Remarks on
Internet Security Issues
and Encryption Algorithms

Spring 99

Geoffrey Fox

Syracuse University

NPAC

111 College Place
Syracuse NY 13244 4100

3154432163

Abstract of Internet Security Presentation

- General Issues
- Review of Java Security Mechanisms
- “Gossip”: Examples of Security problems of various sorts from malicious to annoying
- Cryptography: including RSA Public Keys
- Authentication and Digital Certificates

Some Reference Material

- Web Security and Commerce, Simon Garfinkel and Gene Spafford, Prentice Hall June 1997
  - Lengthy and rather qualitative discussion that covers most important issues
- Network Security: Private Communication in a Public World, Kaufman, Perlman and Speciner, Prentice Hall 1995
  - fine discussion of underlying technologies but as “old” misses some key web concepts
- Java Security: Hostile Applets, Holes and Antidotes, McGraw and Felten, Wiley 1997
  - focuses on Java with a rather secretive qualitative discussion
- See references at http://www.sis.port.ac.uk/~mab/Computing-FrameWork/list.html including:
  - Globus, SET, SSL, Java, SESAME
  - http://www.cs.nps.navy.mil/curricula/tracks/security/notes/cs4601.notes.contents.html

Some General Issues I

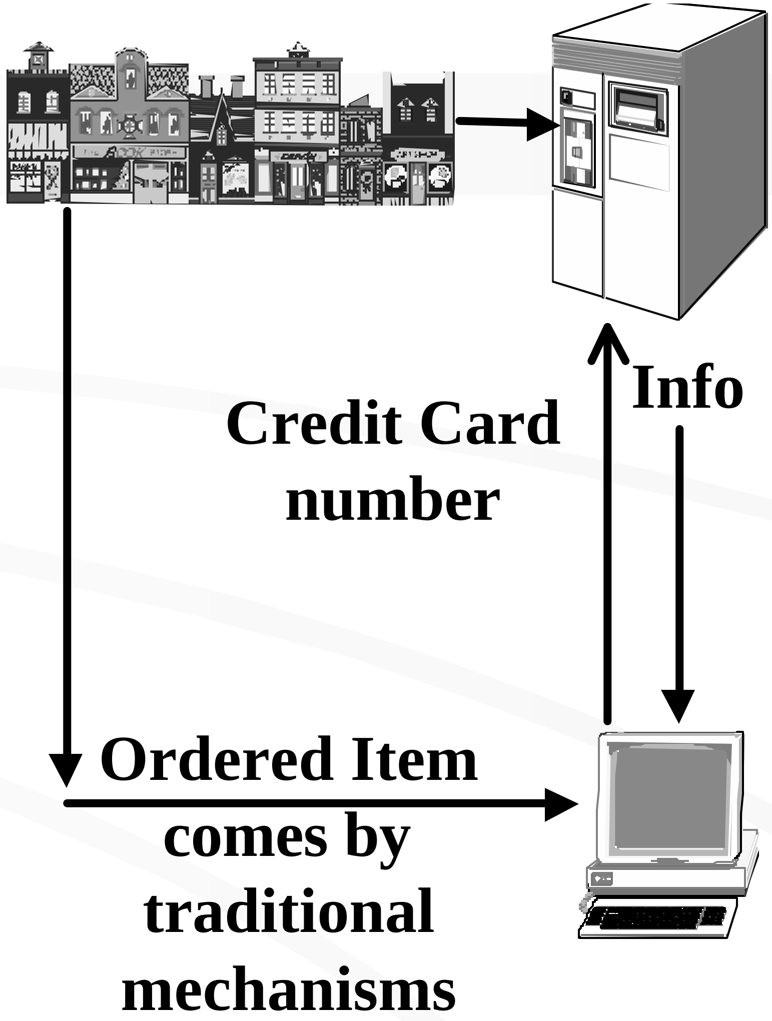
- Security Privacy Confidentiality and even Inconvenience are similar effects addressed by technologies we will discuss
- Why is there so much excitement about the (lack of necessary) security in Java?
- In fact Java is one of the few languages where both the language design (e.g. no pointers) and runtime (e.g. byte code verifier) explicitly address security as a major concern.
- Java’s great contribution was adding programmability to clients but this allows the “bad guys” to program various bad things such as viruses!

Some General Issues II

- **Java does have a very reasonable security model but it does not (did not) come with a complete implementation and so success of its security depends on additional capabilities.**
  - **Such as Server-Browser implementation of connections**
  - **The necessary extra stuff was (and is now to some extent) incomplete and had flaws (bugs in the software)**
  - **Authentication of creator of Java Applet**
- **The Internet is particularly fragile as one mistake/successful “terrorist attack” can impact computers over the whole world**
  - **The previous highways (interstate roadways) could only be attacked at isolated points (unless one used nuclear weapons) and so required less stringent security concerns**

Need for Security in Commerce - I

- Suppose we are buying something from information from an Internet Web Site
- Many of the hazards (security risks) are similar to those in mail or postal shopping with sometimes enhancement due to world wide nature of "attacks" and the ease with which digital (as opposed to analogue phones) information can be eavesdropped reliably.



Info

Credit Card number

Ordered Item comes by traditional mechanisms

Need for Security in Commerce - II

- There is the usual fraud opportunities at both client and server side (fake customer or fake store) and these are already present in non Internet version
  - Security can reduce level of fraud and make this a viable business model
- There is the possibility that merchant is attracting customers to site so that it can download applets and other Internet chicanery which can do bad things -- at least invade privacy
- There is the possibility that credit card number is snooped and this is directly countered using encrypted transmission
- Note we do not need a perfect system and commerce today builds in a certain fraction of loss due to fraud, bankruptcy etc.

Structure of Internet and Security-I

- **Information travels from server to client and back and one needs to discuss server,client and their connection.**
  - **Secure the server: here one needs to be worried about preserving confidentiality of data (different for different parts of information) and privileges/capabilities of CGI scripts**
  - **Scripting capability of Perl can be exploited in unwise CGI programs**
  - **User could input string "I am Geoffrey" or more deviously something like "I am";rm -r *;print "Pretty Evil" and the hidden program can delete files if the Perl CGI script unwisely applied eval(input string)!**
  - **A slightly more complex input can be dangerous with other Perl commands -- this can be circumvented by testing input for special characters**

Structure of Internet and Security-II

- **We need to secure information while it is travelling back and forth from server to client**
  - **If both server and client are in an institution, we can hope that network linking them is "secure" i.e. that data travelling back and forth cannot be diverted or eavesdropped by the bad guys.**
  - **Generally this is not a safe assumption and one needs to encrypt the data and provide authentication so that data cannot be read and you know where it comes from**
  - **Even in a corporation, one often has islands of relatively secure networks linked by insecure links such as the Internet**
  - **Technologies such as SSL (Secure socket Layer) implement encryption of messages between server and client**

Structure of Internet and Security-III

- **Finally we need to secure the client. Here Java is particularly important as it (and JavaScript) are the dominant downloaded programs**
- **Note clients are typically single user PC's with NO security and so particularly vulnerable to attack.**
- **Key difficulty is a bad guy developing a program that when downloaded does something you don't want**
- **In real world, we don't invite arbitrary people into our house -- rather we ask for credentials or believe by context (they are an adult accompanying your child's friend) that they are safe**
- **So we need both security in Java to check that code is what it purports to be and steps to establish confidence that what one is downloading is likely to be safe**

