SECURITY

Today

- Today we will finish up the Unit on Net-centric computing
  - Yes, I know it isn't in order.
- We will cover some general material on computer security.
- We will look at cryptography in a bit of detail
  - Including a brief look at the support Java provides for encryption and decryption.

Security

- Overriding question: *how can we guarantee security?*
  - In the presence of people who attack systems.
- What makes networks applications insecure?
- What forms of attack are possible?
- What are the security requirements for distributed and open systems?

Why is security a problem?

- Internet and WWW computing standards (IP, HTTP, etc.) are public.
  - *Security problems arise from the Internet and WWW being open and distributed systems.*
- The Internet is open and pervasive.
  - Anyone may connect to it, connections are everywhere.
- The Internet has many interconnecting components.
  - A message from one computer to another will pass over many others, thus giving these others opportunities for espionage.

- Web servers are extensible:
  - Can be connected to all types of technologies (e.g., database servers) which may transform their functionality.
- Development speeds are very fast.
  - Until recently, security has often been ignored in software development!
- HTML/HTTP were not designed for e-commerce but for document storage and transfer.
  - Browsers had limited functionality initially.

Forms of attack

- Threats to integrity
  - An intruder modifies data stored in a database.
  - Data is modified in transit.
- Confidentiality threats
  - Reading private data of a company or an individual
- Denial-of-service threats
  - Doing something which prevents some system from fulfilling its function.
- Authentication threats
  - Pretending to be someone who you are not.

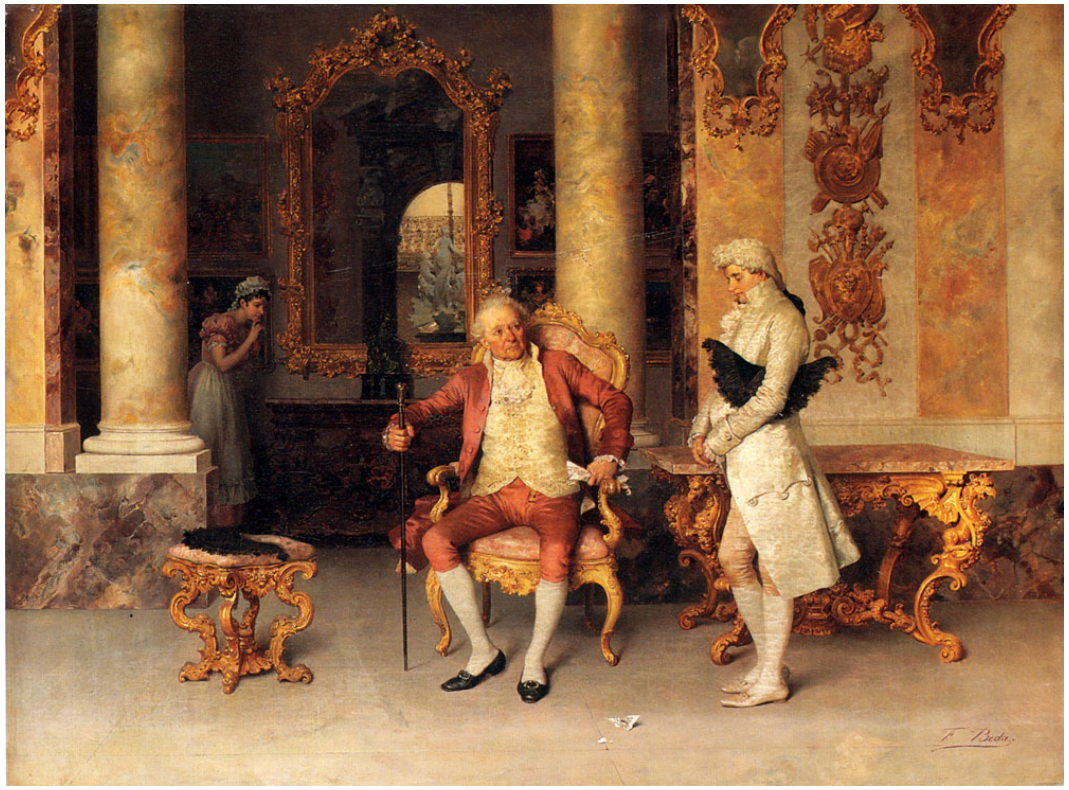
Types of attack

- Passive attacks.
  - Attacker observes.
- Active attacks
  - Attacker engages the attackee's system.

Passive attacks

- Eavesdropping on messages
  - Intercepting and reading.
- Traffic analysis
  - Guess the content of messages by looking at the pattern of traffic.

- In both cases the attacker just observes.



- Francesco Beda: “The Eavesdropper”

Active attacks

- Masquerades
  - Pretending to be someone whom you are not.
- Replay attacks
  - Capturing a data unit and retransmission for an unauthorized effect.
- Modification-of-message attacks
  - Man-in-the-middle attacks
- Denial-of-service attacks

- These kinds of attack are not independent of one another.
- For example, one can use a man-in-the-middle attack to gain a password and then use the password to carry out a masquerade attack.

