 \\ &= 4 + 0 + 1 \\ &= 5 \text{ (base 10)}\end{aligned}$$

$$\begin{aligned}1101 &= 1?2^3 + 1?2^2 + 0?2^1 + 1?2^0 \\ &= 8 + 4 + 0 + 1 \\ &= 13 \text{ (base 10)}\end{aligned}$$

## Try it!

- Convert each binary number to base 10, then check your answer online at <http://www.onlineconversion.com/base.htm>

- a) 1100
- b) 1111
- c) 10010101

- a) 1100 = 12
- b) 1111 = 15
- c) 10010101 = 225

Would you like to practice converting from base 10 to base 2?  
(We didn't do that)

# Stuff to know about binary numbers

- Each digit of a binary number is called a bit
- A group of 8 bits is called a byte

## *Example:*

1100            4-bit number

10010101      8-bit number, or 1 byte

- When a file size is "4K" on your computer, it means the file is made up of

4K bytes    =    4000    bytes  
              =    8(4000) bits  
              =    32000    bits long!

# Least significant bit

- Same as least significant digit
- Is always the right-hand most digit

Example (base 10)

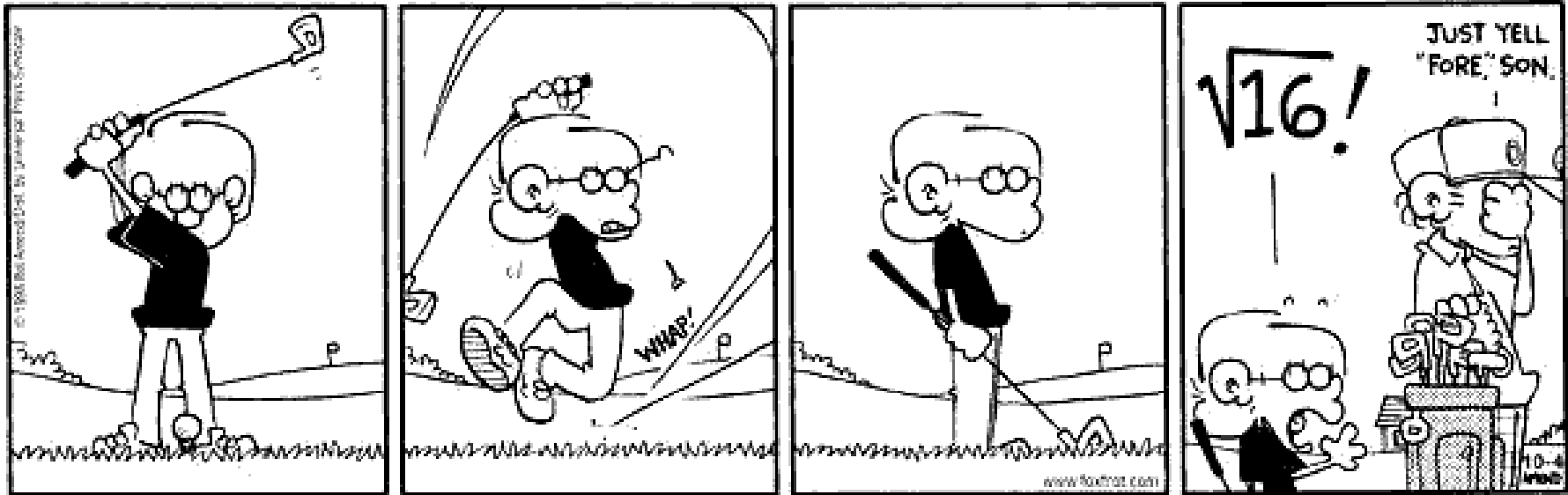
- 526 least significant digit is 6 (Why?)

Example (base 2)

- 101 least significant digit is 1 (Why?)

We're almost there...hang on!

==| Coffee break |==



# How computers store characters

- Computers do **everything** in binary (base 2)
- Therefore, to be useful to a computer, everything you see or use on your computer has to be converted into a number written in binary form
- Text characters are converted using ASCII codes
  - ASCII (American Standard Code for Information Interchange)
  - It's the standard everybody agreed upon for the English character set

Check out <http://www.asciitable.com>

## Question

- If characters are stored as ASCII codes (0-255), how many bits does it take to store a single character?