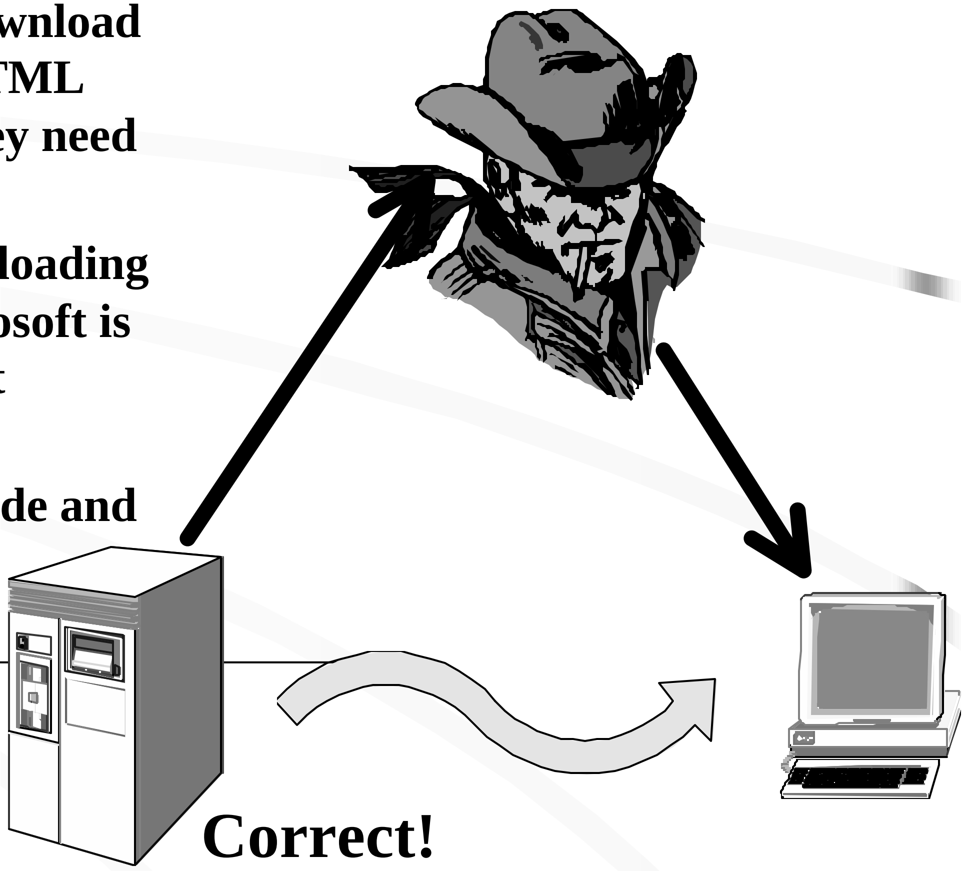
A PKZIP Anecdote

- In 1995 and 1996, a program called PKZIP30.EXE was placed on many Internet software libraries. This purported to be 3.0 beta release of the well known file compression program PKZIP
- Unfortunately, downloading this program, caused ones disk to be erased ……
- This is equivalent to a crook turning up at your door in a fake Niagara Mohawk (or what have you) van. In real world, if we are careful, we ask to see credentials of purported service person.
- In Web security, one needs digital signatures to establish the credentials of a particular program -- in particular one would expect that PKZIP30.EXE be digitally signed by PKWare the company that created PKZip
- Certification Authorities supply “Software Publisher’s Certificates” from “certification authorities” who presumably verify credentials of the organizations that they are certifying

Downloading Software is Dangerous?

- So Java applets are actually safer than downloading C C++ or Java Applications as applets cannot access the local disk (unless there is an implementation bug!)
- However Applets are so much easier to download as they happen automatically when the HTML page containing them is accessed. Thus they need much stronger security
- Note that one typically assumes that downloading from a site such as Netscape MIT or Microsoft is safe but this can be spoofed due to internet routing!
- Note that plug-ins are such C/C++/Java code and subject to security difficulties
  - A Macromedia Shockwave plug-in had a bug that allowed one to use it to read information on client computer and so violate (at least) confidentiality

Rogue Site substitutes Evil Program



Correct!

The Moldavia Pornographic Phone Scam

- The site sexygirls.com promised subscribers free pornography and all you had to do was to download the customized viewer.
- The “viewer” on being downloaded, turned off the modem’s speaker, disconnected you from the Internet and redialed to Moldavia. Then you did get the promised pornography but also a huge phone bill for international calls (presumably many dollars per minute)
- Moldavian phone company, the sexygirls website and the American phone split the proceeds of these bills!
- This is a “security hole” that exists today in phone system outside the internet

An Early Netscape DNS Bug

- Many of the famous Java security problems are in some sense “just bugs” and everything in society has bugs from car safety through conventional policing
  - Again Java bugs are more worrisome because they are potentially so widespread
- Currently Java is restricted to establishing a network connection to site you downloaded it from. This assumes you trust site and wouldn’t connect to iwanttodestroy.yoursystem.org.
- So in a Netscape2.0 bug, it was possible to set up applet so that it could connect to an arbitary site
  - Bug involved a malicious DNS server returning a set of IP addresses including allowed and disallowed ones. Netscape2.0 allowed one to connect to disallowed address
  - Now we have established a connection which could break through a firewall and in principle do arbitary damage/breach of confidentiality
- Netscape2.01 corrected bug by only allowing connection to original IP address

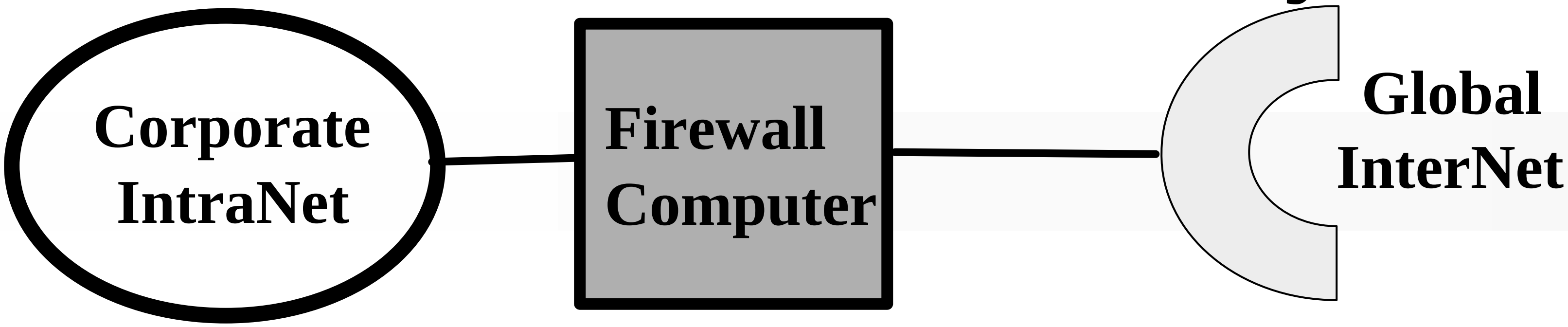
Tempest and Control Zones

- This refers to electronic eavesdropping and most information about this is classified
- Large classified computer centers are surrounded by a (earthed) conducting box so it all electronic signals are trapped inside
- Of course NSA (National Security Agency) tries to protect nation's security by exploiting inter alia ability to intercept and understand communication between unfriendly folks
- One classifies systems by their control zone which measures distance from device up to which one can detect signals
  - One wants a small control zone
- Internet and phone system rely on principle of mass confusion. There are so many messages that it is impossible to detect the sensitive ones!

Military Security Levels

- DoD classifies information into 4 levels
  unclassified < confidential < secret < top secret
- These classifications are applied to information in particular areas such as GUNS MIDDLEEAST etc.
- Documents are labeled by secrecy level and area
- Individuals and programs are given clearances such as (SECRET,{WEBTECH,HPCC})
- A human can only run a process with a security rating equal or below that of human
- A process can only read information marked with a security level equal to or below that of process
- A process can only write information information with a security level equal to or above that of process.

