Cryptography

A basic overview of the cryptographic techniques used in modern network communications

Cryptography – Objectives

- Recognize information risks that can be reduced through the use of cryptography.
- Define encryption.
- Identify the features and benefits of symmetric cryptography.
- Identify the features and benefits of asymmetric cryptography.
- Discuss the potential vulnerabilities of a cryptographic algorithm.

Cryptography– Objectives (Con't)

- List some common cryptographic algorithms and where they would apply.
- Assist a client in choosing appropriate cryptographic solutions.
- Explain the problems solved by digital signatures and digital certificates.

Encryption: What is it?

Encryption takes meaningful data and causes it to appear random to everyone EXCEPT the intended recipient.

Encryption: What is it?

Encryption is the science of protecting information from unauthorized viewing. The word "encryption" is derived from the Greek word "kryptos" meaning, "hidden". Cryptography is the science of hiding communication from others. Until the 1970's, this science of hidden communication was the exclusive purview of the government. However, with the rapid growth of corporate (and personal) computing, cryptography became an essential topic for non-governmental entities to study.

The unencrypted data is called "plain text". Encrypted data is sometimes referred to as ciphered text, or simply encrypted data. The root of the word cipher comes from the Arabic word sirf. Sirf is Arabic for zero, but to Europeans in middle ages it meant, "unclear". Ciphered text is unclear to, or hidden from, everyone except those who possess the key. Furthermore, for reasons that will be addressed later, it is important that the encrypted data not only is unintelligible to any observer who does not have a key – the encrypted data must have no discernable pattern. The ciphered text MUST appear random.

Encryption: Who Needs It?

- **Governments** (Military, Spies and Diplomats)
- **Banks** (Money Transfer, Credit Card Verification, Consumer Info)
- **Corporations** (B2B E-commerce, Proprietary Plans, Payroll)
- **Consumers** (B2C E-commerce, Confidential Personal Data)
- **Subversives** (Good Guys & Bad Guys)

Caesar's Cipher (Early Encryption)

Julius Caesar was able to communicate battlefield commands securely by using a simple substitution code. In one version, Greek letters were substituted for the Latin letters in Caesars' message.

Key: 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
α β χ δ ε φ γ η ι φ κ λ μ ν ο π ρ σ τ υ ϖ ω ξ ψ ζ

"**A secret message**" becomes "α τεχσευ μεττα γε"

Caesar's Cipher

Archeologists have shown that codes were used as early as Pharonic Egypt. However, it was not until Caesar wrote the *Gallic Wars* that we have our first documented explanation of ciphers. These ciphers were instrumental to Caesar's ability to send commands to his troops across large distance without fear that his commands had been intercepted or compromised.

While we often refer to "Caesar's Cipher" there were several "Caesar's Ciphers". In the above slide, Caesar substituted his Latin letters with Greek letters. Another variation uses only Latin letters. If this cipher were implemented in English then each letter would be assigned a number based on its position in the alphabet (i.e. A=1, B=2, C=3, etc.) The key is + 3. To encode a plain text message add three to each letter. For example "This is a test" is translated to 20 5 9 19 9 19 1 20 5 19 20 adding 3 to each number results in 23 8 12 22 12 22 4 23 8 22 23. When the numbers are replaced by letters the result is, "Wklv lv d whvw". This is also referred to as a substitution cipher.

Cracking Substitution Ciphers

"Gentlemen don't read each other's mail"

- Henry Stimson, U.S. Secretary of State (1929)

- **Before World War I, efforts to produce systematic methods to break ciphers were sporadic.**

- **Just before World War II:**
  - **Scientific principles (frequency analysis)**
  - **New technologies (computers)**
  - **The will to devote resources (Japanese & German codes)**

Cracking Substitution Ciphers

Although no one knows who made the first systematic attempts to decipher encrypted text – an Arab scientist wrote the first documented paper in the 1800's. However, until the 20th century cryptography and cryptanalysis were, at best, obscure sciences.

As World War II approached, Germany and Japan made significant gains in encryption science. The German "Enigma", and the Japanese "Purple", encryption machines were major advances. However, as often happens, technology pushes technology.

Cryptologists began developing sophisticated techniques to analyze the frequency with which letters, vowels and letter combinations appeared in various languages. The advent of tabulating and computing devices, combined with powerful frequency analysis techniques brought code breaking to new heights.

Language Patterns

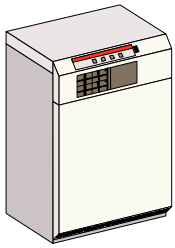
- **Wheel Of Fortune effect.**
- **All languages have discernable patterns.**
- **Cryptologists have developed a sophisticated technique, called *frequency analysis,* to identify and categorize these patterns.**
- **These patterns can be discerned even if the symbols are changed.**

Language Patterns

The reason that games like “hangman” and “Wheel Of Fortune” work is because all of us know intuitively that the English language follows certain patterns. Even if certain letters are removed from the “Wheel Of Fortune” graphic shown above, we still know it says, “Bonus Bucks Sweepstakes”. Cryptologists have developed an entire science around this phenomenon called frequency analysis.

