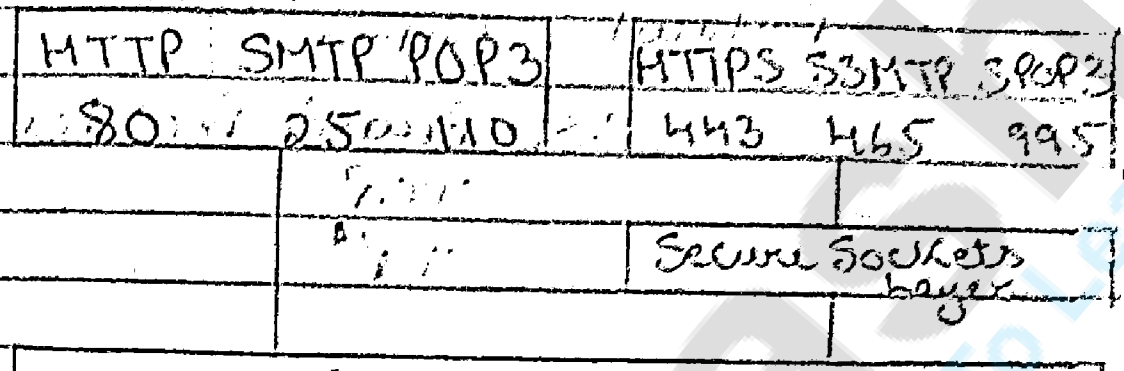
Unit 4

Q. SSL : (Secure Socket layer)

- transport layer service
- originally developed by Netsc-ape
- version 3 designed with public input
- subsequently became Internet standard known as TLS (Transport layer security)
- uses TCP to provide a reliable end-to-end service

• SSL has two layers of protocols

• where SSL fits



Uses Public key Scheme

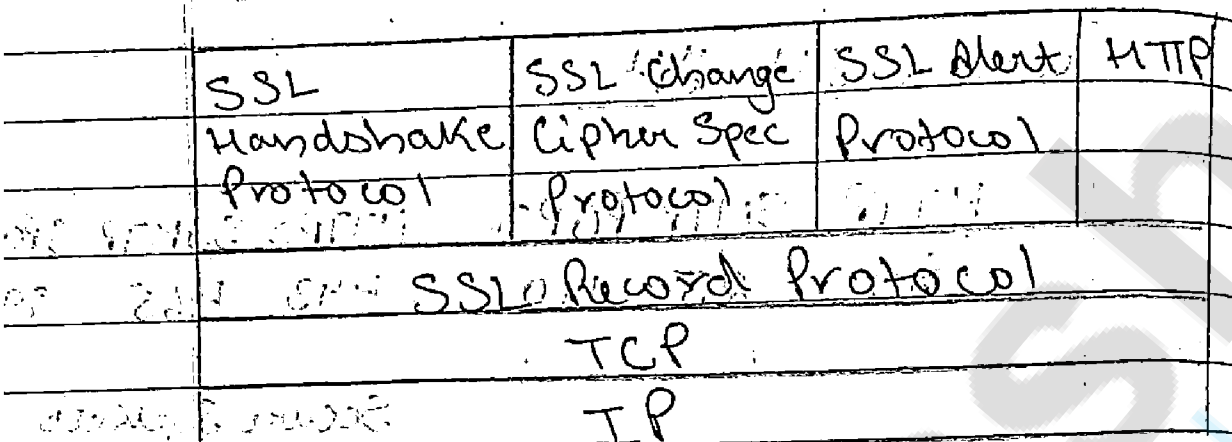
Each client-server pair uses pair of public keys

2 pairs one for client (browser) :-
created when browser is installed on client machine

one for server (http server)
created when server is installed on server hardware

2 private keys :-
one for client (browser)
one for server (http server)

SSL Architecture



SSL session
- an association between client & server
- Created by the Handshake Protocol
- define a set of cryptographic parameters which can be shared by multiple SSL connections
- Can be used to avoid expensive negotiation of new security parameters for each connection

SSL connection
- a transient, peer-to-peer, communications link
- associated with 1 SSL session