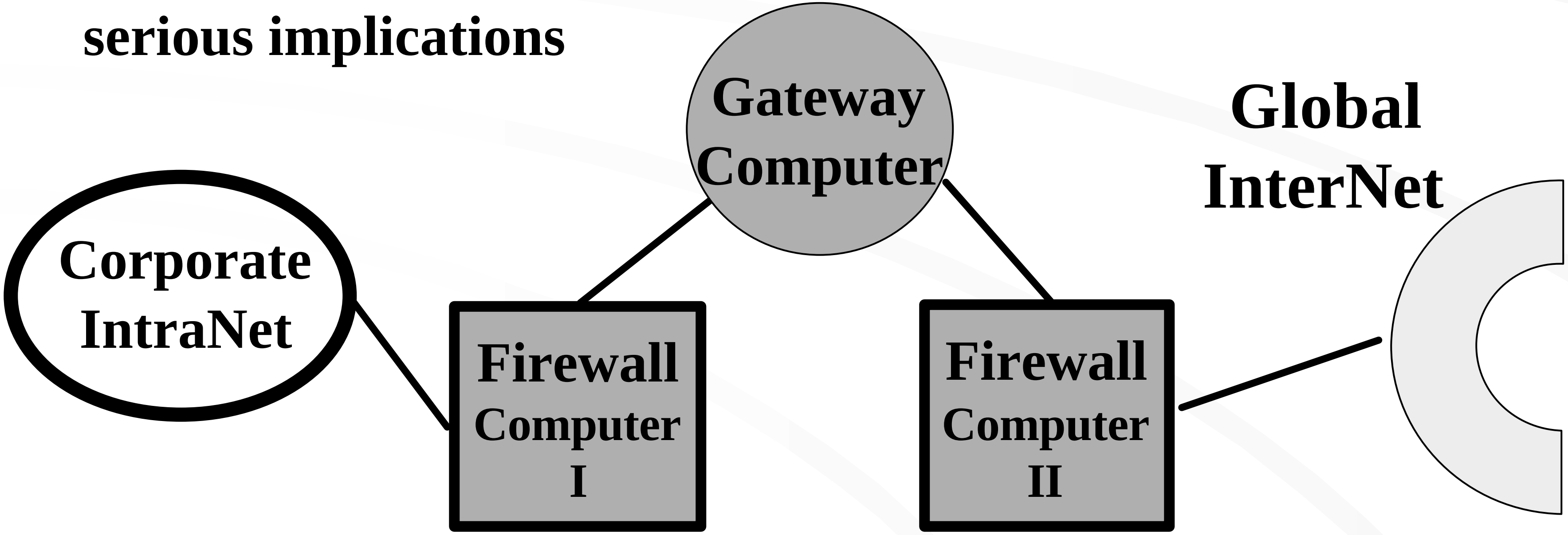
Firewalls and Gateways - I



- **A firewall is a computer that sits between an institutional network and the potentially dangerous insecure Internet/Outside network.**
- **Firewalls can be taught to filter information by address or by content**
  - **e.g. it could only allow inside-->outside messaging or messages to or from certain sites (this can be fooled by forging network addresses)**
  - **e.g. it can prevent file transfer but accept email**
  - **However email is actually a common file transfer mechanism and postscript (for instance) can easily hide rogue programs**
  - **Often firewalls are highly inconvenient and in mysterious ways handicap legitimate traffic when in fact nobody realizes firewall exists!**

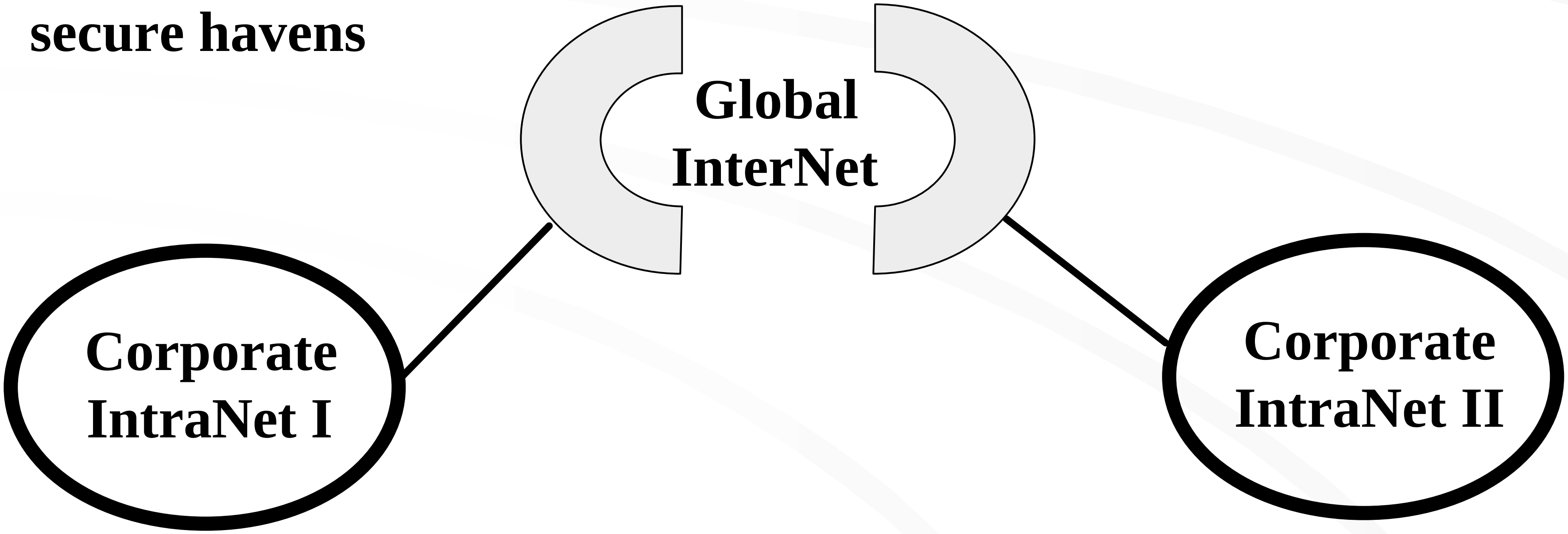
Firewalls and Gateways II

- Now we place a special computer (the gateway) which acts as a functional intermediary between secure IntraNet and Insecure InterNet
- Files are transferred back and forth by being deposited on Gateway. Gateway has security difficulties but it only has ephemeral information and can be compromised without serious implications



Encrypted Tunnels

- These are used to communicate between two or more secure IntraNets
- as opposed to previous discussions which was for Secure <--> Insecure communication
- This is a standard security problem addressed by using certificates to identify secure sites; no special action internal to IntraNet and use encryption over the insecure tunnel between two secure havens



The Great Clipper Controversy

- Unfortunately once we have set up digital security, the government will be severely handicapped in monitoring communication between sundry bad guys!
- They proposed that this be addressed by
  - a)Using a particular (and quite reasonable) encryption technology built into special hardware -- the clipper chip
  - b)The decryption keys used in each chip would be deposited with the government so that they and only they could break any encoding
  - c)The decryption key would actually be shared by two agencies “preventing” abuse by any one high-handed part of government
  - Note mainly aimed at phones and faxes but must address computer networking given growth in internet
- Highly controversial as clearly attacks various deeply held principles of free speech
- I am more or less certain that it is doomed anyway as too hard in today’s world to enforce such a standard

Export Restrictions on Cryptography

- The US government has established restrictions on the “quality” of encryption software that can be exported
- This actually translates into general restrictions on security quality as vendors do not want two types of software -- one domestic and one foreign!
- Note one can break “low quality” encryption e.g. RSA155 (best technology using 512 binary bits) would take 10,000 PC’s a few months to break
  - However the bad guys might to do this to break into Fort Knox but wouldn’t bother for your credit card number!
- Sun ingeniously released latest software using cryptography produced entirely outside the USA and so again the government is attempting something that is bound to fail!

Denial of Service versus “Attacks”

- There are two broad classes of “security” problems
  - 1) Deliberate attacks - NPAC suffered this 2 years ago when a hacker used a trapdoor to login into a Sun machine and remove all the files on three computers. For Java, these are called “attack applets” by Felten.
  - 2)Denial of Service: these include internet activities that stop desired use of your computers by hogging resources, invading your privacy or just annoying you! These are called malicious applets by Felten
    - NPAC suffered denial of service when a flood of unwanted “junk” email from forged addresses caused some of our servers to go down
    - I get annoyed when I or my children get unwanted email advertising pornographic sites
    - This is the basis of the recent “Melissa” virus

Combining Denial of Service with more Malicious Attack

- So a common strategy is to create a "denial of service" diversion
  - As NPAC has noted, you set up an agent which floods main enterprise Web Server with requests
  - Web Server grinds to a halt and all system people go to look at it and bring it up
- In the confusion, you mount a malicious attack -- perhaps on a different machine -- using well known bugs in UNIX ……..