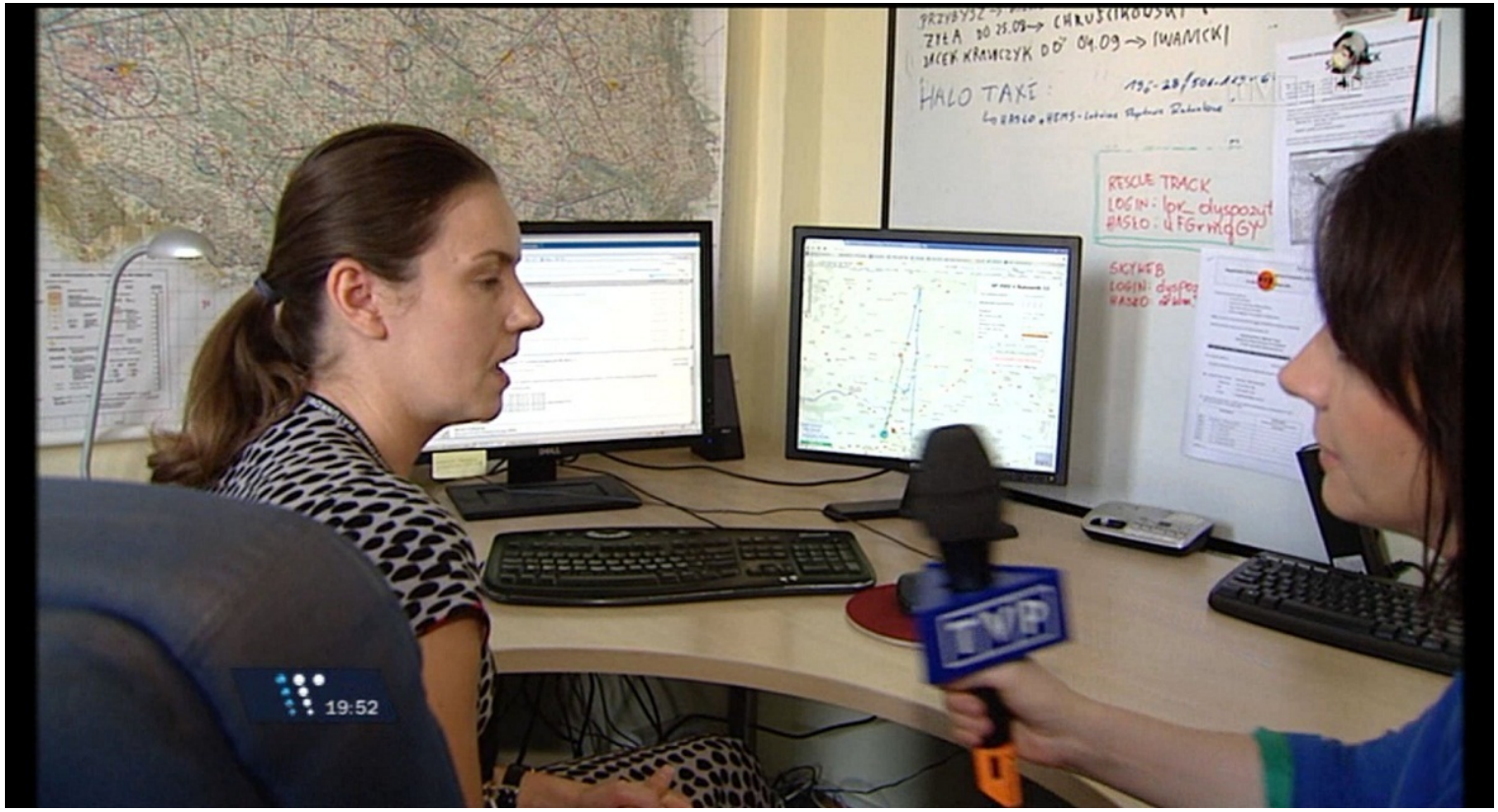
Security requirements

- Confidentiality
  - Information should not be accessible by unauthorized parties.
  - Protect against passive attacks.
- Authentication
  - Make sure that the originator of a message or transaction is who they say they are.
  - Protect against masquerades and replay.
- Integrity
  - Only authorized parties are able to change data, and only under pre-defined circumstances.
  - Protect against modification-of-message attacks.

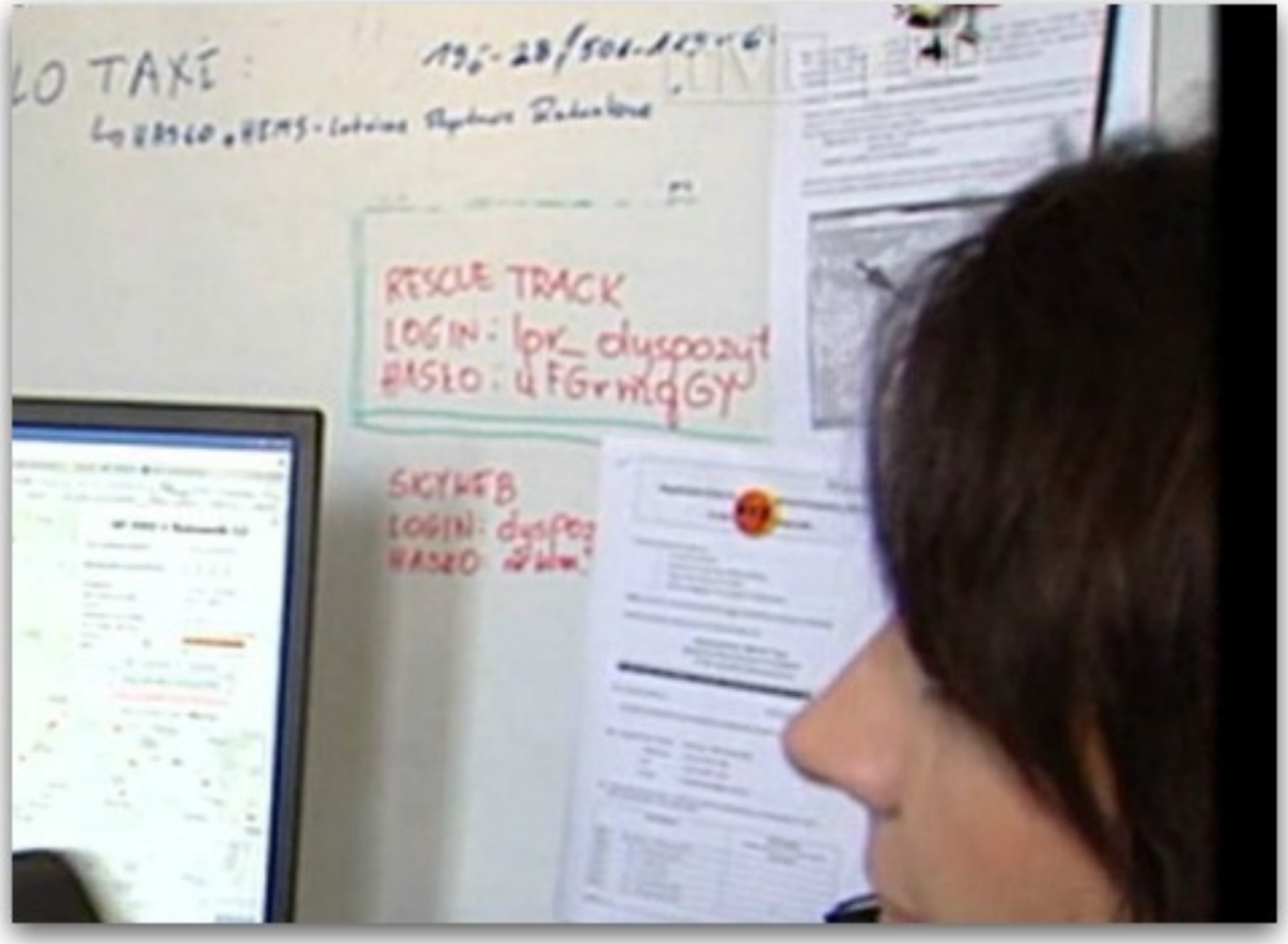
- Non-repudiation
  - Neither the sender nor the receiver of a message can deny that it took place.
- Access control
  - Only authorized users are able to access resources within a system.
- Availability
  - The resources of a system are available to authorized users when they need to use them.
  - Protection against denial-of-service attacks

- Don't underestimate the importance of *physical* security.
- Securing:
  - Machine rooms
  - Network ports
  - Mac addresses

  is a quick security upgrade.
- Reason that physical tokens and biometrics are used.
- Shred documents (especially manuals).
- Don't write down passwords.



- from Polish TV.


TAXI
RESCUE TRACK
LOGIN:
HASŁO:

Low-tech attacks

- Guessing passwords
- Stealing passwords
  - Dumpster-diving
- Taking advantage of poor physical or clerical controls
  - Bank employees creating fictitious accounts
- Phishing
  - Soliciting information from users which appears to be genuine but is not.

Destructive devices

- E-mail bomb
  - An email with a very large attachment, thus overwhelming the connection.
- Denial-of-service attacks
  - Hanging browser requests.
  - Programs which spawn others, which in turn spawn others.
  - The ping of death — Packets of data larger than permitted by TCP-IP often cause the receiving computer to crash.