SSL Record Protocol
- confidentiality of

- using symmetric encryption with a shared secret key defined by Handshake Protocol
- IDEA, RC2-40, DES-40, DES, 3DES, Fortezza, RC4-40, RC4-128
- message is compressed before encryption

* message integrity
- using a MAC (Message Authentication Code) created using a shared secret key and a short message

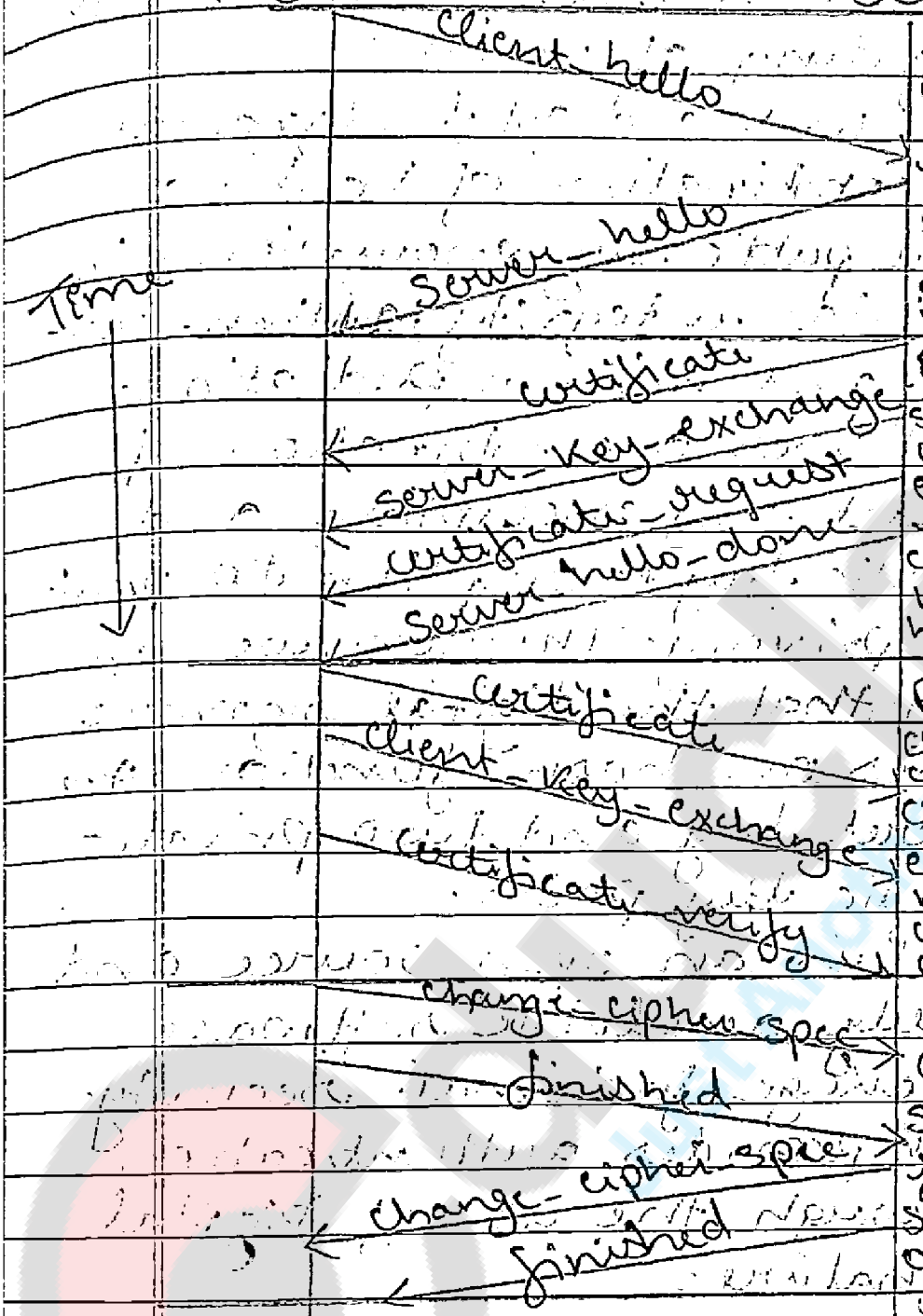
SSL Alert Protocol
- conveys SSL-related alerts to peer entity
- Severity: warning or fatal
- Specific alert: unexpected message, bad record mac, decompression failure, handshake failure, illegal parameter
- close notify, no certificate, bad certificate, unsupported certificate, certificate revoked, certificate expired, certificate