Comments on Denial of Service

- **Examples of accidental denial of service include:**
  - **Mistakes in included JavaScript which cause infinite loops which can only be addressed by rebooting computer**
  - **One of my friends had a bug in his email script which sent 50 copies of his message**
  - **Use of too many graphics in your web pages uses all available network resources!**
  - **In November of 1996, an HTML page containing 100 tables within a table caused client computers to crash!**
- **Clearly such inconveniences are inevitable and can consequences can only be addressed by careful programming and robust operating systems which will reduce impact.**
  - **make certain all your key open files are “saved” before accessing unknown pages so that crashs do not destroy information!**

Some Attacking Concepts

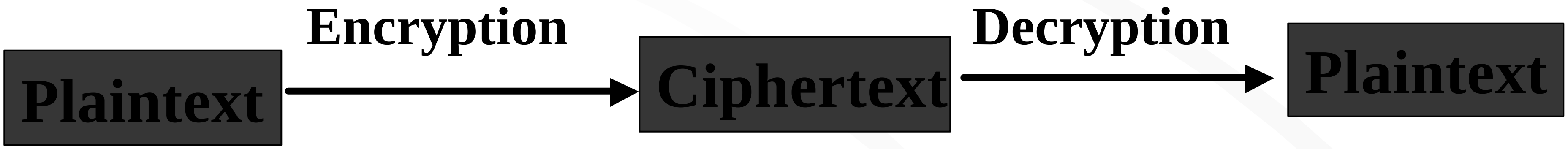
- Virus: A set of instructions that when executed inset themselves into other programs and presumably intend bad consequences.
- Bacterium: A free standing program that replicates itself and causes harm by consuming resources
- Worm: Similar to a bacterium that propagates over a network
- Trapdoor: an undocumented entry point which is written into a program often for debugging purposes
  - A virus can install trapdoors as it propagates
- Trojan horse: Instructions hidden in a seemingly useful program which can or do perform bad things. Viruses add Trojan horses to originally good programs
- Logic bomb: malicious instructions that trigger on some particular event or time in the future

Naïve way Viruses Spread themselves

- Take any good program (for which virus has write privileges) and take instruction at location L1.
- Replace this by a jump to L2.
- Insert the dreadful code at location L2 followed by original code at location L1. Worry about saving and restoring registers while doing this.
- Insert a jump to location L1+1 at end of bad code.
- Net result is a program that does all the old program did plus whatever else bad is inserted
- This naïve approach can be detected by presence of distinctive byte codes formed by code at L2 or more precisely by checking that a particular program has unexpected length or modify time.
- The hacker who entered NPAC installed a trapdoor into UNIX command ps in a way that left length of ps unchanged!
- First entered NPAC by "sniffing" somebody's password and using UNIX bugs to get root permissions.

Introduction to Cryptography

- **This is old technology first attributed to Julius Caesar who used the nifty cipher which replaced every letter with that three letters further in the alphabet**
  - **So A becomes D and Z becomes C (using cyclic wraparound)**
- **Most ciphers involve an algorithm and a parameter (this is 3 in the above) where usually algorithm can be public but parameter is kept secret and is called a key**
  - **key needs to be quite big to be safe (say at least 40 bits long)**
  - **It is usually not possible to keep algorithm secret and in fact making it public can encourage experts to examine and comment on its reliability (i.e. ease of breaking)**



Plaintext —Encryption→ **Ciphertext** —Decryption→ **Plaintext**

Breaking an Encryption Scheme

- **It would be easy to break Caesar's cipher but in general it is hard and in fact exponentially hard as breaking cipher involves some sort of combinatorial (exhaustive) search combined with clever ideas**
  - **such as looking for common English words such as the**
- **Note that encoding/decoding can be computationally expensive but much less so than code breaking**
  - **So as computers get faster, the code breaker's job gets easier for FIXED coding strategy but much harder for a coding strategy that takes a fixed time**
- **Methods exist for three scenarios**
  - **You only have ciphertext gotten by eavesdropping insecure link**
  - **You have some matched (ciphertext,plaintext) combinations and wish to find plaintext corresponding to some new ciphertext**
  - **You have ability to get ciphertext for any plaintext of your (the code breaker's) choice**

Types of Cryptographic Function

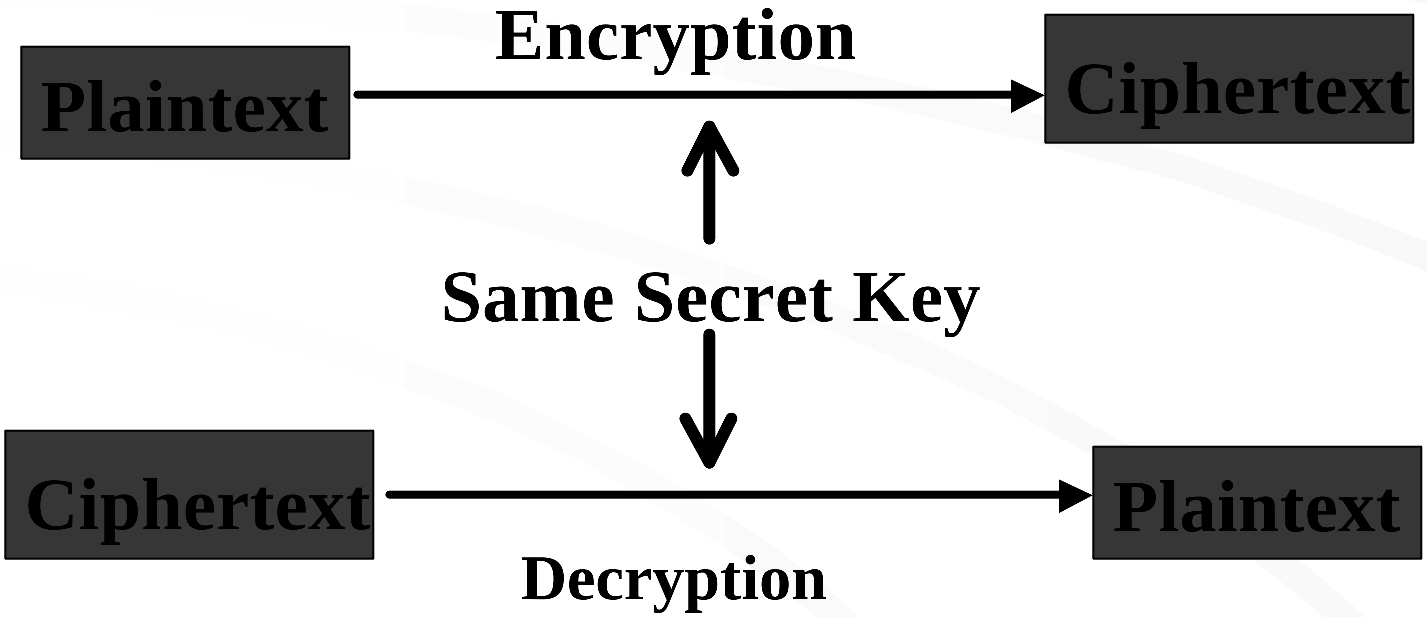
- There are three major types of cryptographic function which differ in functionality and performance
- Secret Key Cryptography is what we are most familiar with and has medium performance and functionality. It has one secret key
- Public Key Cryptography is a remarkable idea with some very important and non-intuitive capabilities. It is computationally intense and requires some infrastructure to implement. It involves one secret (called private) and one public key known to everybody.
  - Based on computational infeasibility of factoring large numbers which can be found by multiplying two primes
- Hash functions (or one way transformations) involve zero keys and encrypt in a way that cannot be decrypted. It is very fast
- Methods are combined to produce hybrid approaches that combine necessary speed and functionality