Brute Force Techniques

- **With simple substitution (in English) there are 4×10^{26} keys combinations.**
- **Hundreds of Cryptographers working full time would take thousands of years to break a code by blindly trying keys.**



- **Modern computers are so fast that they can break these simple codes in minutes (or even seconds).**

Brute force techniques

Although it might take a human being a prohibitively long period of time to break a code, a computer works so fast that simply trying various keys is not out of the question. The combination of brute force using high-speed digital computers and sophisticated frequency analysis have made substitution ciphers (and related techniques) moot.

Key Compromise

- **Time honored technique of code breaking.**
- **Key compromise historically from 3 sources:**
  - **Captured enemy agents**
  - **Spies**
  - **Traitors**
- **Any lessons for the business world today?**

Key compromise

Why crack the code when you can steal the key? Compromising the key could be accomplished with violence (threatening /torturing some poor soul who has the key), stealth (using a spy to steal the key) or bribery (the Soviets used John Walker to compromised many American keys in this way). The same techniques are available for those unscrupulous individuals seeking to compromise confidential corporate information today.

Symmetric Digital Encryption

"It is only the existence of redundancy in the original message that makes a solution possible." – Claude Shannon

- Frequency analysis is based on redundancy (patterns) in the original message.
- Symmetric digital encryption greatly reduces the redundancy in the encrypted message (i.e. the encrypted message looks more random).
- Digital encryption can be implemented quickly with inexpensive hardware.

Symmetric Digital Encryption

Claude Shannon (1916-2001) laid the cornerstone for the field of digital communications. He provided much of the language of we use today to describe digital communications. While working at Bell Laboratories, Shannon developed much of the mathematical underpinnings of modern cryptography. (In addition to his contributions to cryptography, Shannon also discovered the relationship between noise and information carrying capacity of channel and was the first to define a binary digit (BIT) as a fundamental unit of information)

Shannon was able to show that the *Achilles heel* of cryptography is redundancy – patterns of the original message that can be discerned even in the ciphered text. Much of the redundancy can be removed by converting letters to digital numbers (binary in particular).

Binary

8 4 2 1	8 4 2 1	8 4 2 1
0 = 0 0 0 0	5 = 0 1 0 1	10 = 1 0 1 0
1 = 0 0 0 1	6 = 0 1 1 0	11 = 1 0 1 1
2 = 0 0 1 0	7 = 0 1 1 1	12 = 1 1 0 0
3 = 0 0 1 1	8 = 1 0 0 0	13 = 1 1 0 1
4 = 0 1 0 0	9 = 1 0 0 1	14 = 1 1 1 0
		15 = 1 1 1 1

Bit = Binary Digit

Byte = 8 bits Nibble = 4 bits

Binary

The numbering system we are most accustomed to is decimal (base 10). In decimal, each column has 10 times the weight of the column before it. A four-digit decimal number has four columns, each weighted in powers of 10: a thousands column, a hundreds column, a tens column and a ones column. The number 2437 is two thousands, four hundreds, three tens, and seven ones.

In binary (base 2) each column has twice the weight of the column before it. A four-digit binary number also has four columns but they are weighted in powers of two: an eights column, a fours column, a twos column, and a ones column. The binary number 1001 has an eight, no four, no two and a one. Add the columns together and the result is nine (in decimal.)

ASCII, Pixels, and Other Symbols

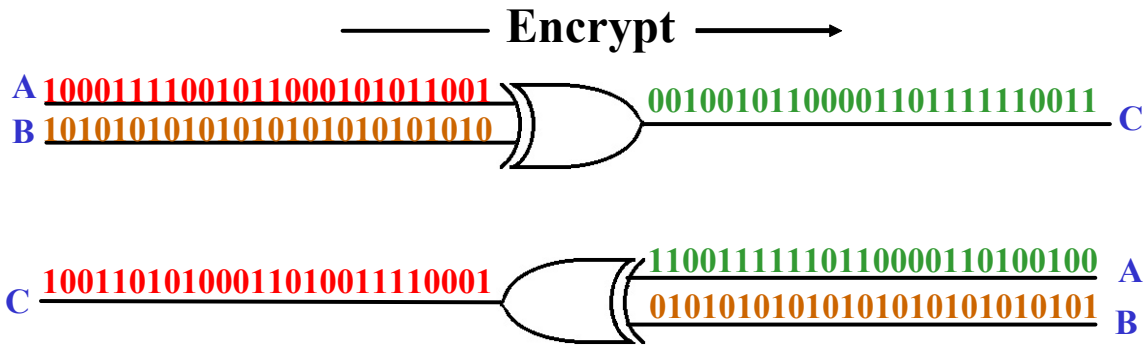
- **Binary is just a number system.**
- **Binary numbers can represent:**
  - **Letters: ASCII code (A = 01000001)**
  - **Graphics: Pixels (color/intensity)**
  - **Anything**
- **Binary is popular because today's computers natively speak binary.**
- **Binary symbols are easy to encrypt.**

ASCII, Pixels, and Other Symbols

A modern computer is a complex collection of millions of transistors. Each transistor has been optimized to act like a switch. Switches have two states – on and off. This makes binary the perfect numbering system for computers. However, not everything we want to manipulate is a number. There are many systems that assign numbers to non-numeric quantities like letters, colors, sounds, etc.

One of the most common binary codes is called ASCII (pronounced "AS KEY"). ASCII stands for **A**merican **S**tandard **C**ode for **I**nformation **I**nterchange. ASCII allows letters to be represented by binary numbers. All information that is entered, stored, manipulated, or displayed by a computer is represented internally by binary patterns.