unknown
- compressed & encrypted
like all SSL data

SSL Handshake Protocol
- allows server & client to :-
1) authenticate each other
2) to negotiate encryption &
MAC algorithms
3) to negotiate cryptographic
keys to be used

- comprises a series of
messages in phases :
1) Establish Security Capabilities
2) Server Authentication
and key Exchange
3) Client Authentication and
key Exchange
4) Finish

SSL Handshake Protocol
Diagram



Q) PGP

PGP stands for Pretty Good Privacy. PGP was designed to provide all four aspects of security i.e privacy, integrity, authentication,

and non-repudiation in the sending of email.
PGP uses a digital signature (a combination of hashing and public key encryption) to provide integrity, authentication and non-repudiation.
PGP uses a combination of secret key encryption and public key encryption to provide privacy. Therefore, we can say that the digital signature uses one hash function, one secret key and two private-public key pairs.
PGP is an open source and freely available software package for email security.
PGP provides authentication through the use of Digital Signature.
It provides confidentiality through the use of symmetric block encryption.
It provides compression by using the ZIP algorithm, and GMAIL compatibility using the radix-64 encoding

PAGE NO.
DATE / /

1 scheme:

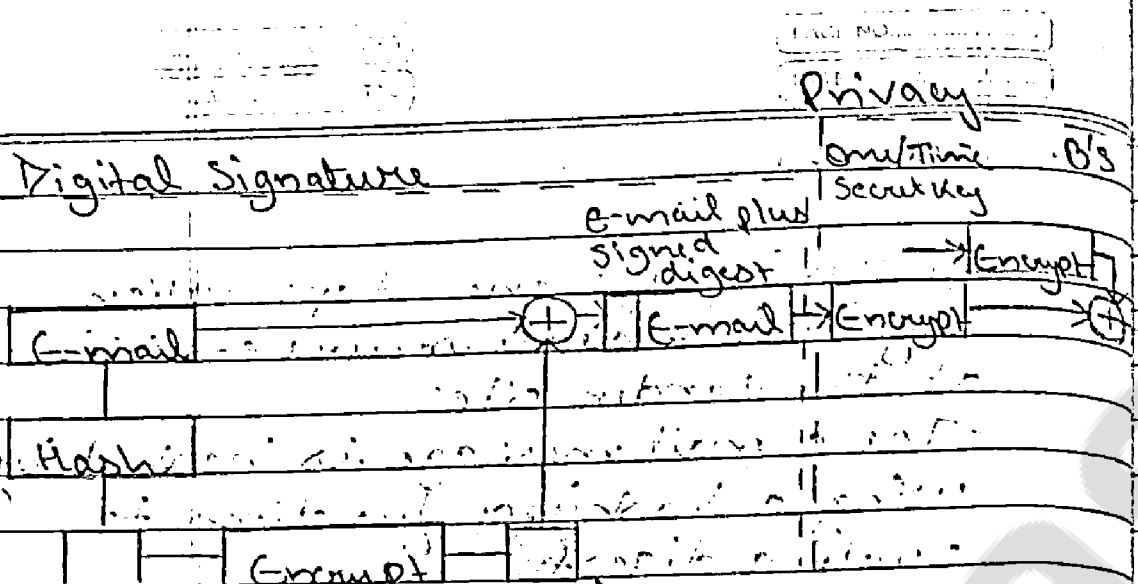
Following are the steps taken by PGP to create secure e-mail at the sender site:

- The e-mail message is hashed by using a hashing function to create a digest.
- The digest is then encrypted to form a signed digest by using the sender's private key, and then signed digest is added to the original email message.
- The original message and signed digest are encrypted by using a one-time secret key created by the sender.
- The secret key is encrypted by using a receiver's public key.
- Both the encrypted secret key and the encrypted combination of message and digest are sent together.