Security Uses of Cryptography

- 1)Transmitting over insecure channels
- 2)Storage on insecure media (essentially the same ideas as 1) but applied to a different need)
- 3)Authentication of computers or people at end of a message transmission. This includes digital signatures and password hashing
- 4)Integrity check that message delivered was the one sent (this is different from ensuring that nobody read information which is 1)). This is called message integrity

Secret Key Cryptography

- This involves a single secret key and standards are DES (Digital Encryption Standard) and IDEA (International Data Encryption Algorithm)
- Commercial systems (used in SSL) are RC2 RC4 RC5
  - – Note 40 bit RC2 takes a 64 MIP computer one year to break

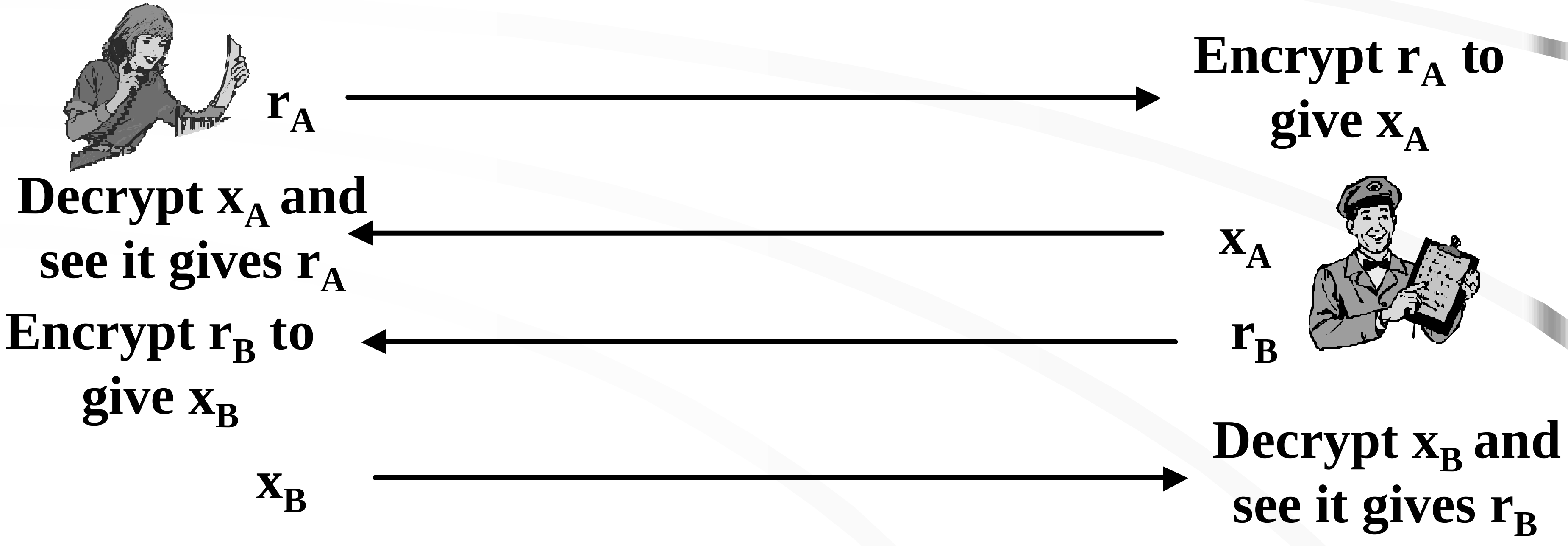


Uses of Secret Key Cryptography

- There is the natural use for either transmission over an insecure network or for storage on an insecure media.
- Strong Authentication implies that one can prove knowledge of a secret (key) without revealing the key and in particular without sending key between two individuals
- This is effective authentication but requires as many secrets as pairs of people who need to communicate.
- Public key version will only require one key for each individual wishing to be authenticated with anybody else and so is more practical for widespread deployment. N keys and not N^2 as for secret key authentication.
- Secret key authentication is however faster and much easier to implement for any sets of sites that wish to establish authenticated communication with a shared secret.

Secret Key Authentication

- Each individual A and B picks a random number r_A and r_B which are only known to themselves and a fresh for session to be authenticated. There is shared key K_{AB} which is not to be transmitted but A needs to know that B knows K_{AB} and B needs to know that A knows K_{AB}. The random numbers are known as challenges.



r_A

Encrypt r_A to give x_A

Decrypt x_A and see it gives r_A

x_A

Encrypt r_B to give x_B

r_B

x_B

Decrypt x_B and see it gives r_B

Message Integrity with Secret Key Cryptography

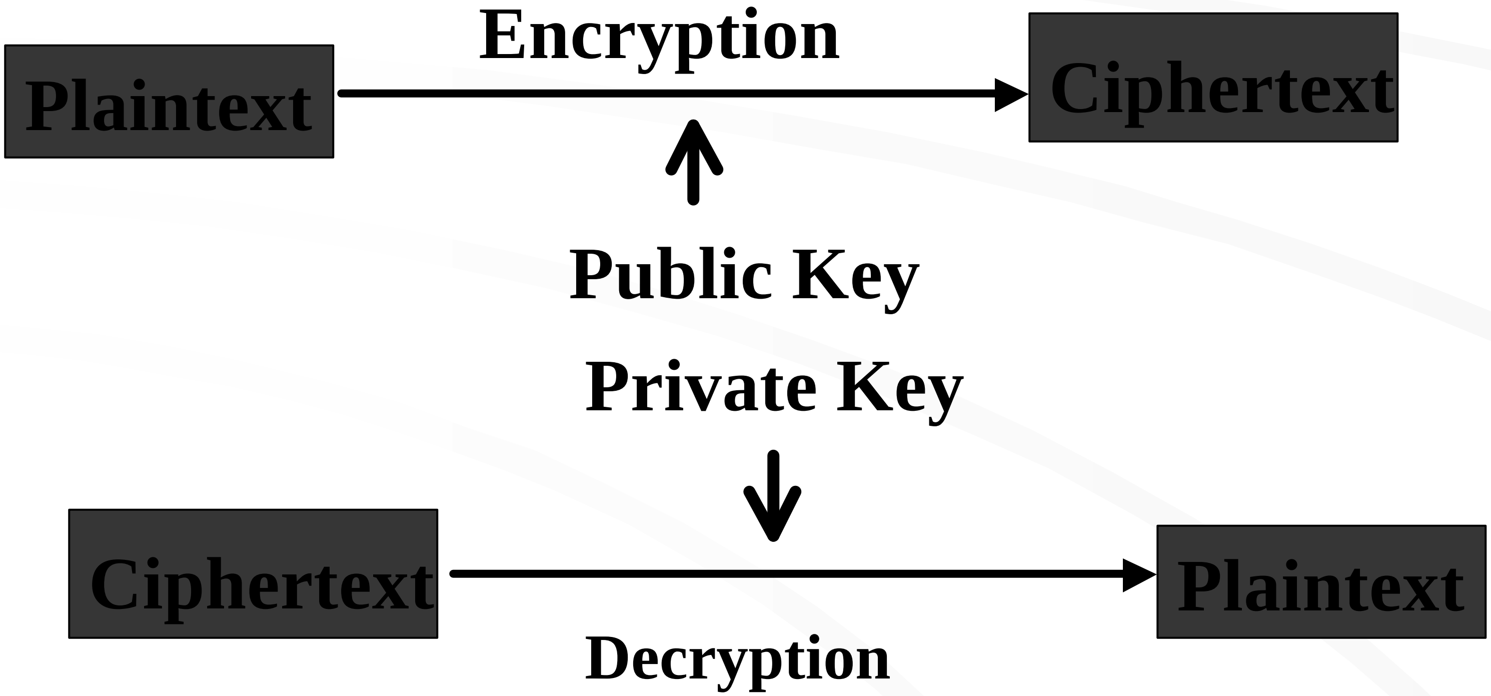
- Checksums are well known and can be gotten by dividing message into 32 bit groups and anding these groups together.
  - This is designed for fault tolerance and ensures that data was not garbled in transmission
  - hashes (designed properly) cannot be inverted and represent a unique fingerprint of original message.
- A secret checksum combines this process with a secret key and produces a MIC (message integrity code) which can be decoded and checked
- This can be used with either a ciphertext or plaintext message and guarantees that information is stored or transmitted faithfully
- Note encrypting a message does not guarantee that it is not changed!
- MIC with plaintext is used by bank electronic fund transfer