XOR – Digital Stream Encryption



XOR Truth Table

A	B	C
0	0	0
1	0	1
0	1	1
1	1	0

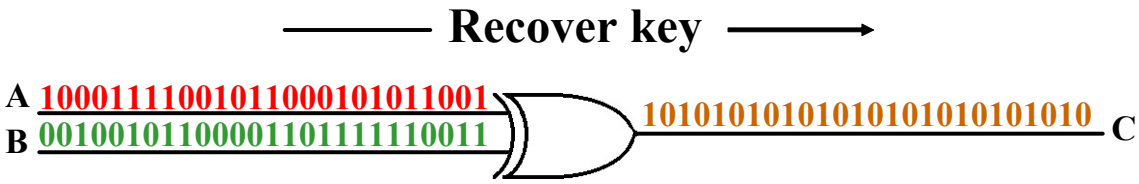
Brown = Key
Red = Plain Text (unencrypted)
Green = Encrypted Text

XOR – Digital Stream Encryption

The exclusive-or gate is at the heart of much digital encryption. What is an exclusive-or gate? An XOR gate is a function of Boolean algebra that follows these rules: if the two inputs (A&B) are the same (match) then the output is 0. However, if the two inputs (A&B) are different (do not match) then the output is 1.

The key shown in this illustration is a simple 10 pattern. This simple "2-bit" key would NEVER be used in a real crypto system. Although larger keys are used in real applications, this example clearly illustrates how easily, and effectively, XOR encryption works.

Cracking Digital Stream Encryption (1)



- **Digital Stream Encryption does a great job of removing redundancy if implemented as a *one time pad* (more on next slide).**
- **However, if a black hat (bad guy) can get a copy of the plain text and the encrypted text then *the key is easily recovered*.**

Cracking Digital Stream Encryption

Although digital stream encryption (based on XOR functions) is very effective in making a redundant data stream (plain text) look random, it does have some vulnerabilities. Those who are trying to compromise security are commonly referred to as "black hats". Black hats find two vulnerabilities in digital stream encryption.

First, if a relatively short key is used on a relatively long document then the key itself becomes a redundancy that MAY be able to be recovered from the ciphered text. This is difficult, but not impossible. An easier attack is for black hat to get a hold of the plain text message and the cipher text. It is then a simple matter for black hat to recover the key.

Cracking Digital Stream Encryption (2)

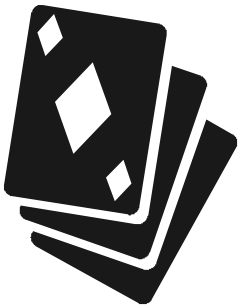
- **Redundancies can come from the plain text *or the key*.**
- ***One time pad* has no redundancy if it follows these rules:**
  - **The key must be truly random. Black hats can exploit any pattern reflected in the key.**
  - **The key must be the same size as the data. If the key is smaller than the data, the key must be repeated. This is a pattern that can be exploited code breakers.**
  - **The key must be used only once. Repetitive use of the key is also a pattern that can be exploited.**
- ***One time pads* are secure but difficult to administer because of the key length and its single use.**
- **Example: RC4**
  - **Very large pseudo random keys with 2^{1700} key combinations.**

Cracking Digital Stream Encryption

Redundancies from the key can be a present a serious weakness in the encryption algorithm to black hat. This vulnerability can be eliminated if the key is truly random and never repeated. A key that is the same length as the plain text and only used once is called a one-time pad. One-time pads are difficult to administer. Distributing huge keys that will be used only once is challenging - even for governments. However, all cryptologists strive to make their algorithms behave like a one time pad.

The Solution – Block Ciphers

- **Block Ciphers use a short, reusable key.**
- **Operate on one block of data at a time.**
- **Use several techniques to make a relatively short key behave like a *one time pad*.**



- **Like the "Three Card Monte" game. If enough moves are made, quickly enough, then the onlookers cannot follow the cards.**

The Solution – Block Ciphers

Remember, ciphers can be broken if the cryptographer can discern ANY pattern in the cipher text. One-time pads (if properly created) do not leave any pattern. However, one-time pads are difficult to manage. The goal of block ciphers is to use a small (manageable) key while removing all redundancy from the key.

Block ciphers use a process called *confusion and diffusion* to hide any pattern created by the repeating key. In 1949 Claude Shannon described these techniques. Confusion is accomplished by series of exclusive-or gates. Diffusion spreads the redundancy of the plain text and the key over the entire cipher text making it more difficult for a cryptologist to find them. Block ciphers rearrange the bits of both the key and the plain text in a cyclic fashion. Cyclic rearrangement means that this rearrangement of bits takes place many times for the same block of data.

Data Encryption Standard

- **30 year old algorithm.**
- **56-bit key length (7.2 x 10^{16} keys)**
- **Still a significant embedded base.**
- **Can be cracked in less than 24 hours with commercially available computers.**
- **US encryption standard (NIST) for past 20+ years.**

NIST: National Institute of Standards & Technology

Data Encryption Standard (DES)

DES was created as a joint effort between IBM and the NSA. The NSA (National Security Agency) is the ultra top-secret agency responsible for, among other things, all things cryptographic. The NSA was notoriously paranoid – even about its own existence. Many thought it was unlikely that the NSA would cooperate in creating a cryptographic algorithm with out creating a hidden "back door" allowing the government easy access to the user's encrypted data. Although never substantiated, rumors of this back door persist.