PGP at the sender site (CA)

Privacy

Digital Signature



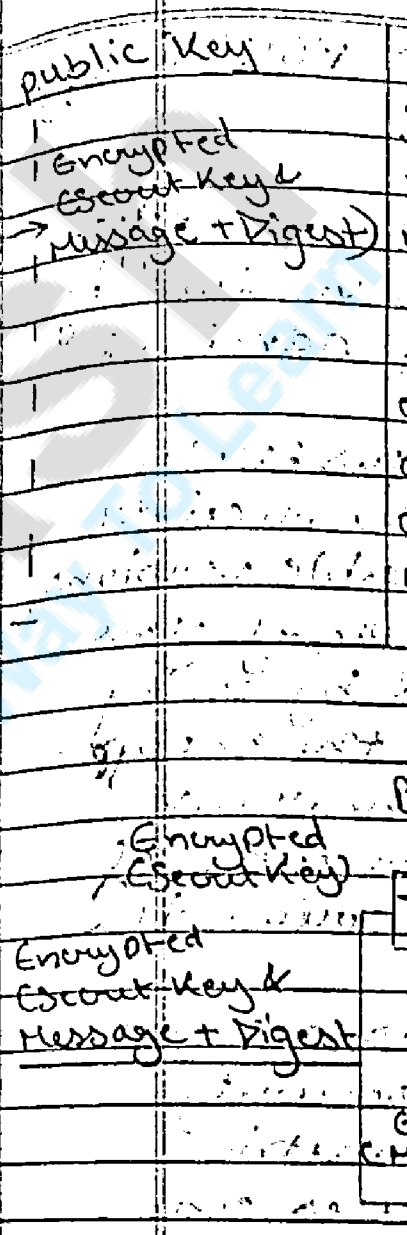
Following are the steps taken to show PGP uses hashing and a combination of three keys to generate the original message.

The receiver receives the combination of encrypted secret key and message digest is received.

The encrypted secret key is decrypted by using the sender's private key to get the one-time secret key.

The secret key is then used to decrypt the combination of message and digest.

