Secure Messaging

Paul Hrycewicz
Computer Science
October 20, 2022

Is any of this actually possible?

- Can you use your computer (or phone) to send a message to somebody that nobody else can read?
  - Answer: Yes.
- Do you need an advanced degree in Computer Science or Math to do this?
  - Answer: No.
- Is the software needed freely available, and easily installed and used?
  - Answer: Yes.
- Is there any way for law enforcement or government to prevent this from happening?
  - Answer: Practically speaking, no.

Agenda

- How do we send and receive “secret” messages?
- Encryption algorithms
- Secure messaging – asymmetric key encryption
- PGP (Pretty Good Privacy)
- Kleopatra Demo
- Is secure messaging really secure? Is it a good thing or a bad thing?

But First…Encryption For Kids

We did this as kids – we sent “secret messages” using a simple substitution cipher

Let’s send “HELLO”

H	->	I
E	->	F
L	->	M
L	->	M
O	->	P

message sent is “IFMMP”

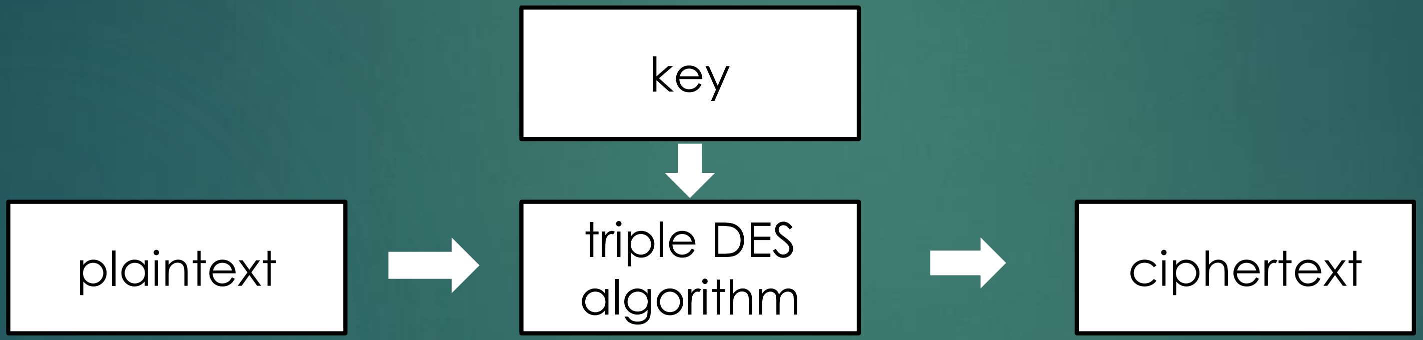
How did we encrypt?

How does the receiver decrypt?

Shared-Key (Symmetric) Encryption

The example we just did shows ***shared-key encryption***

For shared-key encryption, both the sender and receiver must have the algorithm and the key



It's "symmetric" because the same algorithm and key are used by the sender and receiver

Advanced Encryption

One example – Triple DES

Given: the algorithm (Triple DES) and an encryption key (3 x 64 bits)

- Shuffle the bits in the plaintext
- Derive 16 different keys from the given key using a combination of shifting and shuffling
- Split the plaintext into two halves
- Encrypt and re-encrypt each half 16 times using the 16 keys
- Combine the two halves into text
- Repeat the above process three times using the three given encryption keys

Plaintext + key = ciphertext:
hello + 2jd8932k = X5xJCSyc

Ciphertext + key = plaintext:
X5xJCSyc = + 2jd8932k = hello

Characteristics of Shared-Key Encryption

It’s secure as long as:

the encryption algorithm is sufficiently complex

the shared key is known only to the sender and receiver

What do we mean by “secure?”

It means that a third-party eavesdropping on the transmission line will not be able to decipher (decrypt) the message

Big problem:

How does the sender get the key to the receiver? (key exchange)

- Cannot send in plaintext, as that could be intercepted
- Cannot encrypt and send, as the receiver would not know how to decrypt
- No such thing as a “secure” transmission line
- Possible to “hide” a key within an object, such as a picture, but this can be detected

Asymmetric Key Encryption

Solves the key-exchange problem

Developed by Whitfield Diffie and Martin Hellman, with help from Ralph Merkle (all then at Stanford), commonly referred to as ***Diffie-Hellman Key Exchange***, published in 1976 (see "New Directions in Cryptography," IEEE Trans. Information Theory, 1976)

The method centers on two keys: a ***public key*** and a ***private key***

Public Keys and Private Keys

To receive an encrypted message, the receiver creates a public key and a private key

Very easy to do, but underlying mathematics are complex

see: ssh-keygen

The receiver tells the world what their public key is, but keeps the private key secret

The keys are mathematically linked using large prime numbers. The encrypted text is also mathematically linked with the keys.