DES was adopted as the federal standard for encrypting unclassified documents in November 1976 and its algorithm, derived from IBM's "Lucifer" cipher, was published the following January. With official government backing, DES quickly became widely deployed. But over time, shortcut attacks were discovered that could significantly decrease the key cracking time. As a result, NIST abandoned their official endorsement of DES in 1997 in search of a replacement. The Advanced Encryption Standard (AES) officially succeeded DES in October 2000. Yet, DES was extensively used wherever secrets needed to be protected making it still the most widely used encryption algorithm to date.. Many vendors today offer a 3-pass DES option called *Triple-DES* which effectively doubles the cracking difficulty

DES is a block cipher, buts its reliance on a 56-bit keys has made it vulnerable to attacks from amateurs and professionals alike. Today, using DES to protect your sensitive data makes as much sense as using a sign that says, "do not steal" to protect your money.

DES details can be found at: http://home.delfi.ee/~sateks/des-how-to.html

Other Popular Block Ciphers

- **ADVANCED ENCRYPTION STANDARD (AES)**
  - **The new NIST standard**
  - **Key lengths: 128, 192 or 256 bits.**
- **Triple DES**
  - **Successor to DES**
  - **Very slow**
- **Still others block ciphers listed from fastest to slowest:**
  - **FEAL-8, Khufu, NewDES, MDC, Blowfish, RC4, GOST, SAFER, REDOC III, IDEA, 3-Way.**

    Most of these have options which choose between speed and security. This listing assumes that the fastest (least secure) options have been chosen.

Other Popular Block Ciphers

In all cryptographic algorithms, there is a tension between speed and security. Users should always choose the longest key length that can be reasonably (and legally) used. Weak encryption can be easily broken. But even more troubling, using weak encryption can lead to a false sense of security – perhaps making the user more lax than if no encryption was used.

The Bigger the Key …

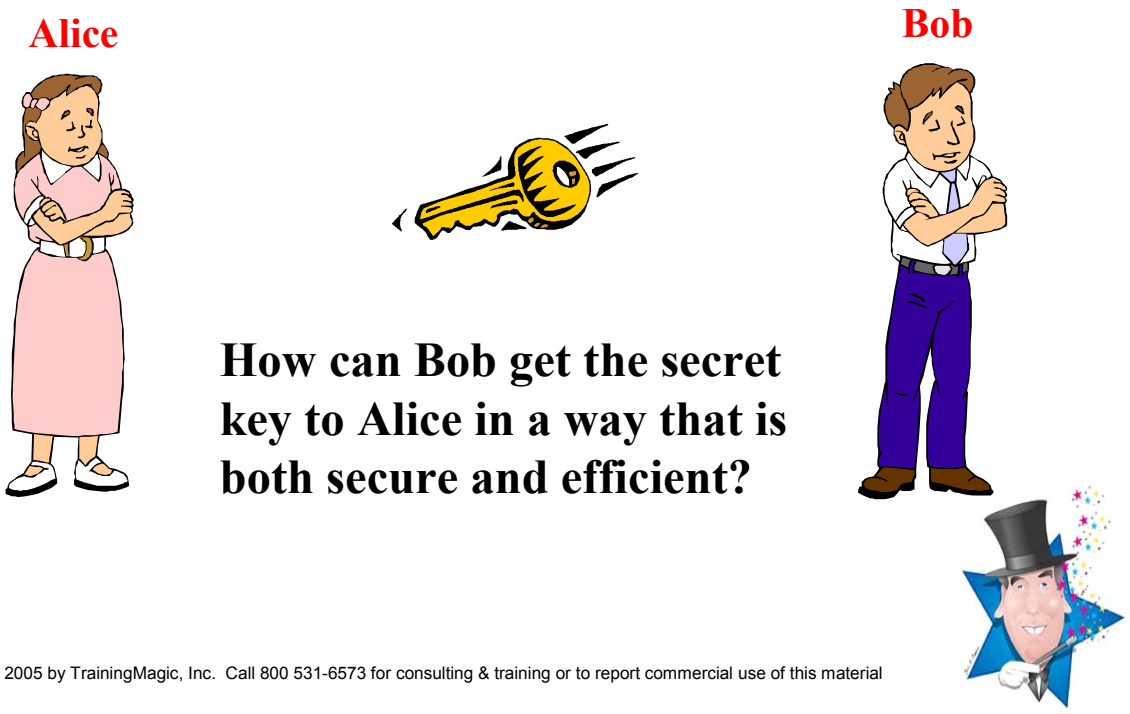
"It's the key stupid" (with apologies to James Carville)

- **Much debate about the security of various encryption algorithms.**
- **Don't relay on "secret" algorithms. Many algorithms have been found to be flawed.**
- **Flaws are discovered in the marketplace.**
- **With established (tested) algorithms, security lies in key length and randomness.**
- **Larger random key = Better security.**

The Bigger the Key …

James Carville is credited with creating the successful political campaign theme that helped Governor Clinton become President Clinton – "It's the economy stupid". Much has been written about the relative value of various encryption algorithms. The research done regarding vulnerabilities of encryption algorithms is important. Security professions should keep up-to-date with identified algorithm weaknesses. Two good sites to check on the latest research are: http://csrc.nist.gov/ and http://www.counterpane.com/. However, absent any proven vulnerabilities, the length of the key and the degree of randomness determines the relative security of an encryption algorithm. Therefore, assuming random key generation, users should probably choose the algorithm that supports the largest key.