The digest is decrypted by



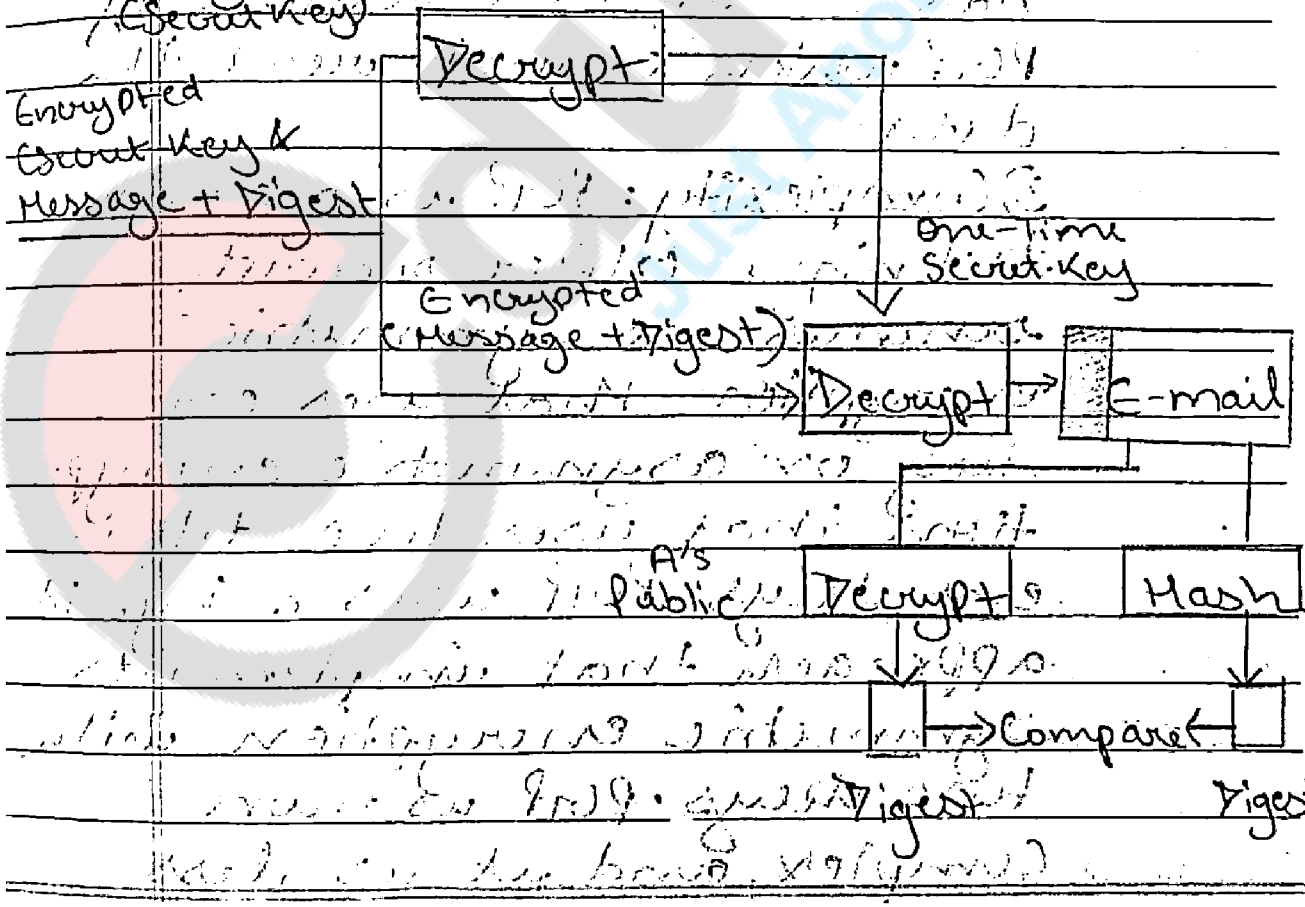
PAGE NO. .....
DATE ..../..../....

public key

Encrypted Secret key & (message + Digest)

using the sender's public key and the original message is hashed by using a hash function to create a digest. Both the digests are compared if both of them are equal means that all the aspects of security are preserved.

PGP at the Receiver site (B)



Disadvantages of PGP Encryption

1) The Administration is difficult :- The different versions of PGP complicate the administration.

2) Compatibility issues : Both the vendor and the receiver must have compatible versions of PGP. For example, if you encrypt an email by using PGP with one of the encryption techniques, the receiver has a different version of PGP which cannot read the data.

3) Complexity : PGP is a complex technique. Other security schemes use symmetric encryption that uses one key or asymmetric encryption that uses two different keys. PGP uses a hybrid approach that implements symmetric encryption with two keys. PGP is more complex, and it is less

functions than the traditional symmetric or asymmetric methods.

h) No Recovery: Encryption administrators face the problem of losing their passwords. In such situations, an administrator should use a special program to retrieve passwords. For example, a technician has physical access to a PC which can be used to retrieve a password. However, PGP does not offer such a special program for recovery; encryption methods are very strong so, it does not retrieve the forgotten passwords results in lost messages or lost files.

i) S/MIME (Secure/Multipurpose Internet Mail Extensions)
S/MIME is standard for exchanging secure mails with the help of encryption.
Previously, Mails were supposed

too carry text only
S/MIME provides support for varying content supported by major email programs like Outlook, Netscape

Working of S/MIME

Message sending mechanism:

Message body → Unique sender information captured → Unique recipient information retrieved → Signing operation performed

Digital Signature appended → Encryption operation performed → Encrypted message replaces original message → Message sent

Message receiving mechanism:

Message receiving mechanism

Message received → Encrypted message retrieved → Unique recipient information retrieved → Decryption operation performed → Decrypted message body retrieved

↓

Message body displayed ← Digital signature retrieved ← Unique sender information retrieved ← Digital signatures compared ← Message validated

Security Properties met by S/MIME

- Message confidentiality via encryption
- Message integrity via digital signature
- Message origin authentication via digital signature
- Non-repudiation of origin via digital signature

Security property not met by S/MIME

- Non-repudiation of receipt

S/MIME does not protect the sender of information against the denial of the receiver, who may say the sender never sent the information, or that he/she did not send it on time. Lack of this property prevents professional use of email.