Hint: The largest character code 255. Convert 255 to a binary number and see how long it is

Let's try converting characters by hand !

# Character conversion

Example: Convert "Sam" into binary form

S	a	m
83	97	109
01010011	01100001	01101101

Therefore, "Sam" = 010100110110000101101101

How many bits / bytes are needed to store Sam?

Example: Convert back to characters

01001010	01101111	01100101	(base 2)
74	111	101	(base 10)
J	o	e	

# How computers store pictures

(We're only going to cover the stuff we need)

- Images are made up of pixels
  - Each pixel has a color : (red, blue, green)
  - Each color is stored as a binary number



# Converting colors to binary form

Pink =	red	green	blue
	255	135	200
	11111111	10000111	11001000

Therefore, Pink = 111111111000011111001000

Q: What is the least significant bit of the **red** component?  
**green** component?  
**blue** component?

Cool! Now we're armed and dangerous!  
(metaphorically speaking...of course)

## Back to our example

What's Steganography?

Which image has the hidden message?



Hidden message!

### How I hid the message

The message is hidden in the least significant bit of the Blue component of each pixel in the top row

## To recover the message...

- Read the blue component for each pixel in the top row
- Convert these values into binary form
- Copy down the least significant bit for each of them
- Once you have a collection of bits, break the collection into groups of 8 and find the ASCII value for each group of 8

**Challenge:** Hide a message in a picture and give it to me or a friend to decode.

**Notes:** Make sure you save the file as a .BMP, otherwise your data may get changed by automatic file compression such as GIF and JPEG compression.

# Guess what!

There are programs to do all this for you!

- Download *Steghide*
- Read "usage.txt"
- Use Steghide to extract the secret message hidden in `mr_f_msg.jpg` (Passphrase is "ferrante")