Public Key Cryptography

- This is much younger than other approaches and was first published in 1975. As we have discussed this has key feature of only needing one key per individual/organization requiring encrypted authenticated messaging
- It has nontrivial infrastructure to distribute the N public keys for N organizations but this is better than N^2 keys for secret key cryptography
- Roughly the public key is a very large number that is the product of two primes. The private key is (related to) one of these primes.
- It is used differently in two cases
  - Transmission over insecure network where one encodes with public key of receiver (and receiver decodes with their private key)
  - Authentication where you encode signature with private key and check the signature with public key

Insecure Link Transmission with Public Key Cryptography

- Suppose A has (public key,private key) <e_A,d_A> and B keys <e_B,d_B>
- A transmits a message m_A to B encrypting it with B's public key e_B
- B decrypts this message and reads it using private key d_B
- Similarly B sends a message m_B to A encrypting with e_A which A decrypts with private key d_A



Authentication with public key Cryptography

- Here A chooses a challenge -- a random number r and can verify that B is at other end using solely public information!

Encrypt r using e_B to give x

Send x

Decrypt x with d_B and send back challenge

Send r

Only B could have sent back r

Note this proves Bob is at other end without A knowing any secret information which could be eavesdropped

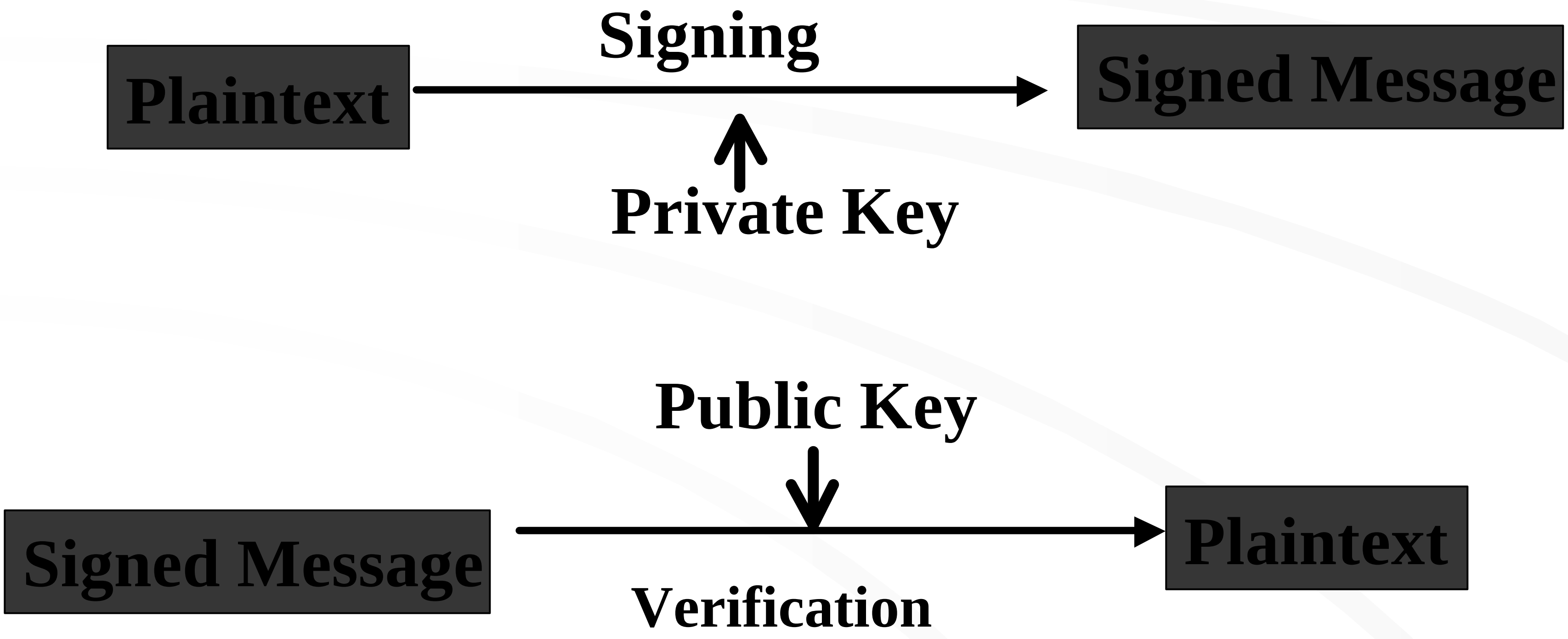
Secret key version requires A and B to have secret information

Alice is A

Bob is B

Digital Signatures and Public Key Cryptography

- **Digital Signatures reverse the use of public and private keys**
- **You encrypt with the private and decrypt with the public key**



Plaintext

Signing

Signed Message

Private Key

Public Key

Signed Message

Plaintext

Verification

Use of Digital Signatures with public key Cryptography

- Here B starts with a document that it is required to prove only could come from B
- This could be a piece of software that we wish to know comes from a reputable source
- We combine software with a “certificate” (a statement that B is Bob) and either encrypt this with d_B or more normally encrypt a message digest (that depends on both message and signature) with d_B
- This use of a message digest is done for performance as it is time consuming to use public key encryption on full message
- Note this signature cannot be forged either by A or any other person pretending to be B.
  - In secret key version A shares B’s secret key and can forge messages that purport to be from B

Hash and Message Digests

- Given a message m, the hash h(m) must satisfy
  - It can be calculated relatively quickly
  - Given h(m), it cannot be be inverted (to find m) by any practical method
  - Even though many m's will be transformed to the same h(m), this will in practice never happen and it is impossible in practice to find two m's that give the same h(m)
- As hash function is known, the security of a hash comes from the unknown message.
  - Messages can be made unknown by concatenating plaintext with a secret key before applying h(m)
- These are called one-way transformations as hashes cannot be inverted
  - Practical methods involve a strange combination of anding and permutations which ensures the cryptography safety of method
- Message Digests (such as MD2 MD4 MD5 -- MD is Message Digest with 128 bit output -- or SHS -- Secure Hash Standard with 160 bit output output) are used in Public key Systems to reduce computational complexity of encryption (see previous foil)

Some Math Behind Secret Key Cryptography

- Secret Key algorithms are based on elaborating a simple idea
- Caesar rotated alphabet in his cipher. An obvious extension of this is use a 1<->1 permutation of a group of N bits
- for DES N=64 and permutation is calculated from a 48 bit key
- To make decoding harder, this is done 16 times with different keys extracted from an original 56 bit secret
  - Note secrets in real world are usually generated randomly
- This strategy is combined with (ad-hoc) transformations to further obfuscate the process
- The full message must be divided into blocks before this and the method of running secret key cryptography on long messages is non trivial (but not very fundamental) as doing in 64 bit separate units would allow information to be freely shuffled!

Some Math behind RSA Algorithm -I

- RSA stands for inventors: Rivest Shamir and Adleman
- Take a number $n = p * q$ where p and q are primes
- Choose a “suitable” number e
- Public key is <e,n> and basic encryption algorithm takes message m to be encrypted and forms
  - $c = m^e \bmod(n)$
- Decryption involves private key d which is found so that
  - $d * e = 1 \bmod((p-1)(q-1))$
- Then $m = c^d \bmod(n)$
- As factorization is computationally infeasible (for n of 512 bits in length or more), this encryption cannot be broken.