The Key Distribution Problem



The Key Distribution Problem

Given a sampling of encrypted data, enough time, and enough computing power, a determined *Black Hat* will eventually find the algorithm and key. By regularly changing their encryption keys, Alice and Bob can confuse the attacker. But recall that in symmetric cryptography, Alice and Bob must use the *same* key. If Bob wants to change the key, how does he let Alice know what the new key value is?

How to get a key from Bob to Alice is a serious problem. The method chosen to transport the key must safe from compromise by black hat. If the method chosen to transport the key is safe from black hat then why bother with encryption at all? Any method that could be used to safely move a key could also be used send plain text data.

Solution: Asymmetric Encryption

- **Use 2 Keys:**
  - Key 1 encrypts only (can not decrypt data).
  - Key 2 decrypts information encrypted with key 1.
- **Most asymmetric encryption methods based the properties of modular mathematics.**
- **Concept generally credited to Whitfield Diffie and Martin Hellman.**
- **Examples: Diffie-Hellman, RSA, PGP**

Solution: Asymmetric Encryption

Before the mid 1970's, all cryptography was symmetric – the same key used to encrypt and decrypt the plain text. In 1976, Whitfield Diffie and Martin Hellman published a new approach. Use two keys: one key that only encrypts and another key that only decrypts. This is an important point – the key used for encryption CANNOT decrypt the data. Using the plain text and the encryption key, the decryption key CANNOT be found. This amazing development was made possible through the use some "math tricks" unique to modular mathematics.

Although some claim that the British Secret Service developed asymmetric cryptography before 1976, nothing was ever published. Diffie and Hellman are generally credited with first developing public key cryptography. For more information on how Asymmetric Encryption was developed read the great book *Crypto* by Steven Levy (published by Viking Press). This book is both fun and informative.

Modular Math

- **Modular math is clock math.**
  - The modulus is the number of places on the clock face (with the position traditionally held by 12:00 considered as 0).
  - Mod 12 has 12 places, mod 8 has 8 places, etc

7 mod 12

3^3 mod 12

9^7 mod 12

Modular Math

At first glance, Modular Mathematics may seem strange and unfamiliar. However, we use modular math every day. Clocks are a great example.

If it is currently 2:00 PM, and we agree to meet in 28 hours, we will meet at 6:00 PM – not at 30 o'clock. To calculate 26 hours after 2 o'clock we begin at 2, adding 10 brings us to 12:00. This leaves us with 18 hours (28-10) to add. Adding 12 hours brings us back around the clock face again. This leaves us with 6 hours (18-12) to add. Adding 6 hours brings us to 6 o'clock.

One important difference between clock math and modular math is the role of zero. Analog clocks do not have a zero on their face. The number 12 occupies that position.

Modulo is yet another word used to signify a number base. Computer systems use a modulo-2 method of counting, frequently called binary or base-2. There are only two symbols in its counting vocabulary, 0 and 1. Most people use a modulo-10 or decimal method to count, convenient given our number of fingers! If you consider modulo-10 math (base-10 or decimal), we use ten unique symbols, 0 thru 9, to represent quantities. When the quantity of something exceeds 9, we use another column to count higher (to 10). But what if we didn't have additional "columns"? We'd count a series as 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 0, 1, 2, 3, and so on. Begins to sound rather like clock math…

One Way Functions (The Trap Door)

- **There are a number of shortcuts that can be used in modular math.**
  - These shortcuts appear to only be in the positive direction (moving across the face clockwise).
  - This can make encryption very fast(milliseconds) and decryption very slow (weeks or years).
  - Ex. If the modulus is a prime number and the exponent is one less than the modulus then the result is 1 (i.e. 176^{10} mod 11 = 1).
- **Functions like this are referred to as trap doors (easy to get in – hard to get out).**

One Way Functions (The Trap Door)

Mathematical it is relatively easy to multiply two large prime numbers – whatever their modulus. However, if I tell you that the result of some function is 1 mod 432 can you tell me what the original number was? This is very difficult (probably impossible – but experience has taught us NEVER to say impossible in cryptography).

Members of the mathematics community have studied this difficult problem over many centuries. Their observation: It is easy to multiply two large prime numbers together. It is far more difficult to factor that number (find the two prime factors that were multiplied togather to create that large number product.)

The reason that we can not factor this result (find the two original prime numbers) is because the only known method of finding the two prime factors of a large number is to check all the possibilities one by one, which isn't practical because of the unthinkably large number of prime numbers.

This is the real secret of Public Key encryption. For example, RSA public key algorithm is based on the well-known hard problem of factoring large numbers into its prime factors.

Multiplicative Inverse - The Short Cut (1)

Question: ***If going backwards is so difficult then how can the message be decrypted?***

Answer: **Pick another number to multiply with the encrypted number that will return us to the original (unencrypted) number.**

(Remember that there are short cuts for multiplication but not for division)

The multiplicative inverse is that number, multiplied by another number, whose product is 1. for example, in mod 20 arithmetic, the multiplicative inverse of 3 is 7. Let's look at why: (7*3=21)/20 leaves a remainder of 1 and (7*3=21/20) also leaves a remainder of 1.