S/MIME: Functions

Enveloped Data: Encrypted content and associated keys

Signed Data: Encoded message + Signed digest

Clear-Signed data : Clear text message + Encoded signed digest
Signed & Enveloped Data : Nesting of signed & encrypted entities

Q) Difference between S/MIME and PGP

From a user's perspective, S/MIME and PGP are different in the way a user obtains his keypair. In S/MIME the user has to obtain his key pair from a trusted Certificate Authority. And if someone wants to verify whether a public key is indeed the sender's authentic public key and is not forged by some attacker, he needs to verify it with the trusted authority and then use the key.

On the other hand, in PGP there is a concept of signing a key pair. Every user needs to sign his own key pair as well as of others with whom the user wants to communicate. Signing a key vouches for the authenticity of the public key. For example, if Alice is sure that a public key belongs to Bob and no one else, she would sign that public key. If another user Charlie wants to verify the authenticity of Bob's public key, Charlie can look at whoever has signed that particular public key. If Charlie knows Alice, he would be able to see that Alice has signed

the public key, which in turn would increase the trustworthiness of the key. Moreover, while verifying someone elses key, one can indicate his trust level on that key by specifying four levels of trust (full, marginal, none, unknown). So, one does not need any trusted central authority to verify a public key.
So, to summarize, both S/MIME and PGP use Public Key Cryptography, yet both are two different standards. The main difference is S/MIME depends on a centralized trusted authority for verification of public keys, but PGP does not need that.

(g) IPSec :-
One of the weaknesses of the original Internet protocol was that it lacked any sort of general purpose mechanism for ensuring the authenticity and privacy of data as it is passed over a network.
A set of protocols named Internet Protocols for Security (IPSec) were developed to provide security enhancements for internet critical applications.
There are two security modes for use depending on network need:
1) Transport mode:
Transport mode provides protection primarily for upper-layer protocols.

It is used to encrypt and optionally authenticate the data carried by IP.
Typically, transport mode is used to end-to-end communication between two hosts.

2) Tunnel mode:
It encrypts an entire IP packet.
After attaching all inner security header, the payload + security headers are treated as a new "payload" and a separate IP header is attached to it.
The entire original, or inner, packet travels through a "tunnel" from one point of an IP network to another; no routers along the way are able to examine the inner IP header.
Eg: VPN (Virtual Private Network)

Transport Mode: protect the upper layer protocols

Original IP Datagram	IP Header	TCP Header	Data

Transport Mode protected packet	IP Header	IPSec Header	TCP Header	Data

protected

Tunnel Mode: protect the entire IP payload

Tunnel Mode protected packet	New IP Header	IPSec Header	Original IP Header	TCP Header	Data
		↑ protected			↑

Two important protocols which are also referred as core IPSec protocols are:

IPSec Authentication Header (AH):
This protocol provides authentication services for IPSec.
What this means is that it allows the recipient of a message to verify that the supposed originator of a message was in fact the one that sent it.
It also allows the recipient to verify that none of the data in the datagram has been changed by any intermediate devices en-route.
It also provides protection against so-called "replay" attacks, where a message is captured by an unauthorized user and re-sent.

The Authentication Header consists of the following fields:
Next Header (8 bits): Identifies the type of header immediately following this header.
Payload length (8 bits): Length of Authentication Header in 32-bit words, minus 2.
Reserved (16 bits): For future use.
Security Parameters Index (32 bits):

Identifies a security association.
Sequence Number (32 bits): A monotonically increasing counter value.
Authentication Data (variable): A variable-length field (must be an integral number of 32-bit words) that contains the Integrity Check Value (ICV) or MAC

Bit: 0	8	16	31
Next Header	Payload length	RESERVED	
Security Parameters Index (SPI)			
Sequence Number			
Authentication Data (variable)			

Encapsulating Security Payload (ESP):
The Authentication Header ensures integrity of the data in datagram, but not its privacy.
When the information in a datagram is "for your eyes only", it can be further protected using the ESP protocol, which encrypts the payload of the IP datagram.