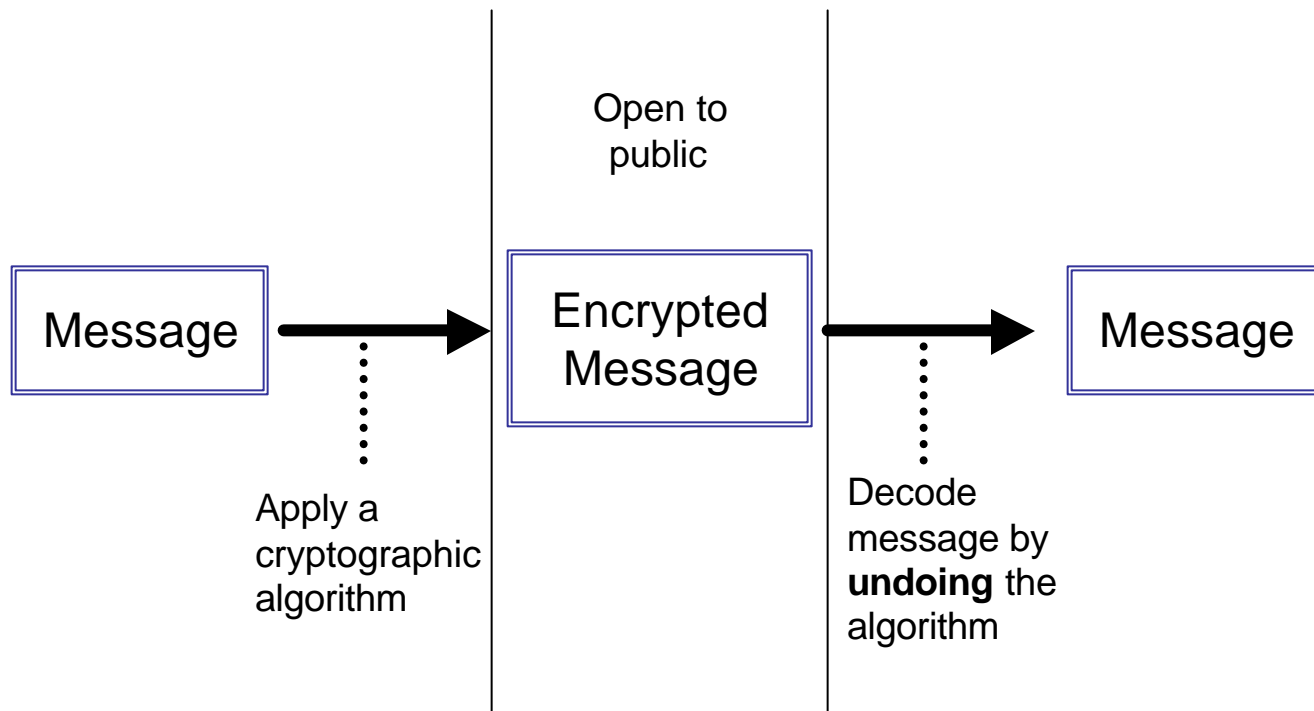
# Session 3

- Learn how modern cryptography works
- Send a message to a friend using public-key cryptography

# Review

## (Restricted algorithm)

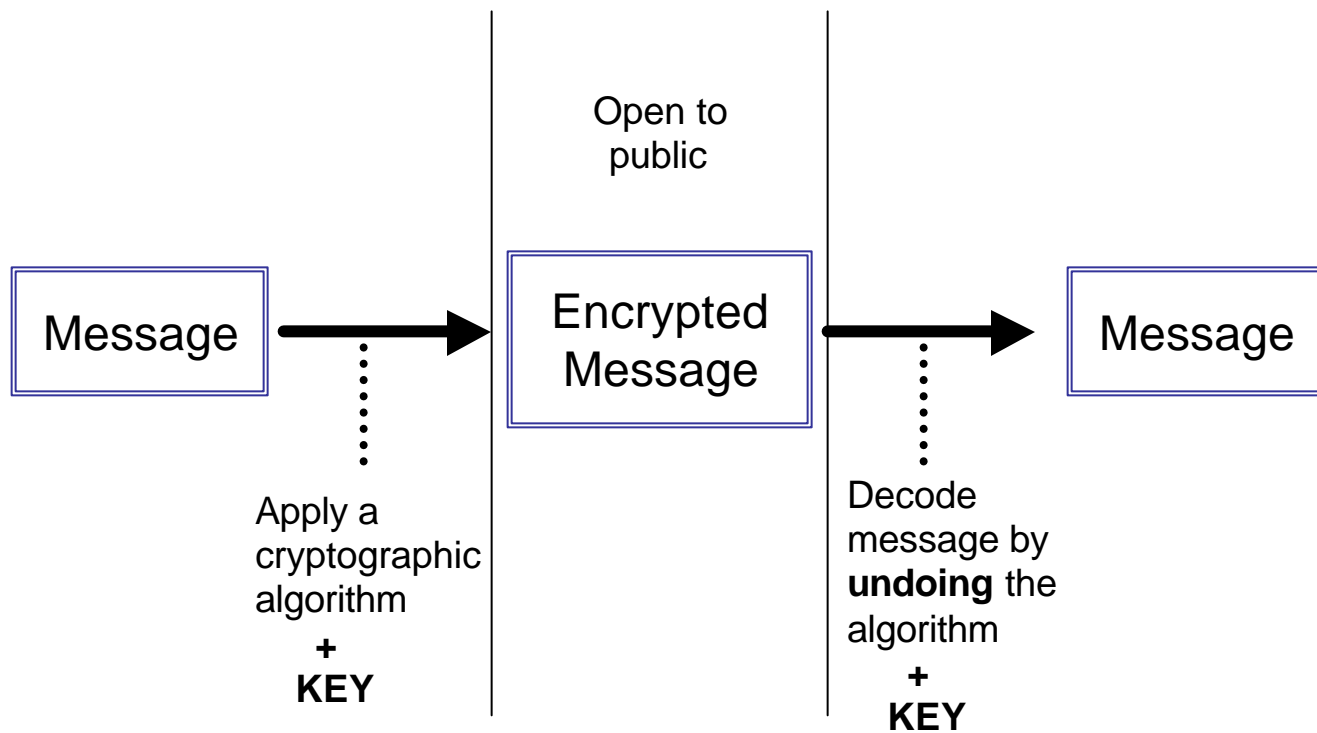
Q: When the security of encrypted data depends on secrecy of the algorithm used, what kind of algorithm is it? (name)



As it turns out, there are major problems with using restricted algorithms...  
What do you think some of them are?