Multiplicative Inverse - The Short Cut (2)

- **In modular math, ½ revolution and 500 ½ revolutions look exactly the same.**
  - Begin with 19 mod 55, and raise that to the 3rd power (19^3) the result is 6859 (39 mod 55).
  - 3 would be the public key.
  - The secret key is 27.
  - The result (39 mod 55) is raised to the 27th power resulting in 3.79351387778256192 E10 (19 mod 55) right back where we started.
- **The public key is multiplied with the clear data to encrypt. The private key is multiplied with the encrypted data to decrypt.**

Multiplicative Inverse

This is the basis of a family of magic tricks popular with grade school children. Example would be, "Think of a number. Multiply it by 2. Add 16. Subtract 4. Multiply by 1/2. Subtract your original number. The result is 6."

In the case of modular math I can create two numbers. The first number will be used to multiply by my plain text number to produce an encrypted (obscured) number. The second number will be used to multiply by my encrypted number to bring it back to its original value.

The great theoretical weakness of public key cryptography (asymmetric cryptography) is how to choose the key pair. Only certain number pairs will work as key pairs. These numbers have a distinct mathematical relationship. If black hat can figure out the mathematical relationship between the two keys, then the code is compromised. The trap door mentioned earlier involving the well-known hard problem of factoring large numbers into its prime factors is used to hide the relationship between the two keys from black hat.

Key Pair

Encryption Key

Decryption Key

- **Together, the public and private keys are called the key pair.**
- **A shortcut used by RSA to create the key pair uses two prime numbers.**
- **The product of these prime numbers is used as the modulus for encryption.**

Key Pair

The most popular implementation of public key is RSA. Euler's theorem is at the heart of RSA. Here it is in a nutshell:

> [$m^{(p-1)(q-1)}$ mod n =1 where p and q are prime, m and n are relatively prime (relative prime numbers share no common factors other than 1), and n is the product of p and q.]
>
> Notice that result of raising M to the product of (P-1) and (Q-1) in Mod N is 1. That means that all of this complicated math will get us right back to our original number. The remaining math in RSA is designed to enable us to usefully separate the functions of (P-1) & (Q-1) while hiding their values from black hat.

For those not repulsed by math, two papers available on line will provide more RSA details: http://rr.sans.org/encryption/diffie.php and http://www.strongsec.com/zhw/KSy_RSA.pdf.

Additionally, for those who want to understand the math of RSA and other public key systems, one of the best books available is C*ryptography Decrypted* by Doris Baker and H.X. Mel, published by Addison-Wesley. The authors make the math of public key cryptography comprehensible to anyone with a high school and a willingness to read.

Cracking Public Key Encryption

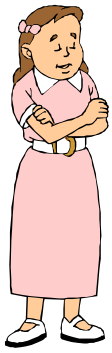
- **Usual key recovery techniques.**
- **Public Key Cryptography uses very large prime numbers.**
- **Easy to multiply – hard to factor.**
- **Is slow when large primes are used – is weak when small primes are used.**
- **Quantum computers (within 10 years) will be able to factor these numbers quickly.**

Cracking Public Key Encryption

Public key cryptography (asymmetric cryptography) is not without its vulnerabilities. As with all forms of cryptography keys can be recovered through both human vulnerabilities (bribery, extortion, etc.) and computer vulnerabilities (computer memory, hard drive, etc.)

Current vulnerabilities that are unique to public key cryptography are as much a factor of human behavior as a factor of the technology. Public key algorithms are only secure when the keys are based on very large prime numbers. However, public key algorithms can be very slow when very large keys are used. Given a choice between fast performance and security many users will choose fast performance.

Private Key Distribution via Public Key Encryption



1. **Alice creates a public key and sends it to Bob.**
2. **Blackhat may intercept Alice's public key, but it does not matter. Only Alice has the corresponding private key.**
6. **Alice decrypts the DES key with her private key.**
7. **All communication between Alice & Bob is now encrypted with DES.**

3. **Bob creates a secret key to be used in symmetric algorithm like DES.**
4. **Bob uses Alice's public key to encrypt his DES symmetric key.**
5. **Bob sends the encrypted DES key to Alice.**

Most secure e-commerce transactions take place this way

Private Key Distribution via Public Key Encryption

Public key cryptography is slow but makes it possible to distribute keys over a network that is not secure. Symmetric cryptography is fast, but distributing the secret key is difficult. Many e-commerce applications use the best of both worlds. The slower public key algorithm is used to distribute the secret keys used by symmetric cryptography.

- Beyond Secrecy -
Authenticity, Non-repudiation, and Integrity

- **Secrecy is not the only security concern.**
- **Authentication**
  - How does Bob know that an order, supposedly from Alice, is not from someone *pretending* to be Alice?
- **Non-repudiation**
  - If Alice sends an order to Bob, then later changes her mind, how can Bob prove that Alice did send the order.
- **Integrity**
  - How does Bob know that the order received from Alice has not been changed in transit by someone else.

Beyond Security: Authenticity, Non-repudiation, and Integrity

Business conducted solely in a digital environment has three major vulnerabilities.