Bit: 0	16	24	31
Security Parameters Index (SPI)			
Sequence Number			
Payload Data (variable)			
Padding (0-255 bytes)			
Pad length	Next Header		
Authentication Data (variable)			

Confidentiality coverage
Authentication coverage

Security parameters Index (32 bits): Identifies a security association.
Sequence Number (32 bits): A monotonically increasing counter value; this provides an anti-replay function.
Payload Data (variable): This is a transport-level segment (transport mode) or IP packet (tunnel mode) that is protected by encryption.
Padding (0-255 bytes): - for various reasons
Pad length (8 bits): Indicates the number of pad bytes immediately preceding this field.
Next Header (8 bits): - Identifies the type of data contained in the payload data field by identifying the first header in that payload.
Integrity Check Value (variable): A variable-length field that contains the Integrity Check Value computed over the ESP packet minus the Authentication Data field.

Internet Key Exchange (IKE) is another important protocol involved in IPSec. It involves the determination and distribution of secret keys. A typical requirement is four keys for communication between two applications: transmit and receive pairs for both AH and ESP.

Q] Web Services Security:-

XML-RPC (Remote Procedure Call) :-

- XML-RPC is a simple protocol that uses XML messages to perform RPCs.
- Requests are encoded in XML and sent via HTTP POST.
- XML responses are embedded in the body of the HTTP response.
- XML-RPC is platform-independent.
- XML-RPC allows diverse applications to communicate.
- A Java client can speak XML-RPC to a Perl server.
- XML-RPC is the easiest way to get started with web services.

SOAP

SOAP is an XML-based protocol for exchanging information between computers.

- SOAP is a communication protocol.
- SOAP is for communication between applications.
- SOAP is a format for sending messages.
- SOAP is designed to communicate via Internet.
- SOAP is platform independent.
- SOAP is language independent.
- SOAP is simple and extensible.
- SOAP allows you to get around firewalls.

-SOAP will be developed as a W3C standard.

WSDL

WSDL is an XML-based language for describing web services and how to access them.

-WSDL stands for Web Service Description language.

-WSDL was developed jointly by Microsoft and IBM.

-WSDL is an XML based protocol for information exchange in decentralized and distributed environments.

-WSDL is the standard format for describing a web service.

-WSDL definition describes how to access a web service and what operations it will perform.

-WSDL is a language for describing how to interface with XML-based services.

-WSDL is an integral part of UDDI, an XML-based worldwide business registry.

-WSDL is the language the UDDI uses.