Viruses

- Malignant code which attaches itself to files resident on a computer.
  - Delete or overwrite files
  - Send malicious emails.
- Types of viruses:
  - Executable virus: attaches to an executable file.
  - Data virus: infects a file containing data (e.g., a startup file)
  - Device driver virus: infects the device drivers of a system

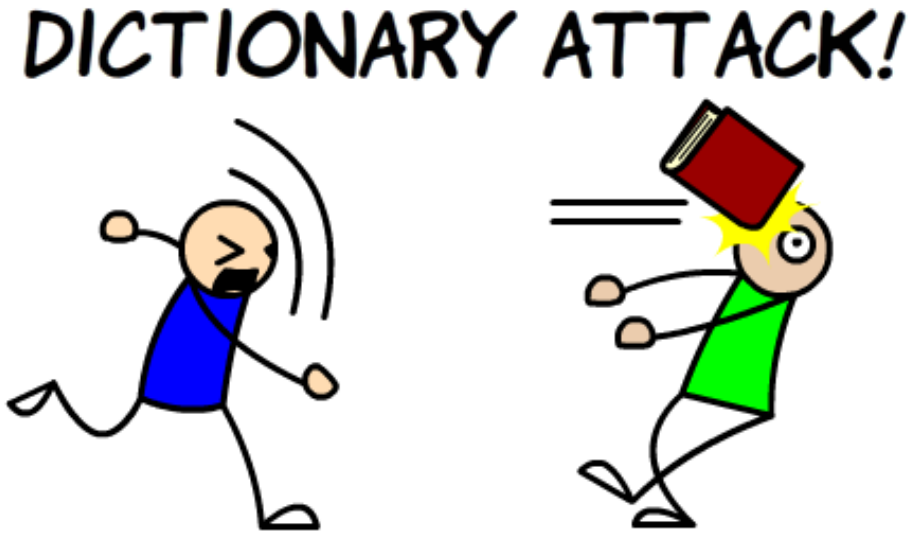
- Anti-virus software
  - Usually looks for known viruses or for sudden changes in the size and content of files.
- Clever viruses work against anti-virus software
  - Mutate: changing their form or location
    "polymorphic viruses"
  - Hide themselves in other files ("stealth viruses")

Scanners

- Programs which detects security weaknesses.
- Usually developed to help human system administrators to find security flaws.
- Looks at the components of an Operating System and checks for security.

Password crackers

- Software which attempts to find passwords (e.g., by randomly generating all, or most likely, possibilities).
- Often use common words (e.g., “password”, “pword”).
- Dictionary attacks.



(Not this kind of dictionary attack)

Sniffers

- Devices which read the packets of data travelling on a network.
- Usually used by system developers and administrators to assess system effectiveness.
- May be able to detect passwords.
- Used to gain the data for traffic analysis.

Trojan Horse

- Code which looks legitimate but executes something not expected or authorized.

- Difficult to detect, because they often masquerade as utilities or OS components.
- *Social engineering*.

Spoofing

- Using one computer to masquerade as another, where the latter has privileged access to some third system
- MAC address spoofing, for example, can be used to gain unauthorised access to a wifi network.

Designing a secure system

- Assume two computers which wish to communicate over the Internet. We call these two the Principals.
- Techniques for security have two basic components:
  - Transformation of the information to be sent, e.g., putting a message into code before transmission.
  - Some secret information held by the two principals but not by any opponent, e.g., the code to be used for transformation.
- A trusted third party may be necessary to ensure secure transmission.

- Design a security service
  - Design an algorithm for transforming the information to be communicated.
  - Generate the secret information to be used with the algorithm. e.g., the coding scheme.
  - Develop methods for the distribution and sharing of the secret information, e.g., via a trusted third party
  - Specify a protocol to be used by the two principals that uses the security algorithm and the secret information.

Basic principles of cryptography

- Messages are put into code (encrypted) by the sender and decoded (decrypted) by the receiver.
- Basic ingredients of conventional cryptography:
  - Plain text input
  - Encryption algorithm
  - Key for encryption and decryption
  - Cipher text (coded input text)
  - Decryption algorithm

Example

- Suppose input text is

  THE SKY IS BLUE

- Algorithm:
  - Replace each letter by the letter in the alphabet 1 step along.
- Key: 1
- Output:

  UIF TLZ JT CMVF

- A key of 2 would produce:

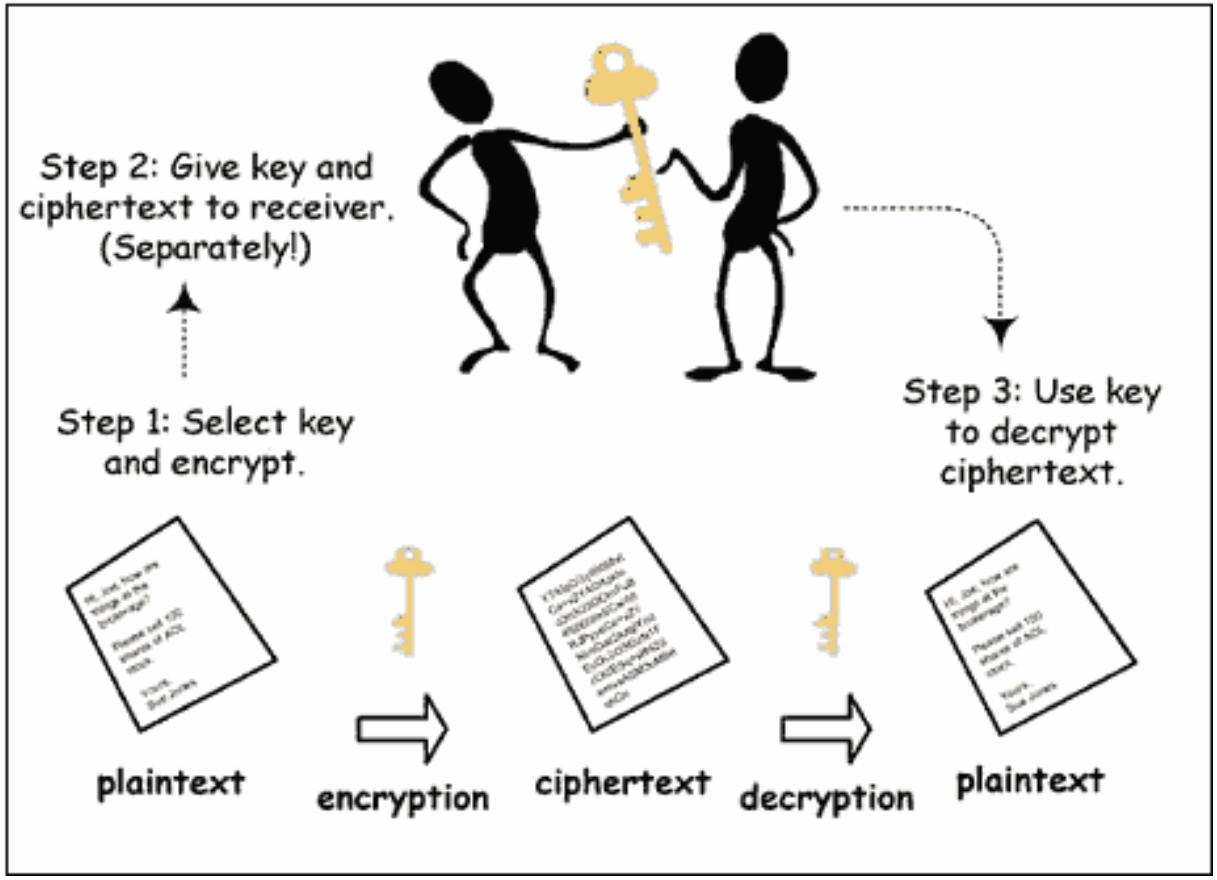
VJG UMA KU DNWG

- When we use a key of 3, this is known as the *Caesar cipher*

- The process used here is called *substitution*
  - Substituting one element (in this case a letter) by another.
- Another process is *transposition*
  - Moving parts of the message around:

TLZ UIF JT CMVF

- If we were to use the Caesar cipher to exchange messages, we would have a setup like this:

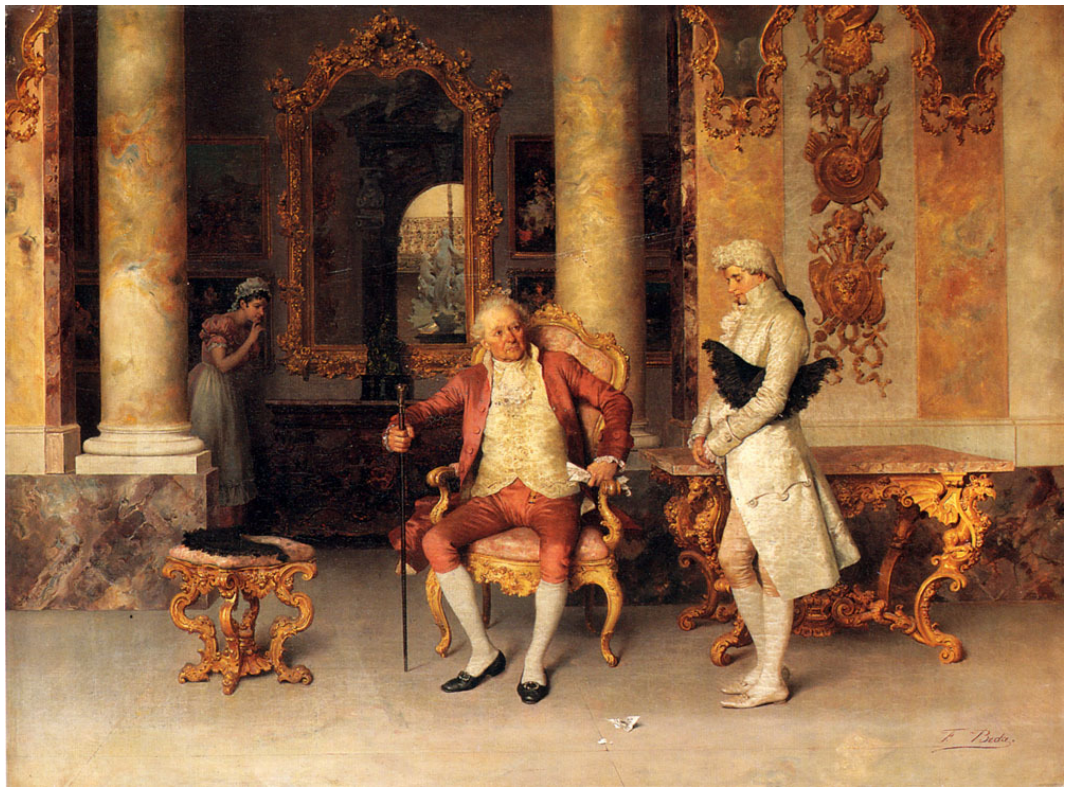


Requirements

- An encryption algorithm is needed.
  - The sender must have this.
- The receiver must have a decryption algorithm
  - Undoes the encryption algorithm.
- Ideally, we would like a *strong* encryption algorithm, secure against attack.
- This is usually stated as follows:

  An opponent should be unable to decrypt the ciphertext or discover the key even if s/he is in possession of a number of ciphertexts together with the plain text which produced them.

- Assume someone is listening in.



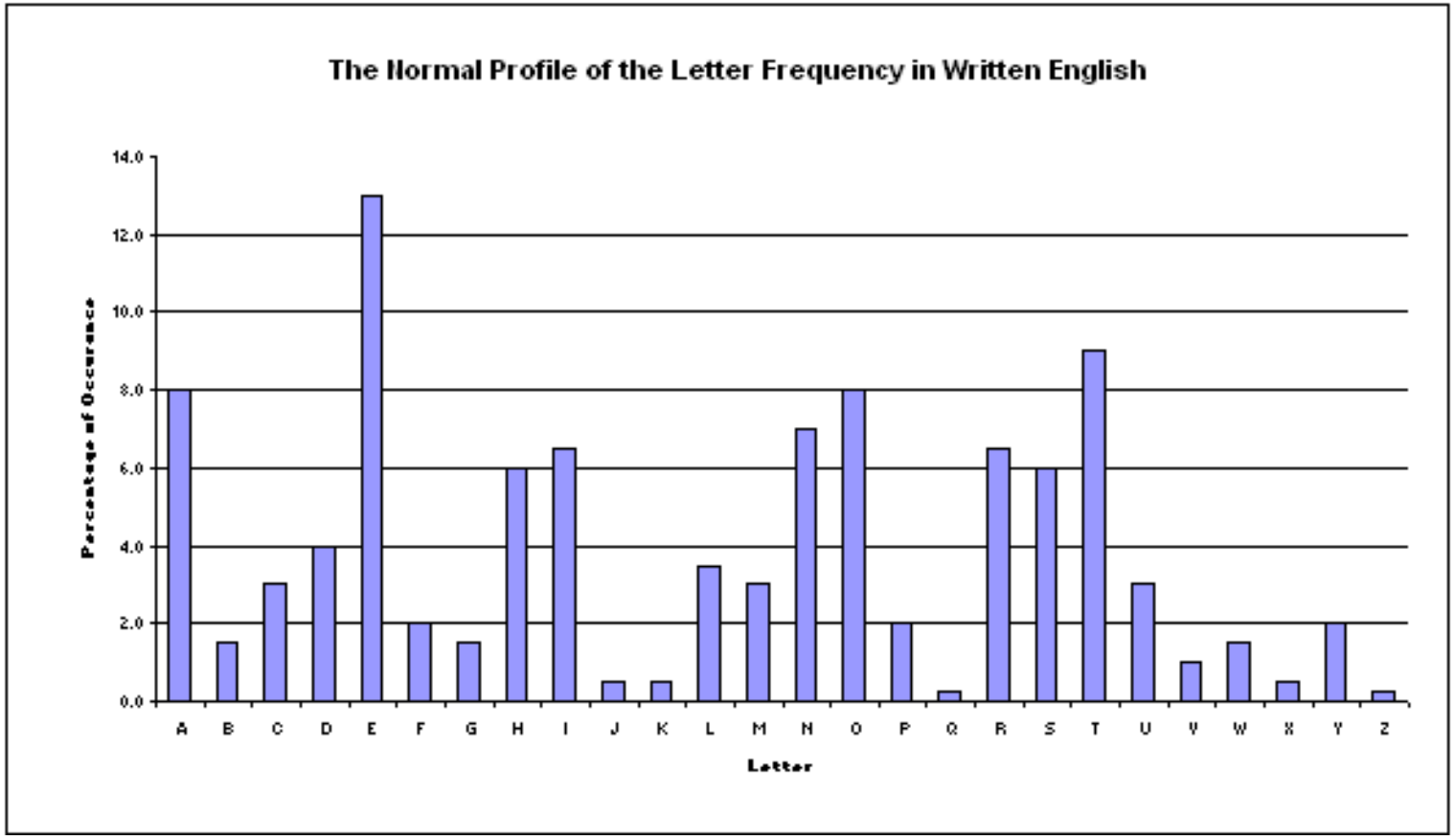
Classification of cryptographic systems

- The type of operations used to transform plaintext to ciphertext:
  - *substitution*
  - *transposition*
  - Usually some complex combination of these is used.
  - In any case, no information can be lost in the process.
- Whether sender and receiver use the same keys
  - *symmetric*: sender and receiver use the same keys
  - *asymmetric*: sender and receiver use different keys

- How the plaintext is processed.
  - A *block cipher* processes the input one block of elements at a time, producing an output block for each input block.
  - A *stream cipher* processes the input elements continuously, producing one element at a time as it goes along.
- The number of keys used

Cryptanalysis

- The process of attempting to discover the plaintext or the key.
  - From one you can (often) get the other.
- Problem with Caesar cipher and variations is that they are vulnerable:
  - Frequency analysis
- Known by (at least) the 16th century.