Authenticity: How do we protect against one party pretending to be someone else? If black hat claims to Sears.com, an unsuspecting user might be willing to release credit card information to black hat, or transfer funds to a black hat account.

Non-repudiation: How do we protect against one party, or the other, claiming that they did not agree to a contract? In the physical world we use signatures and witnesses to prove that both parties agreed to a contract. Things are a bit more challenging in the digital world.

Integrity: How do we protect against someone changing a contract after it has been agreed to? Black hat may be clever enough to change the amount of a bank transfer from $100 to $10,000. In the physical world this type of change is difficult – in the digital it is just a matter of changing a few bits.

Public Key Solutions (1)

- **Key pairs work in either direction.**
  - The private key can encrypt, public key can decrypt.
- **Provides assurance**
  authentication, non-repudiation, integrity
- **If algorithm also provides encryption then:**
  - **one key MUST be used for encryption**
    and one for assurance.
- **Slow, especially on large files.**
- **Popular algorithms: RSA and DSA**

Public Key Solutions

So far, we have assumed that the public key would be used to encrypt data and the private key would be used to decrypt the data. But is this necessarily the only useful order? If Alice encrypts data with her private key and sends it to Bob, how will Bob decrypt the data? Bob can only decrypt the data using Alice's public key.

What would happen if the data had *not* come from Alice? Could her public key decrypt it?

If Alice's public key *does* decrypt the data, who *must* have sent it?

Public Key Solutions (2)

- **Authentication**
  - Alice encrypts her sales order with her private key. If Bob can decrypt the order using Alice's public key, then he knows the order is authentic.
- **Non-repudiation**
  - Because only Alice has her private key, she can't later deny sending the order.
- **Integrity**
  - But how does Bob know that the order received from Alice has not been changed by someone else?

Public Key Solutions

In addition to authenticating the sender, public key encryption can ensure non-repudiation by the sender. Only Alice has her private key. If a document has been encrypted with Alice's private key then she is responsible for that document. She cannot later repudiate her responsibility for the document. This concept of non-repudiation is at the heart of a new federal law that took effect in October of 2000 giving e-signatures the same legal standing as their handwritten counterparts. The Electronic Signatures in Global National Commerce Act, or E-Sign, as it is known, is designed to streamline commercial purchases of everything from stocks to houses.

Integrity can be assured by using a combination of compression and public key encryption known as a message digest or a hash.

Additional information on e-signatures can be found at:
http://rr.sans.org/encryption/esignatures.php

Hashing (Message Digest)

- **Extreme Compression!**
  - **200 M bytes can become 20 bytes.**
- **Collision**
  - **2 different documents producing the same Hash.**
  - **Weak collision resistance – likely**
  - **Strong collision resistance - unlikely**
- **Some algorithms require a key – some don't.**
- **Example algorithms:**
  - **MD (1-5), MDC, SHA-1, RIPE-MD, HAVAL**

Message Digest/Hash

A message digest is created by using an extreme compression algorithm. What is an *extreme* compression algorithm? Most compression algorithms remove redundant information that the receiver can predict. Programs like PKZip are able to first compress, and then restore the original message. However, compression algorithms used in making a *message digest* make the message MUCH smaller. In fact, this compressed message is made so small that the original message can no longer be recovered. However the compressed result is still a function of the original message. Any change in the original message will change the compressed message. This compressed message is known as a message digest or a hash.

How is this extreme compression used to ensure integrity? First, Alice compresses her message, creating a *hash*. Then the compressed hash is encrypted with Alice's private key. Both the original message and the encrypted hash are sent to Bob. Bob uses the same compression algorithm that Alice used on the received message to create his own message hash. Bob decrypts Alice's hash and compares it with the message hash that he created. Any difference indicates that Alice's message has been tampered with. If the hashes are identical, there has been no change to the message in transit, and Bob can be assured of Alice's message integrity.

Certificates – Trust in the Digital Age

- **Public Key algorithms, message digests, and hash functions can assure the authenticity and integrity of Alice's message.**
- **But how can Bob know that the person who claims to be Alice is really Alice?**
- **Someone that Bob trusts vouches for Alice.**
- **This is the concept of digital certificates.**

Certificates – Trust in the Digital Age.

A certificate is similar to a driver's license or a passport. It is used to prove the identity of the holder to others. Digital certificates users have two primary needs: Some users want to verify the identity of another user (i.e. on-line merchants), Some users want to have their identities verified by others.

X.509 vs. PGP Webs of Trust

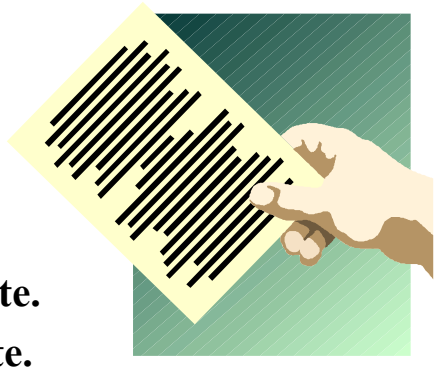
- **Certificates "vouch" for authenticity of the certified individual or organization.**
- **Two main types:**
  - **PGP webs of trust.**
    - **Designed to meet the needs of individuals - free.**
    - **Philosophy: If enough people say that you are Bob – then you must be Bob.**
  - **X.509**
    - **Designed for commercial use – fee based.**
    - **If an authoritative source says that you are Bob – then you must be Bob.**

X.509 vs. PGP Webs of Trust

Pretty Good Privacy (PGP) mail encryption and authentication is typically used in closely-knit communities (like people working on a common project). PGP webs of trust do not have a centralized authority to authenticate users; instead they depend on personal links that can be established between any two people in the community using a relatively small number of "hops" on the basis of common relationships. Can Carol trust Alice if they have never met before? Probably yes, because Alice has a good friend, Bob, who will vouch for her.