# Modern Cryptography

Security in modern cryptography lies in the use of keys

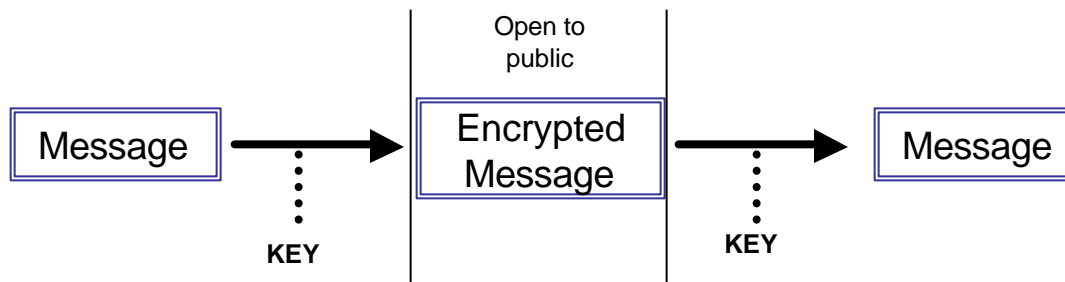


Why is this any better than using restricted algorithms?

# Classification of key-based algorithms

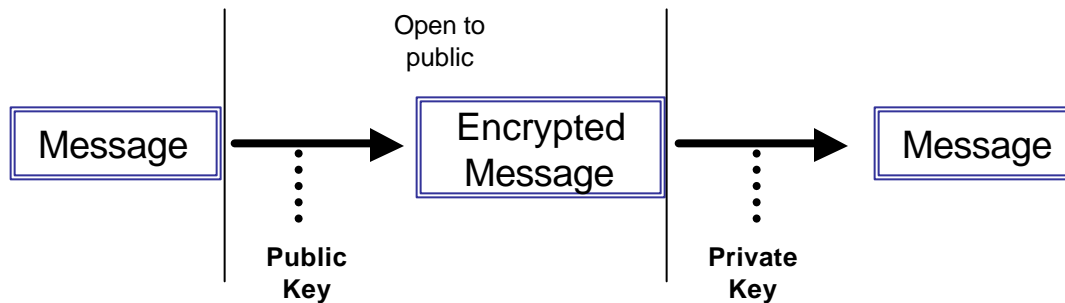
Symmetric

Same key used for encryption/decryption



Public-Key

Different key used for encryption/decryption



# Pros/cons of each type

## Symmetric



- Usually very fast and efficient
- Can handle large amounts of streaming data



- Key must be agreed upon first
- Keys generally need to be thrown away

## Public-Key



- Convenient for sender (browser knows keys)
- Very high, custom security



- Very slow (computation intensive)
- Message must be ? key size

# Modern cryptography combines technology

**Example 1:** Suppose you want to buy something online

1. Your computer obtains the public key for the website.
2. You enter your credit card # and the SSL code in your Web browser encrypts it with the public key, then sends it off.
3. Only the website has the private key. Your data is safe.

**Example 2:** Suppose you want to send and receive a large amount of private data (ex: bank statements)


1. Your computer obtains the public key for the website.
2. Your computer generates a private key, encrypts it with the website's public key, then sends it to the website.
3. Both parties now have a secret private key.
4. Data is then sent using symmetric algorithms such as RC4, DES, 3DES, or AES.

# Try it!


## Work with a partner

- Download PKE
- Practice sending and receiving encrypted messages

## Your steps

1. **File**  **New Key**, write down the 3 numbers
2. Give your public key and modulus to your partner

## Your partner's steps

3. **File**  **New Msg**, then type your message
4. **Encode** using the public key and modulus
5. **Save** the message and give it to your friend

## Your steps

6. **File**  **Open Msg**
7. **Decode** using your private public key and modulus

# Session 4

- Learn the mathematics used in public-key cryptography
- Learn the RSA algorithm

# Mathematics of public-key cryptography

If you really want to understand public-key cryptography, you need mathematics.