The Normal Profile of the Letter Frequency in Written English
Percentage of Occurance
14.0
12.0
10.0
8.0
6.0
4.0
2.0
0.0
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
Letter

Cryptanalysis attacks

- Frequency analysis
  - As above.
- Known plain text attack:
  - The opponent has a sample of plaintext and ciphertext, and from this infers the keys;
  - Easier if you know the algorithm.
  - Note that plain text may be compressed and may be numerical in origin, so brute force methods usually require some knowledge of the type of plain text used.
  - For a key of length 128 bits, it would take an opponent about 10^{18} years to crack!
  - Part of what cracked the Enigma cipher in WW2

- Chosen plain text attack:
  - The opponent gets the computer doing the encryption to encrypt some specially-chosen text (chosen to give clues about the ciphertext); e.g., the blocks may have many blanks or repeated words.
- Differential cryptanalysis attack:
  - The opponent gets the computer doing the encryption to encrypt several blocks of text which differ only slightly; e.g., the opponent then looks at the differences in the ciphertext.
- Differential fault analysis:
  - The opponent attacks the hardware of the encryption computer to force it to make mistakes, in order to discover the key or algorithm.

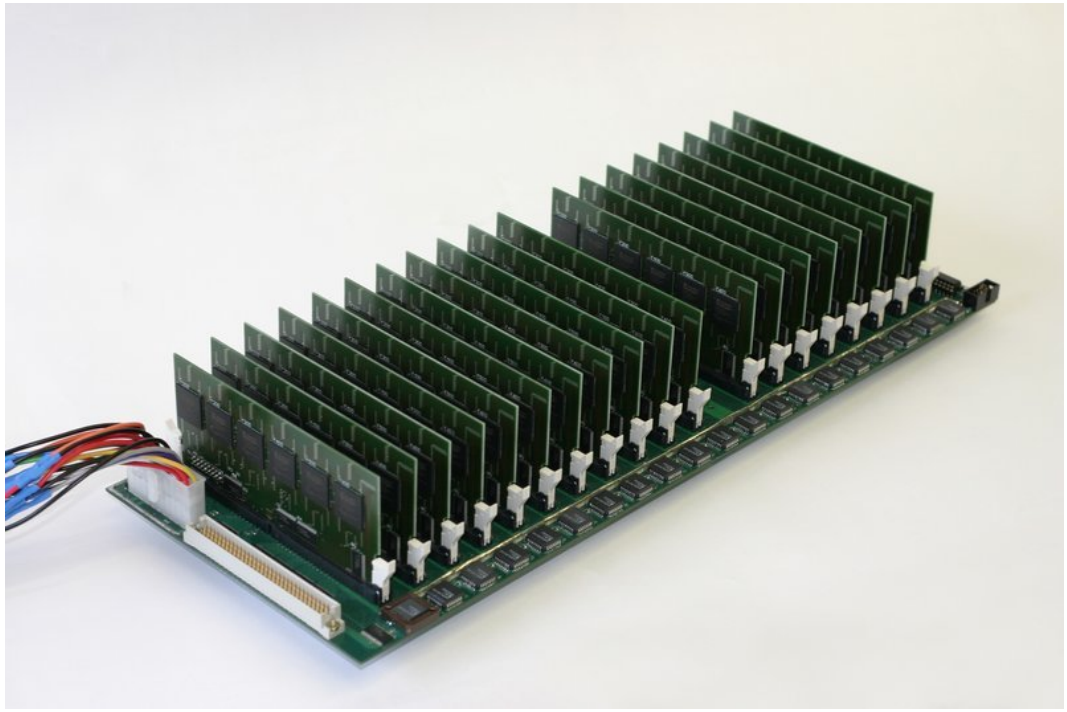
- Steadily more complex forms of cipher were developed, but they did was to make it harder to obtain the key through cryptanalysis.
- Polyaphabetic ciphers:
  - Vignère
  - Enigma
- Eventually got to keys and algorithms that can provide strong encryption.

Some symmetric algorithms

- Data Encryption Standard (DES)
  - US Government standard in use 1977–1998.
  - Algorithm works on blocks of data (each 64-bits), and uses a 56-bit key.
  - Decertified in 1998 and replaced by 3DEA.
  - Possible to crack with brute force to find the key (i.e. key is too short).

- Triple Data Encryption Algorithm (3DEA)
  - Proposed in 1979, became a standard in 1999.
  - Applies the DES algorithm 3 times to plain text.
  - Encrypt with Key A, decrypt with Key B, encrypt with Key C.
  - Decryption is the reverse with the keys reversed.
  - Decrypt with Key C, encrypt with Key B, decrypt with Key A.
  - Each key of length 56 bits, so effective key length of 168 bits.

- Note that what counts as "strong" is clearly dependent on the current state of hardware.
- DES
  - 56 bits $= 7.2 \times 10^{16}$ keys
  - About 2300 years at one million a second



- 9 days to crack a DES key.

Weakness of Symmetric Encryption

- Both sender and receiver must have the secret key(s) for the process to work.
- This is the key weakness of symmetric encryption methods.
- Note: the security of symmetric encryption depends on the secrecy of the key, not secrecy of the algorithm.
- If the key is big enough
  - Meaning there are enough possibilities
- If we assume that it is impractical to decrypt a message on the basis of the ciphertext plus knowledge of the algorithm, then security hinges on keeping the key secret.

Location of encryption

- Where should the encryption occur:
  - End-to-end encryption.
    The entire message is encrypted.
  - Link encryption.
    Each link along the way is encrypted.
    Not as secure as end-to-end.
- End-to-end encryption can only encrypt the data portion (contents) of packets.
- Headers are not encrypted (contain destination and source).
- Thus, the traffic pattern is not secure.
- Traffic analysis.

- Both methods can be used together.
- One key and method for the end-to-end encryption of data portion of packets.
- One key and method for the link encryption of the packet header.