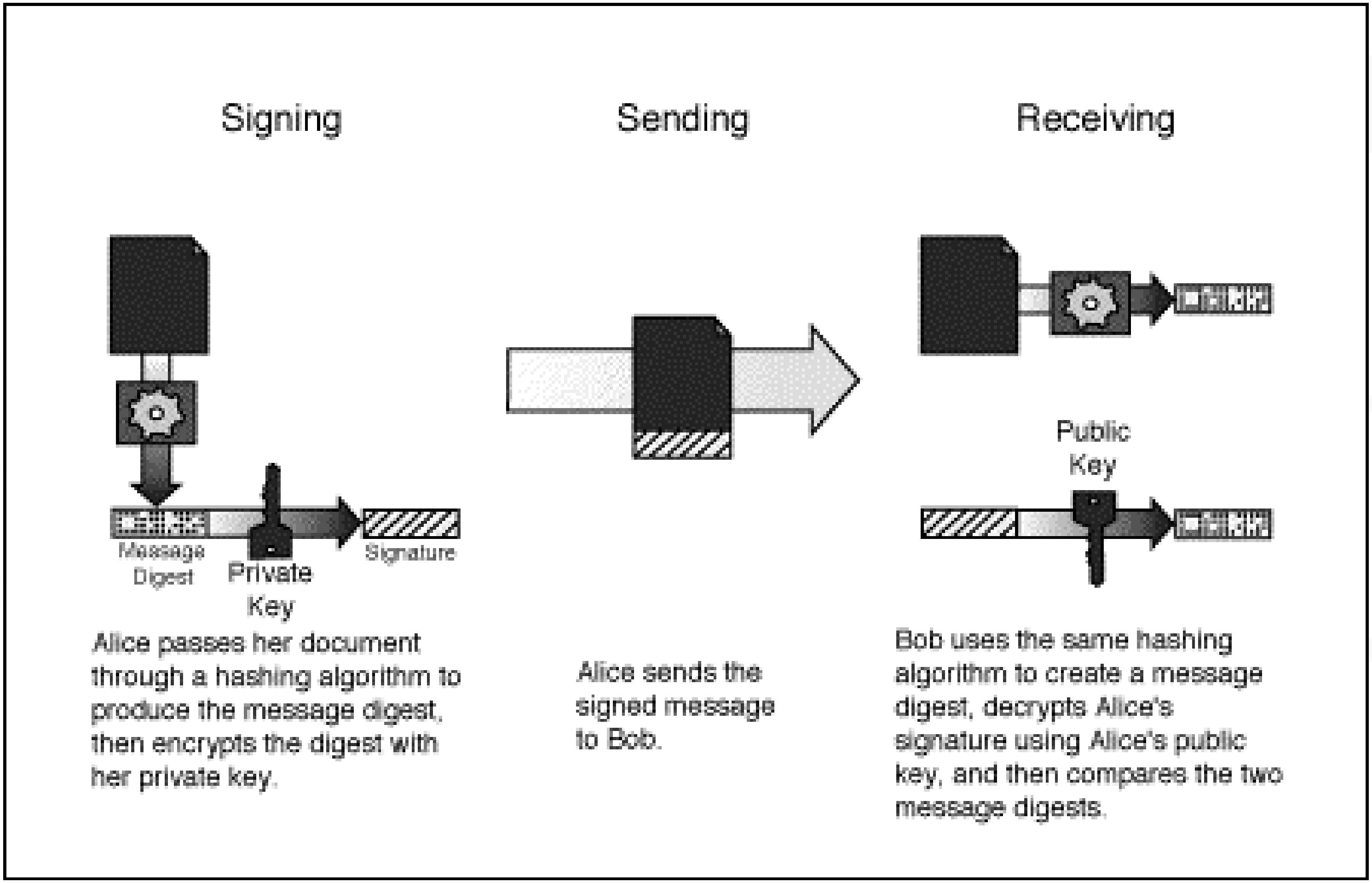
Some Math behind RSA Algorithm -II

- **n,c,d are 512 bits; p,q are 256 bits; e could be small (3 or 65537); m must be less than or equal to bit length of n**
- **lengths are doubled in recent implementations**
- **As encoding is time consuming, we only use RSA for small messages anyway. However as in secret key methods, one must in general break longer messages into smaller sizes**
  - **Deployed schemes use secret key methods (with key exchanged using public key method) for large amounts of data**
- **PKCS (Public Key Encryption Standard) is a standard from RSA for encoding the information to be signed or encrypted through RSA. It incorporates “know-how” to make RSA work reliably.**
- **Diffie-Hellman, El Gamal and DSS (Digital Signature Standard) are RSA like approaches aimed at digital signatures**

Certificate Authorities

- Certificates are essential for using asymmetric public-private keys
- RSA set up VeriSign company which offers web certificates for individuals, servers and software publishers
- There are many other authorities
- See http://digitalid.verisign.com/info_ctr.htm for a good description of certificates both in general and for Verisign services (http://www.verisign.com)
- Individual certificates are for use in Web browsers and for secure web mail(S/MIME) such as that offered by Netscape
- There is no agreed format for certificates but X.509 v3 certificate is common (PEM extends this)

Review of Certificate Process

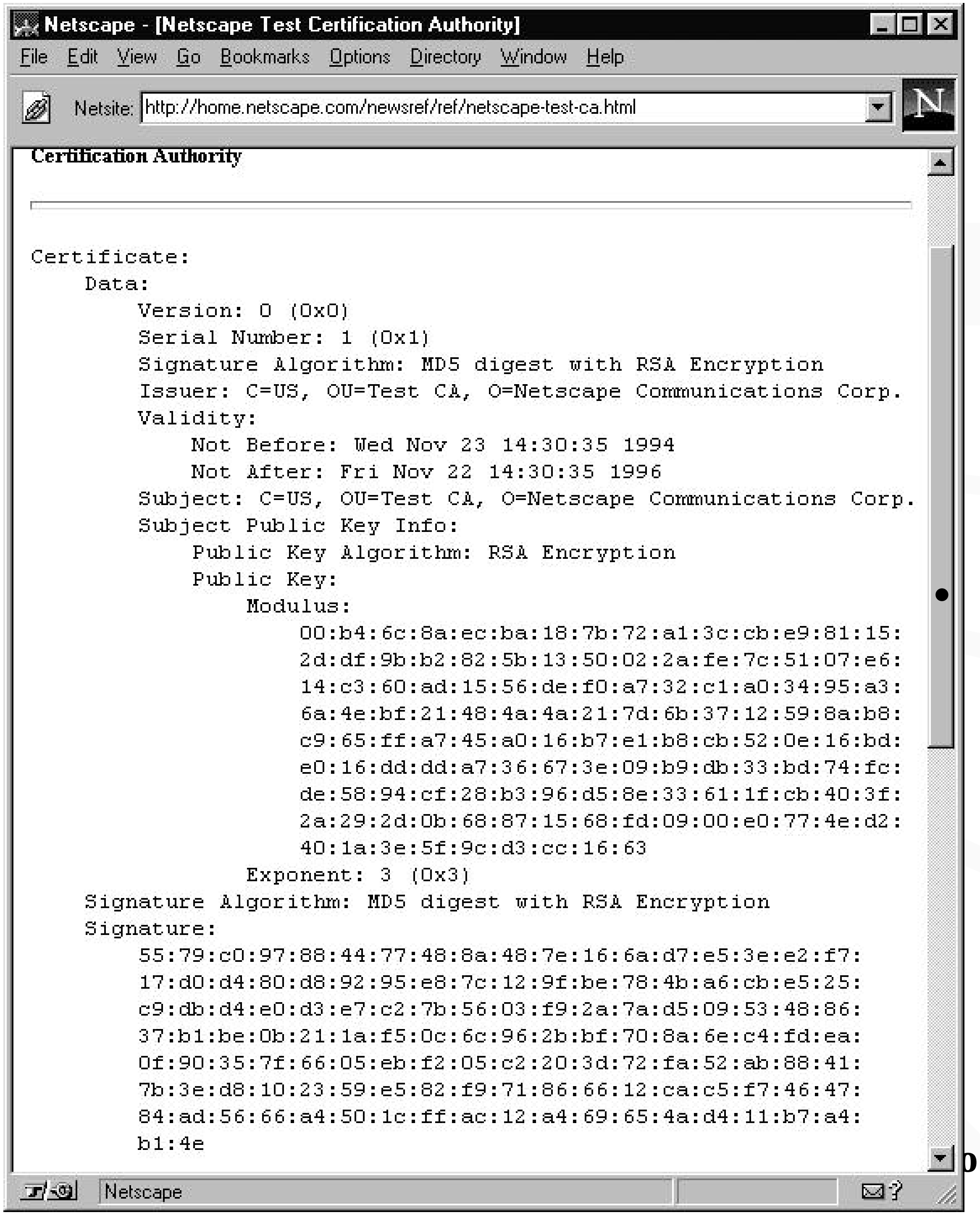


Alice passes her document through a hashing algorithm to produce the message digest, then encrypts the digest with her private key.