## Background needed

- Modular arithmetic, relative primes
- Euler's phi function
- GCD and Euler's theorem
- Modular inverses

# A new operator: MOD

Using MOD is just another way write a division problem with remainders

## Notation:

$a \equiv b \pmod{c}$       "a is congruent to b mod c"

## What it means:

b is the remainder when you divide a by c.

c is called the *modulus*.

## Examples:

$24 \equiv 0 \pmod{8}$

$24 \equiv 4 \pmod{5}$

$50 \equiv 1 \pmod{7}$

# Modular arithmetic

Modular arithmetic is the same as usual but in the end you only care about remainders

## Examples:

$$5 + 8 \pmod{3} = 1$$

$$5 + 8 \pmod{4} = 1$$

$$5 + 8 \pmod{7} = 6$$

-----

$$5 \cdot 8 \pmod{4} = 0$$

$$3^3 \pmod{5} = 2$$

# Modular arithmetic

It doesn't matter when or how often you take the modulus during a problem. As long as you take the modulus in the end, you'll get the correct answer.

## Example:

$$5 + 8 \pmod{3} = [5 \pmod{3} + 8 \pmod{3}] \pmod{3}$$

$$13 \pmod{3} = [ \quad 2 \quad + \quad 2 \quad ] \pmod{3}$$

$$1 \quad \quad = \quad \quad 1$$

# Relative Primes

A **prime number** is a number that has no factors other than 1 and itself

Which of these is prime?

- |       |       |
|-------|-------|
| a) 9  | d) 81 |
| b) 13 | e) 83 |
| c) 80 | f) 1  |

Two numbers are **relatively prime** if they have no common factors other than 1

Which pairs are relatively prime?

- |           |             |
|-----------|-------------|
| a) (6, 8) | d) (3, 100) |
| b) (4, 9) | e) (5, 32)  |
| c) (3, 3) | f) (1, 4)   |

# Euler's phi (?) function

also called Euler's **totient**

$\phi(n)$  is the number of positive integers less than  $n$  that are relatively prime to  $n$ .

**Find  $\phi(n)$  for these:**

- |       |         |
|-------|---------|
| a) 9  | d) 23   |
| b) 13 | e) 101  |
| c) 18 | f) 1021 |

**Special case:** For any prime number  $n$ ,  $\phi(n) = n - 1$

# GCD

- The greatest common divisor (GCD) of two numbers is their largest common factor.
- Written as  $\text{gcd}(p, q) = \dots$  or  $(p, q) = \dots$

**Find the GCD for these:**

- |                   |                    |
|-------------------|--------------------|
| a) $(6, 8) = 2$   | d) $(101, 56) = 1$ |
| b) $(4, 9) = 1$   | e) $(18, 36) = 18$ |
| c) $(20, 42) = 2$ | f) $(12, 56) = 4$  |

# Euler's theorem

- If  $\gcd(a, n) = 1$  then  $a^{\varphi(n)} \pmod{n} = 1$

# Modular Inverses

- Inverses are functions that undo each other
- Example **without** mod  
To find the inverse of 5 (under multiplication), ask:  $5x = 1$  ?  
Answer:  $1/5$ , because  $5(1/5) = 1$
- Example **with** mod  
To find the inverse of 5 (mod 7), ask:  $5x = 1 \pmod{7}$  ?  
Answer: 3, because  $5 \cdot 3 = 1 \pmod{7}$

There's another number that works... what is it?

Answer:  $3 + 7n$ , for any  $n > 0$ . Ex: 10

But since, it's mod 7, we don't count any of them.

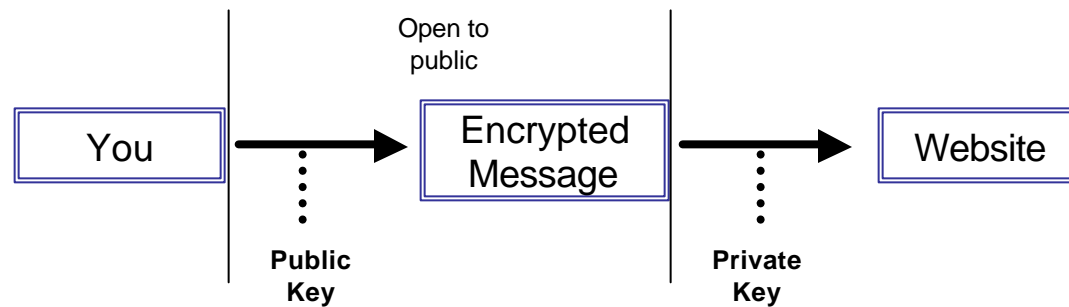
# RSA algorithm

- RSA is a public-key algorithm commonly used today
- Developed by Ron Rivest, Adi Shamir, and Leonard Adleman