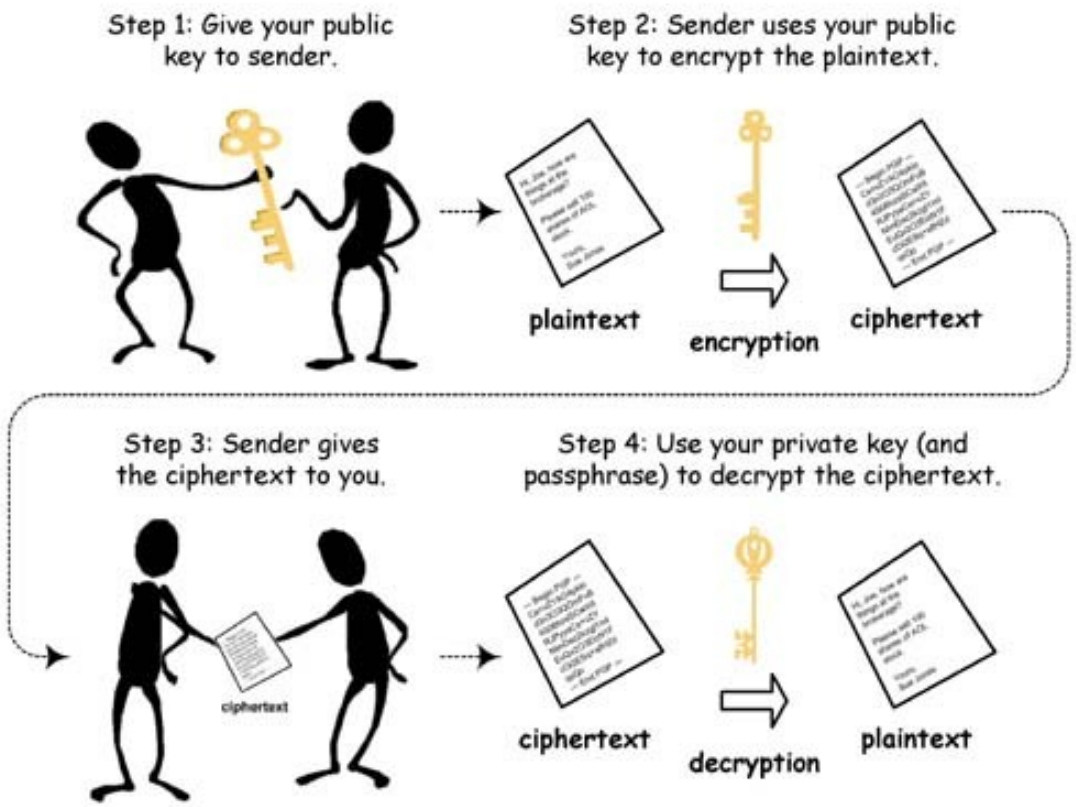
Key distribution

- How to A and B share keys?
- Options:
  1. A selects key and delivers it physically to B.
  2. A third party C selects the key and delivers it physically to both A and B.
  3. If A and B have previously used a key, they could send a new key using an encrypted message.
  4. If A and B each have an encrypted connection to a third-party C, C could deliver a new key to each of A and B on encrypted links.

- For link encryption, options 1 and 2 may be feasible.
- But these are not usually feasible for end-to-end encryption.
- Option 3 is vulnerable:
  - If an opponent ever finds a key, all future communications can be read.

- Options 1 and 2 especially feasible if you have enough money or only a few folk to communicate with
  - Secure couriers
  - One-time pads
- What governments often rely on.
- But what about the rest of us?

Asymmetric (or public) key algorithms



- Similar to symmetric key encryption, but we use at least 2 keys: One for encryption (the public key), and one for decryption (the private key).
- The steps involved are:
  - Keys are generated in pairs, a public key and a private key, by each person or computer.
  - The public key is made public.
  - Anyone who wants to send a message to Bob, uses his public key.
  - Bob decodes the message using his private key.
- This approach dates from 1976.

- Public key methods are not necessarily more secure than symmetric algorithms.
- But there is a larger computational overhead, and this makes brute-force attacks harder to execute successfully.

Requirements for public key algorithms

- It is computationally easy for party B to generate a pair of keys.
- It is computationally easy for sender A to generate the cipher text on the basis of the plain text and the public key.
- It is computationally easy for party B to decrypt the resulting ciphertext using his private key and so generate the plain text.
- It is computationally infeasible for an opponent to determine the private key from the public key.
- It is computationally infeasible for an opponent to recover the original plain text from the public key and the ciphertext.

Ecryption basics

- The number theory that underpins curent encryption techniques hinges on two notions.
- One is the modulus operation.

$$6 \bmod 4 = 2$$

$$100 \bmod 27 = 19$$

- The other is the greatest common divisor.
- For a and b, the gcd (a, b) is the largest number that divides a and b with no remainder.

$$(15, 21) = 3$$

$$(76, 190) = 38$$

- The message is a number M.
- The coding is done by a public function $f(M)$, where A is the only person who knows how to compute f^{-1}.
- Q: How is this possible?
- A: Trapdoor function, something that is much harder to undo than to do.

- A chooses two large prime numbers p and q, and keeps them secret.
- A announces:

  $$n = pq$$

  and e where:

  $$(e, p-1) = (e, q-1) = 1$$

  that is the public key is the pair $\langle n, e \rangle$.
- The encoding is:

  $$f(M) = M^e \bmod n$$

  where M and $f(M)$ are both $\leq n - 1$.

- Decryption:

$$f^{-1}(C) = C^d \bmod n$$

  so the private key is the pair $\langle n, d \rangle$.

- d can also be computed from p and q
  (though by a more complex operation than multiplication).

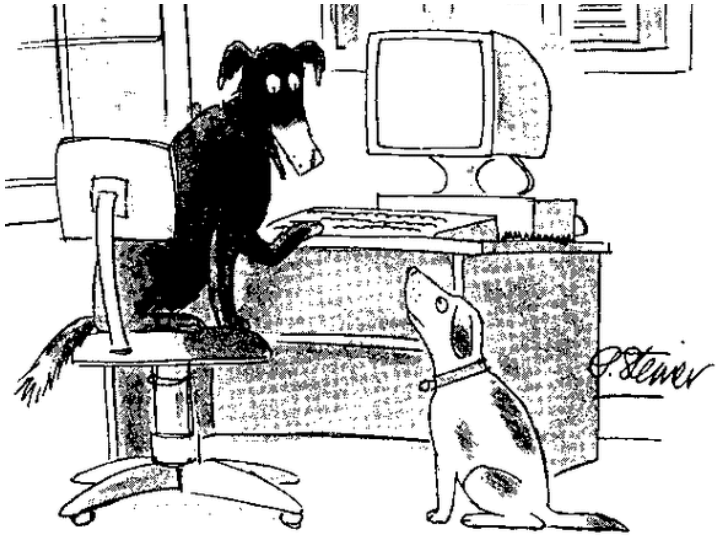
- Both encryption and decryption are reasonably efficient, even for large n.
- Factoring n to find p and q is still hard.
  - Better than naive algorithms exist.
  - But no efficient algorithm exists.

RSA cipher

- Above idea implemented as a block cipher
- The plaintext and cipher text are integers between 0 and $n - 1$ for some $n = pq$.
- The bigger p and q are, the bigger the chunk of text encoded by a single operation
  - Efficiency
  - Security
- The bigger p and q, the harder to factor n.
  - Security
- Algorithms to find prime numbers efficiently are valuable.

Digital signatures

- Algorithms like RSA provide secure encryption.
- Howewver, that is not all that we need.
  - Also need ways to verify who is who.



"On the Internet, nobody knows you're a dog."

- We use the public and private keys in reverse order to do this

- Alice sends a message to Bob
  - Alice encrypts the message using her private key, which only she knows.
  - Bob receives the cipher text, and decrypts it using Alice's public key.
  - If Alice really did send the message, the output should be plain text.
  - If someone else (say, Mary) sent the message, then Alice's public key will not work on this message.
- So, this provides a way to authenticate a message, assuming the private key has not been stolen or somehow made public.

- Note that this method does not keep the message secret.
- Anyone can use Alice's public key (since it is public!) to decode Alice's message.
- This makes it possible to use this approach for *digital signature*, analogous to the use of your handwritten signature.

Digital Signature Standard (DSS)

- A standard agreed in 1993 for digital signatures in the American National Institute of Standards.
- Only used for digital signatures (not for encryption or key exchange).

Digital certificates

- How do we distribute public keys?
- Answer is simple: put on your web-site, email your friends, shout it from the roof-tops!
- But if Alice gets an email from Bob telling her that 1023 is his public key, how does she know it really is his?
  - Maybe someone is impersonating him and sending out a false key in his name!
- Digital Certificates seek to get around this.

- A user (e.g., Bob) presents his public key to a trusted third party and receives a digital certificate.
- The certificate contains a public key together with a a user ID for the key owner (Bob), all signed by the third party.
  - Examples of third parties: Government agencies or a bank.
- The user (Bob) can then give the digital certificate to anyone else (e.g., Alice).
- A standard for digital certificates is X.509.

Key Distribution Revisited

- Can use public key encryption to share secret keys:
  1. Bob sends Alice his public key using a public key certificate.
  2. Alice prepares a message.
  3. Alice encrypts the message using one-off symmetric key. Session key.
  4. Alice encrypts the session key using Bob's public key.
  5. Alice attaches the encrypted session key to the message and sends it to Bob.
- Only Bob is able to decrypt the session key (since only he has his private key).
- So, only Bob can read the original message.