Alice sends the signed message to Bob.

Bob uses the same hashing algorithm to create a message digest, decrypts Alice's signature using Alice's public key, and then compares the two message digests.

Sample Certificate from Netscape


Netscape - [Netscape Test Certification Authority]
File Edit View Go Bookmarks Options Directory Window Help
Netsite: http://home.netscape.com/newsref/ref/netscape-test-ca.html
Netscape

Certification Authority

```
Certificate:
    Data:
        Version: 0 (0x0)
        Serial Number: 1 (0x1)
        Signature Algorithm: MD5 digest with RSA Encryption
        Issuer: C=US, OU=Test CA, O=Netscape Communications Corp.
        Validity:
            Not Before: Wed Nov 23 14:30:35 1994
            Not After: Fri Nov 22 14:30:35 1996
        Subject: C=US, OU=Test CA, O=Netscape Communications Corp.
        Subject Public Key Info:
            Public Key Algorithm: RSA Encryption
            Public Key:
                Modulus:
                    00:b4:6c:8a:ec:ba:18:7b:72:a1:3c:cb:e9:81:15:
                    2d:df:9b:b2:82:5b:13:50:02:2a:fe:7c:51:07:e6:
                    14:c3:60:ad:15:56:de:f0:a7:32:c1:a0:34:95:a3:
                    6a:4e:bf:21:48:4a:4a:21:7d:6b:37:12:59:8a:b8:
                    c9:65:ff:a7:45:a0:16:b7:e1:b8:cb:52:0e:16:bd:
                    e0:16:dd:dd:a7:36:67:3e:09:b9:db:33:bd:74:fc:
                    de:58:94:cf:28:b3:96:d5:8e:33:61:1f:cb:40:3f:
                    2a:29:2d:0b:68:87:15:68:fd:09:00:e0:77:4e:d2:
                    40:1a:3e:5f:9c:d3:cc:16:63
                Exponent: 3 (0x3)
    Signature Algorithm: MD5 digest with RSA Encryption
    Signature:
        55:79:c0:97:88:44:77:48:8a:48:7e:16:6a:d7:e5:3e:e2:f7:
        17:d0:d4:80:d8:92:95:e8:7c:12:9f:be:78:4b:a6:cb:e5:25:
        c9:db:d4:e0:d3:e7:c2:7b:56:03:f9:2a:7a:d5:09:53:48:86:
        37:b1:be:0b:21:1a:f5:0c:6c:96:2b:bf:70:8a:6e:c4:fd:ea:
        0f:90:35:7f:66:05:eb:f2:05:c2:20:3d:72:fa:52:ab:88:41:
        7b:3e:d8:10:23:59:e5:82:f9:71:86:66:12:ca:c5:f7:46:47:
        84:ad:56:66:a4:50:1c:ff:ac:12:a4:69:65:4a:d4:11:b7:a4:
        b1:4e
```

- **http://home.netscape.com/newsref/ref/netscape-test-ca.html**

VeriSign Digital ID’s or Certificates - I

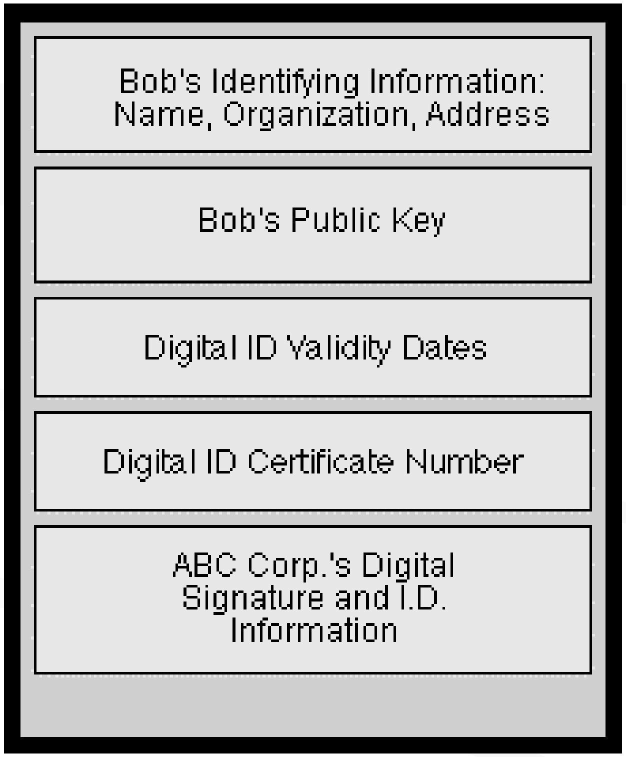
- VeriSign provides issuing, revocation, and status services for four types of Digital IDs --
  - Secure Server Digital IDs,
  - Digital IDs for software publishers,
  - Personal Digital IDs for use with Web Browsers, and
  - Personal Digital IDs for use with S/MIME applications.
- VeriSign Digital IDs for servers enable web servers to operate in a secure mode.
  - A VeriSign Digital ID unambiguously identifies and authenticates your server and encrypts any information passed between the server and a web browser.

VeriSign Digital ID’s or Certificates - II

- VeriSign software Digital IDs are used in conjunction with Microsoft Authenticode Technology (software validation) provide customers with the information and assurance they need when downloading software from the Internet.
- Personal Digital IDs used by individuals when they exchange messages with other users or online services.
- VeriSign offers four classes of personal Digital IDs. The classes are differentiated by their assurance level--the level of confidence that can be placed in the Digital ID based on knowledge of the process used to verify the owner's identity. The identification requirements are greater for higher numbered classes-
  - Class 1 Digital ID offers minimal assurance of the owner's identity,
  - while a Class 4 Digital ID offers assurance of not only the individual's identity, but also of that person's relationship to the specified company or organization.

VeriSign’s Description of Digital ID’s

- A Digital ID typically contains the:
  - Owner's public key
  - Owner's name
  - Expiration date of the public key
  - Name of the issuer (the CA that issued the Digital ID)
  - Serial number of the Digital ID
  - Digital signature of the issuer


Bob's Identifying Information: Name, Organization, Address
Bob's Public Key
Digital ID Validity Dates
Digital ID Certificate Number
ABC Corp.'s Digital Signature and I.D. Information

- The most widely accepted format for Digital IDs is defined by the CCITT X.509 international standard; thus certificates can be read or written by any application complying with X.509.
  - Further refinements are found in the PKCS standards and the PEM standard.

VeriSign’s Description of Certificate Revocation I

- A Certificate Revocation List (CRL) is a list of Digital IDs that have been revoked before their scheduled expiration date.
- There are several reasons why a key might need to be revoked and placed on a CRL.
  - A key might have been compromised.
  - A key might be used professionally by an individual for a company; for example, the official name associated with a key might be "Alice Avery, Vice President, NPAC."
  - If Alice were fired, her company would not want her to be able to sign messages with that key and therefore the company would place the key on the CRL.

VeriSign’s Description of Certificate Revocation II

- When verifying a signature, you can check the relevant CRL to make sure the signer's key has not been revoked if the signed document is important enough to justify the time it takes to perform this check.
- Certification Authorities (CAs) maintained CRLs and provide information about revoked keys originally certified by the CA.
  - CRLs only list current keys, since expired keys should not be accepted in any case; when a revoked key is past its original expiration date it is removed from the CRL.
  - Although CRLs are maintained in a distributed manner, there may be central repositories for CRLs, that is, sites on networks containing the latest CRLs from many organizations.
  - An institution like a bank might want an in-house CRL repository to make CRL searches feasible on every transaction.