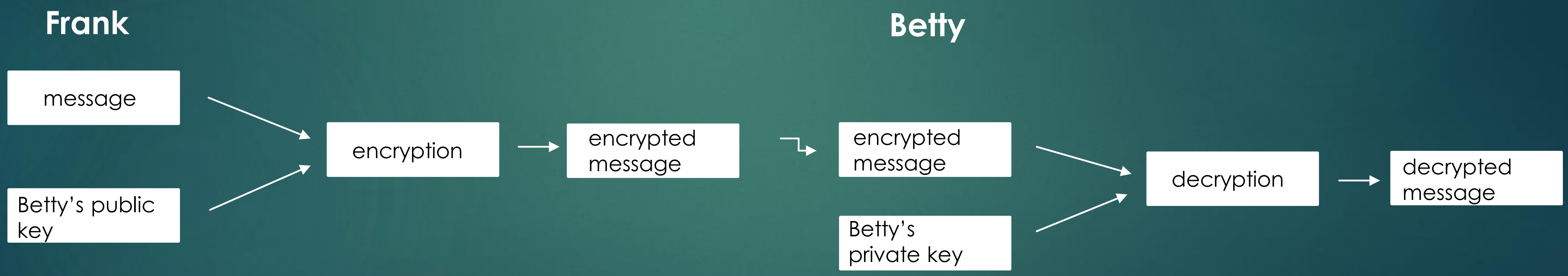
Given keys of sufficient complexity, it is practically impossible to derive the private key from its public key

Sending a Secure Message

Let's assume Frank wants to send Betty an encrypted message.

Betty creates a public key/private key, and broadcasts the public key, but keeps the private key secret

Frank encrypts the message using Betty's public key, and Betty decrypts it using the private key



As long as Betty's private key stays private, the message is secure

The messaging is secure since the private key is never exposed, like the key with symmetric encryption

PGP (Pretty Good Privacy)

Phil Zimmerman developed PGP and released it in 1991 as the first package to use asymmetric key encryption

The US government began a criminal investigation of Zimmerman for "munitions export without a license." Zimmerman responded by publishing the source code for PGP. The government eventually abandoned its investigation

Zimmerman's work eventually became the OpenPGP standard

Today PGP is the most widely-used encryption process

It's easy to download, install, and use. See a version called Kleopatra

Another popular use of asymmetric key encryption, in conjunction with "cryptographic hashing," provides digital signatures

A digital signature is a method of preventing an impostor from sending you a message

Public / Private Key Generation

It’s easy for a user to generate keys – use ssh-keygen command on Windows, Mac, Linux

Generates keys approximately 14,000 bits long

Using ssh-keygen is easy. PGP is easy. Kleopatra is easy. The real challenge is generating keys that cannot be discovered

Many such key generation algorithms exist

The easiest method to understand is based on the multiplication of large prime numbers

The public key is chosen using the product of the primes

The private key is derived using the public key and the original two primes

It is very easy and fast to multiply the original two primes, that’s just straightforward multiplication

But it’s time-impossible to determine the factors of the public key, which are needed to derive the private key

Kleopatra Demo

Kleopatra
Crypto Manager

Send a secure message from frank@laspositascollege.edu to betty@laspositascollege.edu

Download kleopatra from www.openpgp.org

Software is open-source and signed, so you can be sure you download, install, and run the real software and not some impostor

The software can be embedded into any applications you wish, including email and text messaging

Is Secure Messaging Really Secure?

The short answer is yes, at least for now, if you do it correctly

You need to generate sufficiently random keys of sufficient length to prevent patterns from occurring and to guard against brute-force attacks

Potential weaknesses:

- poor random-number generation
- cursor movement patterning
- man-in-the-middle attack
- human factors
- quantum computer attacks

Estimates of the length of time required for a brute-force attack range from 70 years to millions of years, depending on the key length

Key generation and other security-related algorithms are being developed that are impenetrable by quantum computers

Is Secure Messaging Good or Bad?

On the plus side, HTTPS is based on asymmetric key encryption

On the negative side, secure messaging can be used for criminal or terrorist activity

No matter what your point of view, or that of any government, there is no way to prevent secure messaging

Prevention is also beyond the reach of device manufacturers. They can provide encryption software, or allow it to run on their devices, but since they do not control the keys or algorithms, they cannot decrypt messages or provide information to governments

A backdoor into an encryption system can be built-in, but the system then becomes immediately untrusted. Why would anybody use an untrusted system?

OpenPGP is one example of open-source software, which can be inspected and validated by anybody before it is used

Thank You!