- Why do this?
- Well, encryption and decryption is typically faster using symmetric key techniques.
- So, less computationally intensive than using public key for the whole message.
- Also, since the public key bit is only encrypting a session key you reduce the possibility of cryptanalysis.
  - Public key is only used to encrypt a key that keeps changing.
- This is used in PGP (see later).

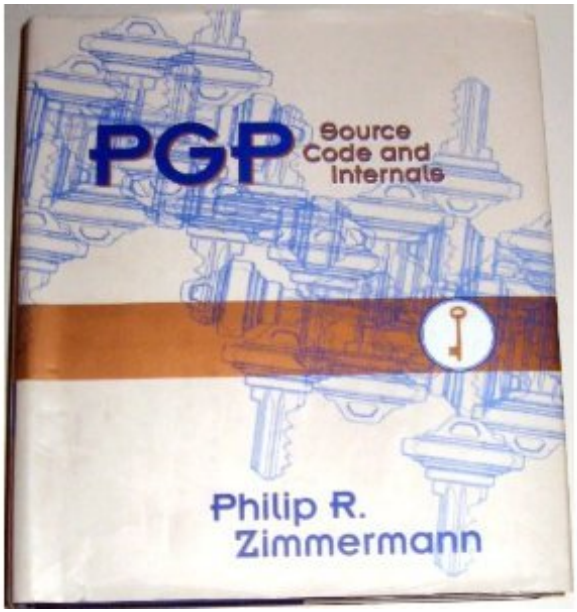
Message digest functions

- These produce a summary digest of a file, and can be used to see if the file has been altered.
- Useful for detecting presence of viruses or tampering by opponents.
  - I give you the file and the digest.
  - You check if the file you receive matches the digest.
- Can use for message authentication.
- Examples: HMAC, MD series (128 bit digest), SHA series (160 bit digest).

PGP

- "Pretty Good Privacy"
- A publicly-available system for encrypting files and email messages
- PGP uses:
  - RSA for management of keys in symmetric encryption
  - IDEA algorithm for sending data using symmetric encryption
  - MD5 scheme for ensuring no tampering.
- Main weakness: if a public key is compromised, than a revocation certificate has to be issued to everyone in contact with the person whose keys are compromised.

- At the time PGP came out, the US government had decided that strong cryptography was a muntion.
  - Therefore subject to export licences
- Essentially declared it illegal to distribute PGP

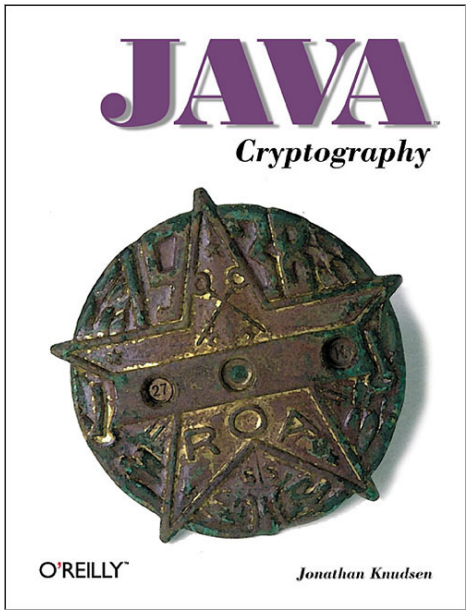


- Printing this book was illegal as was exporting it.
- MIT (and MIT Press) faced down the NSA.
- (Or maybe the NSA decided they didn't care since they can break PGP).

Other notable uses of crypto

- PCT — Microsoft product similar to SSL
  (Secure Sockets Layer, to be discussed later)
- SHTTP — A secure version of HTTP. Did not take off.
- SET (Secure Electronic Transactions)
  For credit card information.

Crypto in Java



- Java has a lot of support for cryptography.
  - `java.security.*`
  - `javax.crypto.*`

Java and message digests

- `Masher` is an example of a program that can computes a message digest.
  - Also illustrates reading from a file.
- Important to note that:
  - Most of the hard work is done by a method from the Java security package that implements MD5.
  - A good chunk of the code in `Masher` and all of `Base64` is bit really necessary.
  - Just allows us to see a readable representation of the digest.

Java and encryption

- `SecretWriting` is an example of a program that uses symmetric key methods to encrypt and decrypt text.
- (Don't make the mistake of thinking that this program alone could be useful to do serious encryption/decryption since its key management is horrible).
- Again it mainly demonstrates how easy this kind of thing is in Java.
  - Key creation is a single line of code.
  - Encryption is a single line of code.
  - Decryption is a single line of code.

  and a lot of the code, again, is producing readable output.
- Uses the `javax.crypto.*` implementation of DES.

Other security tools

- Crypto is not the whole story.
  - If you think cryptography is the answer to your problem, then you don't know what your problem is.

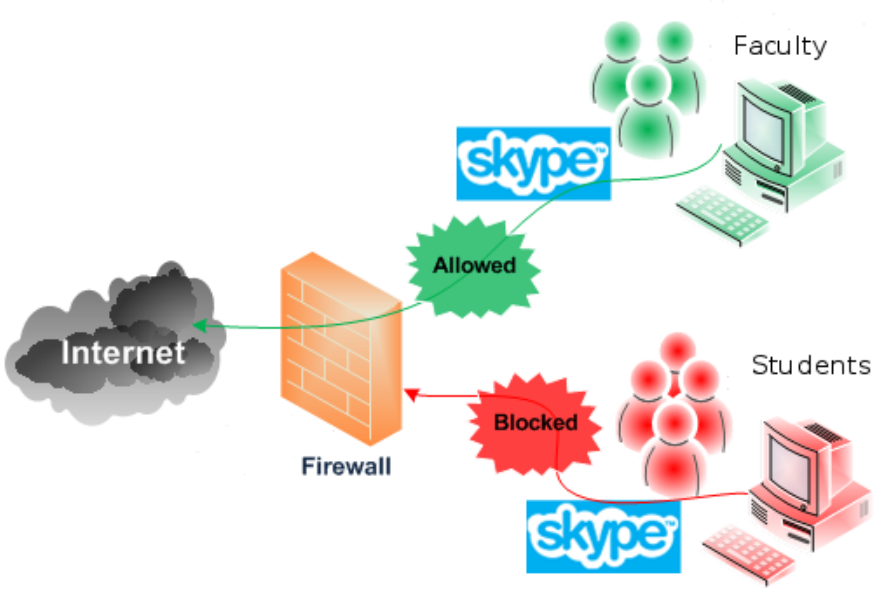
    Peter G. Neumann

- Logging tools
  - Software tools which monitor use of a computer
  - Log particular events, e.g., user logins, transferring web-pages, attempts to access secure files
  - Check for unusual events, e.g., access at unusual times, a user logging in and out repeatedly (could be seeking to gather info), a user mistyping a password (could be an attempted hack), a user accessing strange web-sites (e.g. military sites)
- Virus scanners — software which looks for unusual changes to files.
  - Especially operating system files.

- Security Checking software — used by system administrators to identify potential problems, e.g., scanners, password cracking software
- Network topology techniques
  - Design the topology of the network so as to make intrusion difficult
  - A common technique is a *firewall*
  - This is an extra element placed between a network (or a network element) and the external world.

Firewalls

- Functions of a firewall
  - To monitor traffic into or out of a private network
  - To reject traffic which is considered suspicious or unauthorized



Components of a firewall

- Router
  - To monitor traffic into the private network
  - To reject access from unauthorized users, typically identified by IP address
  - To reject or reroute rejected packets, typically filtered by port number.
- Bastion host (or proxy server)
  - To provide a temporary store (cache) of pages held on a real web-server
  - Keep the identity of machines secret.