The X.509 model is based on certification authorities and is hierarchical. At the top of the hierarchy are a few Root Certification Authorities which are well known and which intrinsically must be trusted. Next in the hierarchy are Intermediate Certificate Authorities. Although Root CAs can directly issue user certificates, large or medium organizations may find it more flexible to set up a certificate authority of their own, so that they can issue and revoke certificates for employees without involving any higher authority. This certificate authority is called an Intermediate Certificate Authority.

What's in a X.509 Certificate?



Major Items include:

- **The Version (1, 2 or 3).**
- **A unique serial number.**
- **Expiration of the certificate.**
- **The issuer of the certificate.**
- **The issuers private key and method.**
- **The subject of the certificate.**
- **The subjects private key and method.**

What's in an X.509 Certificate?

X.509 was originally an ITU-T recommendation, but now most development seems to be coming from IETF (Internet Engineering Task Force). Because of its reliance on a hierarchical authority, X.509 seems to fit the corporate model more closely that PGP webs of trust.

Included in an X.509 certificate is: the version number (v1, v2, or v3), a unique serial number assigned by the responsible CA, the signature method used to sign the certificate, the ID of the CA that issued the certificate, not valid before / not valid after dates, subject name (ID), subjects public key, certificate signature (the hashed certificate body encrypted by the CA's private key) and other optional extensions.

X.509 Certificate Authorities

- **Some X.509 certificate authorities:**
  - **ANX eBusiness**
  - **Entrust**
  - **Geotrust**
  - **GTE Global Trust**
  - **Microsoft Authenticode**
  - **SecureNet**
  - **Thawte**
  - **VeriSign**

Steganography – hide in plain site

- **Steganography communicates while hiding the existence of that communication.**
- **Modern Steganography uses the redundancy in digital files (visual and audio).**
- **Useful for:**
  - **Ultra secure communications by combining steganography & encryption.**
  - **Copyright protection.**

Steganography – hide in plain site

Steganography is the practice of hiding information. This is a science as old as cryptography. The ancient Greeks hid information *under* the wax on writing tablets (the writing normally took place on top of the wax). Every school child has used lemon juice to make invisible ink. However, today steganographic techniques are able to hide information in binary files. These files could be programs, pictures, music, or almost anything.

While steganography does not encrypt data, it can be combined with encryption to provide an ultra-secure method of communication. This may be useful for governments. However, it is a serious challenge for law enforcement officials. In February of 2001, CIA Director George Tenet testified before the US Congress that steganography and encryption are the primary computer means used by Osama Bin Laden to coordinate his terrorist attacks.

However, steganography has some beneficial commercial applications. Enforcing copyrights in the digital age has been troublesome. It might be easier if there were some way to track a file to its original licensed owner (or creator). Steganography provides a tool for proving who created a file and tracking an illegally copied file back to its source. This use of steganography is known as digital watermarking.

Steganography – an Example

The logo on the left is the original graphic. The graphic on the right has a 1400 word document (45 KB) hiding in it.

Can you tell the difference?

Steganography – an Example

These two copies of the TrainingMagic corporate logo look identical. Both logos are in the bit mapped graphics (BMP) format. Both logos are the same size (1,302,528 Bytes). Yet the graphic on left has a 45 KB encrypted file hidden inside of it.

The Legal Issues

- Cryptography (and related technologies) can be legally complicated.*
- As a rule:
  - There are no legal limitations on cryptographic technologies used inside the US.
  - Recent changes in US law allow the *export* of strong cryptographic technologies to all but a handful of countries.
- The situation is less clear outside of the US
  - Check *import* laws for country in question.
- Even in developed western countries, laws regarding cryptography can be draconian.

*This slide is meant to provide a general overview – not legal advice. You must consult a qualified attorney for accurate legal advice.

Cryptography – Summary

- **Encryption takes meaningful data and causes it to appear random to everyone except the intended recipient.**
- **Cryptography reduces the risk of information compromise in a shared environment.**
- **Symmetric encryption is:**
  - **Fast**
  - **Well understood.**
  - **Has the key distribution problem.**
  - **Examples: DES, 3DES, AES, Blowfish, RC5, GOST.**

Cryptography – Summary

- **Asymmetric encryption:**
  - **Has very large keys.**
  - **Is slow.**
  - **Keys can be distributed publicly.**
  - **Is often used to exchange private keys.**
  - **Enables authentication and non-repudiation.**
- **Cryptographic algorithm vulnerabilities include:**
  - **Redundancies in the remaining information.**
  - **Redundancies in the key (especially in multiple keys).**
  - **Key compromise through social engineering.**

Cryptography – Summary

- **Non-repudiation of a message is achieved through digital signatures.**
- **Authentication of a message (or entity) is achieved through digital certificates.**

"It's personal. It's private. And it's nobody's business but yours." Phil Zimmermann, author of PGP