## Review

Public-Key

**Different key used for encryption/decryption**



# RSA algorithm

## Operations needed

1. Create keys: private, public
2. Encrypt
3. Decrypt

## Create keys

- Generate two large primes  $p, q$ . (1024-bits or more)
- Compute  $n = p \cdot q$
- Compute  $c = \phi(n)$  note:  $c = (p-1)(q-1)$
- Choose a private key  $d = n$ , such that  $(d, c) = 1$
- Public key is the inverse of the private key (mod  $c$ )  
 $e = d^{-1} \pmod{c}$  or  $e \cdot d = 1 \pmod{c}$

# RSA algorithm

## Encrypt

- Break the message  $M$  into blocks  $m_i$  smaller than the modulus  $n$
- Encrypt each block using
$$s_i = m_i^e \pmod{n}$$

## Decrypt

- Decrypt each block using
$$m_i = s_i^d \pmod{n}$$

# RSA algorithm

## Why it works

- The keys are inverses. They undo each other.
- Since  $s = m^e \pmod n$   
 $m = s^d \pmod n$   
 $= (m^e)^d \pmod n$  substitution  
 $= m^{e \cdot d} \pmod n$   
 $= m^{k \cdot c + 1} \pmod n$  because  $e \cdot d = 1 \pmod c$   
 $= m^{k \cdot c} m^1 \pmod n$   
 $= (m^c)^k m^1 \pmod n$   
 $= 1 \cdot m^1 \pmod n$  Euler's theorem,  $c = \phi(n)$   
 $= m$  because  $m = n$

## Other stuff

- For more info, read *pkehow.txt* or search the Web for “RSA algorithm”
- The security of RSA lies in the keys and modulus
- If your keys are too small, the modulus  $n$  will be easy to factor and recover the private key (use 1024-bits or more)
- If you want a challenge, try breaking a message encrypted with PKE. It involves factoring and inverses ...  
(Give me a message and key file...I'll break it ✍)

# Session 5

- Learn Boolean logic
- Learn how the XOR algorithm works
- Write a simple XOR program in JAVA

# XOR Algorithm - the perfect model

- Is a symmetric algorithm
- Is simple and extremely fast
- Is unbreakably secure if
  - key is never reused
  - key length matches message length

(This is unreasonable and thus the perfect XOR model is not practical nor able to be implemented effectively. Nonetheless, it's fun for non-professionals like us to play around with.)

**How it works:** Each message bit is XOR'd with the next key bit

X-WHAT?

# Boolean logic

Three logical operations to know:

x (AND) y            means both x and y must be true  
x (OR) y            means x or y **or both** must be true  
x (XOR) y            means x or y, **but not both**, must be true

Binary examples:

1101 **AND** 0101 = 0101

1101 **OR** 0101 = 1101

1101 **XOR** 0101 = 1000

# XOR example

**Message:** Joe (in characters)

010010100110111101100101 (in bits)

**Key:** a52 (I made it up randomly)

011000010011010100110010

Message (XOR) Key

010010100110111101100101

011000010011010100110010

= 001010110101101001010111

00101011 01011010 01010111

43

90

87

+

Z

W

# Try it!

**Message:** Sue (in characters)

???????????????????????????????????? (in bits)

**Key:** 523

???????????????????????????????????? (in bits)

Message (XOR) Key

????????????????????????????????????

????????????????????????????????????

= ?????????????????????????????????????

?????????? ??????????? ???????????

???

?

# Try it!

**Message:** Sue (in characters)

010100110111010101100101 (in bits)

**Key:** 523

001101010011001000110011 (in bits)

Message (XOR) Key

010100110111010101100101

001101010011001000110011

= 011001100100011101010110

01100110 01000111 01010110

102 71 86

f G V

# Simple XOR JAVA program

How the program will work

1. User gives the program a single-character key and a text file to encrypt
2. Program XOR's each byte of the text file with the key (which is also 1 byte, because it's a character)
3. As the program reads and XOR's bytes, it writes them to a new encrypted file.