Firewall operation

- When a request for (e.g.) a web-page is received by the router from some client, it it accepted or rejected.
- If accepted, it is passed to the proxy server.
  - If the page is in the cache, it is sent to the client who requested it.
  - If the page is not in the cache, the proxy server requests a copy from the real web-server.
  - When this is received by the proxy host, it is then sent to the client.
- No direct contact between external machine and the web server.

How this protects you

- Any attacker can only reach the proxy server — e.g., deleting web-pages from the proxy just deletes them from the cache, not from the real web server.
- A stronger form of protection is a screened subnet. This has 2 layers of protection, with routers on each side.
- In the central zone, between the two routers, are proxy servers for various functions:
  - A proxy web-server
  - A proxy email server
  - ...

  Each proxy contacts a real server in the internal network
- The inner router can be set up to only allow traffic between the inner network and the central zone.

Types of firewall

- Packet-filtering router
  - Applies rules to each incoming packet and forwards or rejects them, one by one.
  - Filters may be on source or destination address fields.
- Application-level gateway
  - Clients attempt to use a specific TCP/IP application, such as Telnet or FTP.
  - To gain access past the firewall, they must enter access details (e.g., username and password).
  - If details are correct, the application is allowed to proceed; otherwise, not.

- Circuit-level gateway
  - This firewall does not permit an end-to-end TCP connection.
  - The firewall sets up two TCP connections:
    - One between it and the external client.
    - One between it and the internal client.
  - Messages are relayed across from one connection to the other.
  - Normally, message are not examined – the security occurs in setting up the connections.
- Typically, a bastion host is the platform for an application-level or a circuit-level gateway.

Limitations of firewalls

- Firewalls cannot protect against everything!!
  - e.g., disgruntled employees on the inside network
- Some attacks bypass the firewall
  - Internal systems may have a dial-out facility to connect to an ISP.
  - A modem that allows travelling employees to dial-in to the internal network remotely.
- Firewalls cannot screen every type of email or request
  - Viruses and malicious code may still get past a firewall, particularly if they are not presented in the usual form (e.g. email attachments).

Secure Sockets Layer (SSL)

- SSL was developed by Netscape for the Netscape Navigator browser.
- It operates at the levels between:
  - HTTP and FTP; and
  - TCP/IP
- This is what is operating when you connect using `https`.
- Provides several levels of security on the connection.
  - Why it gets used for commercial transactions.
- Widely adopted.

What SSL provides

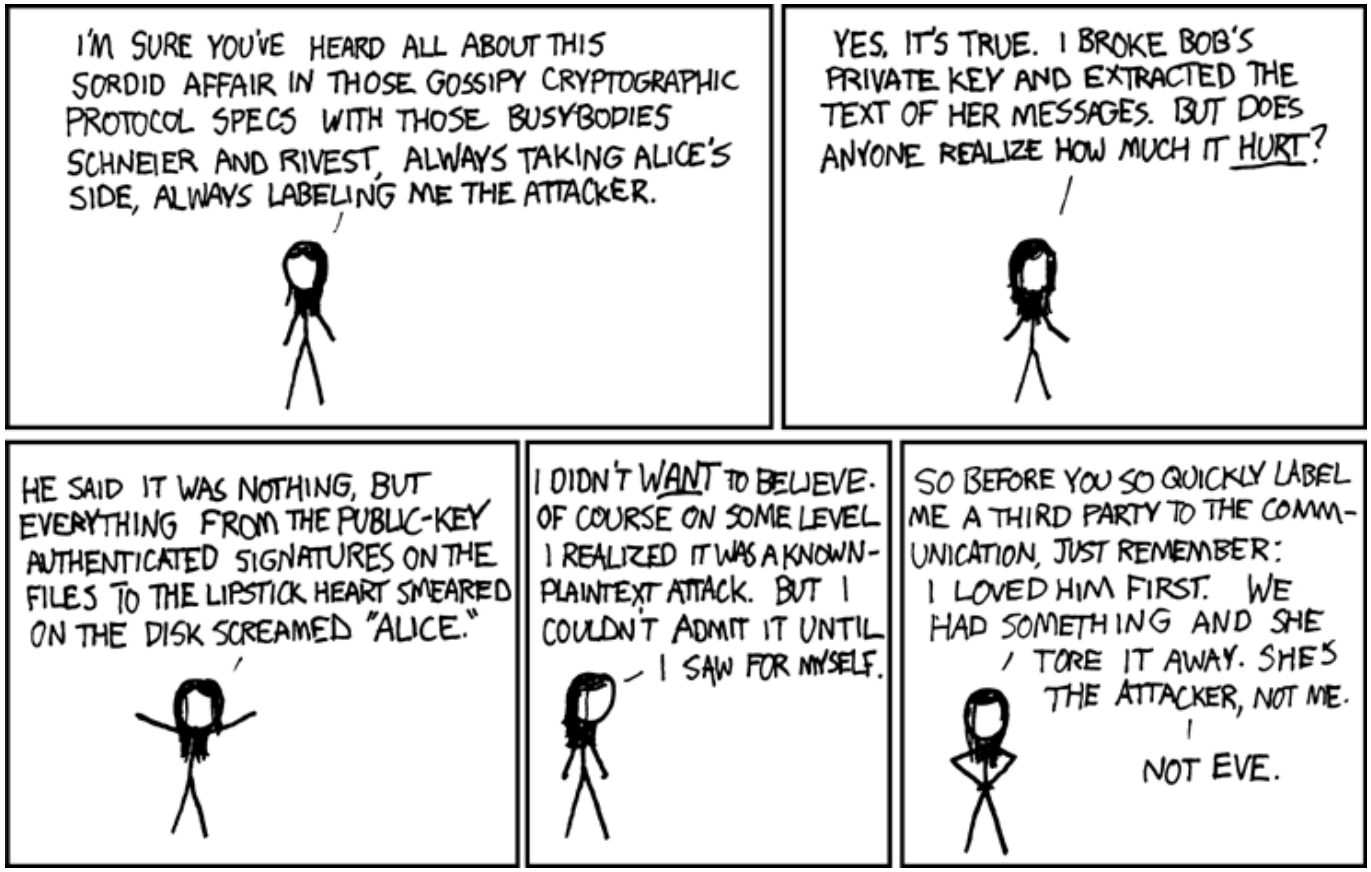
- SSL server authentication
  - Enables a client to confirm the identity of a server.
  - Uses public key cryptography to validate the digital certificate of a server and confirm that it has been issued by a valid certification authority.
- SSL client authentication
  - Enables a server to confirm the identity of a client
- SSL encryption
  - Uses symmetric encryption to send data to/from servers/clients.

- There are 2 sub-protocols
  - SSL Record Protocol — used for transmission of bulk data
  - SSL Handshake Protocol — used to establish the keys and algorithms to be used for data transfer

The SSL process

- Phase 1: Handshake
  - To authenticate server
  - To authenticate client (optional)
  - To agree secret keys and algorithms for part 2.
- Phase 2: Data transfer.
- SSL uses public key cryptography for the handshake, i.e.,
  - To authenticate client and server
  - To establish keys and algorithms for encryption of data transfer.
- SSL uses symmetric key cryptography for encryption and decryption of data in the data transfer.

Eve's story



- `http://xkcd.com/177/`

Summary

- This lecture wound up our look at net-centric computing.
- It dealt with the important topic of security.
- As we have done for many of the topics we've covered, we looked at:
  - General concepts
  - Specific support that Java provides.