## Try it!

- Open CryptXOR.java
- Study the code
- Compile & run the program

**Challenge:** Modify the source code

# Programming Challenge

## History

- A professionally secure variant of XOR was created by Ron Rivest in 1987 for RSA Data Security Inc.
- The algorithm "RC4" was proprietary for seven years
- Then, someone anonymously leaked the code

## Challenge

- Search Yahoo for "Arcfour algorithm"
- Find the paper written by Kaukonen and Thayer
- Read the paper and study it
- Implement RC4 in Java

Slide 6 Challenge

Break the encrypted secret: **L OLNH LFH FUHDP**

Slide 6 Challenge

Break the encrypted secret: **L OLNH LFH FUHDP**

Slide 6 Challenge

Break the encrypted secret: **L OLNH LFH FUHDP**

Slide 6 Challenge

Break the encrypted secret: **L OLNH LFH FUHDP**

Slide 6 Challenge

Break the encrypted secret: **L OLNH LFH FUHDP**

Slide 6 Challenge

Break the encrypted secret: **L OLNH LFH FUHDP**

Slide 6 Challenge

Break the encrypted secret: **L OLNH LFH FUHDP**

Slide 6 Challenge

Break the encrypted secret: **L OLNH LFH FUHDP**

Slide 11 Challenge

Break the ciphertext: **Turtpoerritoleeeehodabdeksuasyr**

Slide 11 Challenge

Break the ciphertext: **Turtpoerritoleeeehodabdeksuasyr**

Slide 11 Challenge

Break the ciphertext: **Turtpoerritoleeeehodabdeksuasyr**

Slide 11 Challenge

Break the ciphertext: **Turtpoerritoleeeehodabdeksuasyr**

Slide 11 Challenge

Break the ciphertext: **Turtpoerritoleeeehodabdeksuasyr**

Slide 11 Challenge

Break the ciphertext: **Turtpoerritoleeeehodabdeksuasyr**

Slide 11 Challenge

Break the ciphertext: **Turtpoerritoleeeehodabdeksuasyr**

Slide 11 Challenge

Break the ciphertext: **Turtpoerritoleeeehodabdeksuasyr**

### Slide 33 Challenge

Open "happy\_face\_message.bmp" in Paint Shop Pro and zoom in to 10x or 15x. It also helps if you turn on the image grid and set it to 1 pixel. To do that, go to the menubar and choose **View  $\mathcal{Z}$  Grid**, then choose **View  $\mathcal{Z}$  Change Grid and Guide Properties**. Make sure the units are in Pixels and that both Horizontal and Vertical spacing are set to 1.

To read pixel color, select the Dropper tool and move your mouse over each pixel. Read the Blue component, convert it to binary, and look at the least significant bit. Copy it down and go to the next pixel. Keep copying bits until you have 8 of them. That's one character (letter) in the message. Now keep going. Every group of 8 bits is a character in the message. There are only 5 characters in the message (for obvious reasons, right? You don't want to decode like this all day!)

### Slide 33 Challenge

Open "happy\_face\_message.bmp" in Paint Shop Pro and zoom in to 10x or 15x. It also helps if you turn on the image grid and set it to 1 pixel. To do that, go to the menubar and choose **View  $\mathcal{Z}$  Grid**, then choose **View  $\mathcal{Z}$  Change Grid and Guide Properties**. Make sure the units are in Pixels and that both Horizontal and Vertical spacing are set to 1.

To read pixel color, select the Dropper tool and move your mouse over each pixel. Read the Blue component, convert it to binary, and look at the least significant bit. Copy it down and go to the next pixel. Keep copying bits until you have 8 of them. That's one character (letter) in the message. Now keep going. Every group of 8 bits is a character in the message. There are only 5 characters in the message (for obvious reasons, right? You don't want to decode like this all day!)

### Slide 33 Challenge

Open "happy\_face\_message.bmp" in Paint Shop Pro and zoom in to 10x or 15x. It also helps if you turn on the image grid and set it to 1 pixel. To do that, go to the menubar and choose **View  $\mathcal{Z}$  Grid**, then choose **View  $\mathcal{Z}$  Change Grid and Guide Properties**. Make sure the units are in Pixels and that both Horizontal and Vertical spacing are set to 1.

To read pixel color, select the Dropper tool and move your mouse over each pixel. Read the Blue component, convert it to binary, and look at the least significant bit. Copy it down and go to the next pixel. Keep copying bits until you have 8 of them. That's one character (letter) in the message. Now keep going. Every group of 8 bits is a character in the message. There are only 5 characters in the message (for obvious reasons, right? You don't want to decode like this all day!)

## Session 1 Resources

How to break the Caesar Cipher

<http://www.trincoll.edu/depts/cpsc/cryptography/caesar.html>

World War I German ADFGVX cipher

<http://www.und.edu/org/crypto/crypto/lanaki.crypt.class/programs/adfgvx/grc.txt>

or Search the Web for “German ADFGVX Cipher”

## Session 1 Resources

How to break the Caesar Cipher

<http://www.trincoll.edu/depts/cpsc/cryptography/caesar.html>

World War I German ADFGVX cipher

<http://www.und.edu/org/crypto/crypto/lanaki.crypt.class/programs/adfgvx/grc.txt>

or Search the Web for “German ADFGVX Cipher”

## Session 1 Resources

How to break the Caesar Cipher

<http://www.trincoll.edu/depts/cpsc/cryptography/caesar.html>

World War I German ADFGVX cipher

<http://www.und.edu/org/crypto/crypto/lanaki.crypt.class/programs/adfgvx/grc.txt>

or Search the Web for “German ADFGVX Cipher”

## Session 1 Resources

How to break the Caesar Cipher

<http://www.trincoll.edu/depts/cpsc/cryptography/caesar.html>

World War I German ADFGVX cipher

<http://www.und.edu/org/crypto/crypto/lanaki.crypt.class/programs/adfgvx/grc.txt>

or Search the Web for “German ADFGVX Cipher”

## Session 2 Resources

ASCII Table

<http://www.asciitable.com>

Base conversion

<http://www.onlineconversion.com/base.htm>

Steghide v.0.4.6 (windows)

<http://www.ecn.org/crypto/soft/stego.phtml>

## Session 2 Resources

ASCII Table

<http://www.asciitable.com>

Base conversion

<http://www.onlineconversion.com/base.htm>

Steghide v.0.4.6 (windows)

<http://www.ecn.org/crypto/soft/stego.phtml>

## Session 2 Resources

ASCII Table

<http://www.asciitable.com>

Base conversion

<http://www.onlineconversion.com/base.htm>

Steghide v.0.4.6 (windows)

<http://www.ecn.org/crypto/soft/stego.phtml>

## Session 2 Resources

ASCII Table

<http://www.asciitable.com>

Base conversion

<http://www.onlineconversion.com/base.htm>

Steghide v.0.4.6 (windows)

<http://www.ecn.org/crypto/soft/stego.phtml>

### Session 3 Resources

PKE Demo

<http://members.telocity.com/~acanright/pke/pkecomp.exe> (Download Software)

<http://members.telocity.com/~acanright/pke/> (Main page)

### Session 3 Resources

PKE Demo

<http://members.telocity.com/~acanright/pke/pkecomp.exe> (Download Software)

<http://members.telocity.com/~acanright/pke/> (Main page)

### Session 3 Resources

PKE Demo

<http://members.telocity.com/~acanright/pke/pkecomp.exe> (Download Software)

<http://members.telocity.com/~acanright/pke/> (Main page)

### Session 3 Resources

PKE Demo

<http://members.telocity.com/~acanright/pke/pkecomp.exe> (Download Software)

<http://members.telocity.com/~acanright/pke/> (Main page)

### Session 3 Resources

PKE Demo

<http://members.telocity.com/~acanright/pke/pkecomp.exe> (Download Software)

<http://members.telocity.com/~acanright/pke/> (Main page)

### Session 3 Resources

PKE Demo

<http://members.telocity.com/~acanright/pke/pkecomp.exe> (Download Software)

<http://members.telocity.com/~acanright/pke/> (Main page)