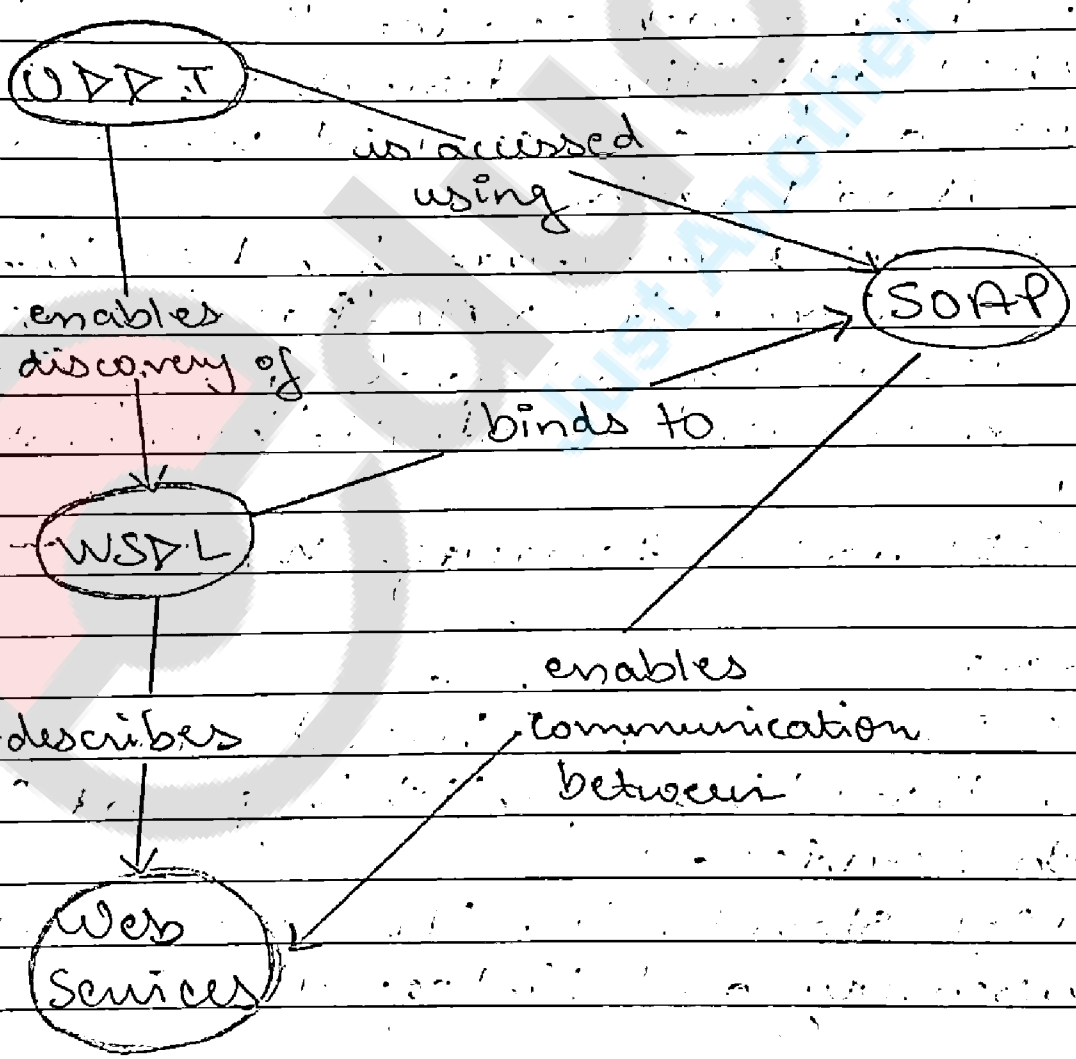
-

UDDI

UDDI is an XML-based standard for describing, publishing and finding web services.

-UDDI stands for Universal Description, Discovery and Integration.

- UDDI is a specification for a distributed registry of web services.
- UDDI is platform independent, open framework.
- UDDI can communicate via SOAP, CORBA and Java RMI Protocol.
- UDDI uses WSDL to describe interfaces to web services.
- UDDI is seen with SOAP and WSDL as one of the three foundation standards of web services.
- UDDI is an open industry initiative enabling businesses to discover each other and define how they interact over the Internet.



classmate
Date
Page

9] WS-Security

The aim of WS-Security is to ensure that communication between two parties is not interrupted or interpreted by an unauthorized third party. The receiver needs to be assured that the message was indeed sent by the sender, and the sender should be assured the receiver cannot deny receiving the message. Finally, the data sent during communication should not be altered by an unauthorized source. All data related to security is added as part of the SOAP header. Therefore, a considerable overhead is imposed on the SOAP message formation when security mechanisms are activated.

WS-Security SOAP Header:

The developer is free to choose any underlying security mechanism or set of protocols to achieve their goal. Security is implemented using a header which consists of a set of key-value pairs where the value changes appropriately with changes in the underlying security mechanism used. This mechanism helps to identify the caller's identity. If a digital signature is used, the header contains information about how the content has been signed and the location of the key used to sign the message. Information related to encryption is also stored in the SOAP header. The

ID attribute is stored as part of the SOAP header, which simplifies processing. The timestamp is used as an additional level of protection against attacks on the message integrity. When a message is created, a timestamp is associated with the message indicating when it was created. Additional timestamps are used for the expiry of the message and to indicate when the message was received at the destination node.

WS-Security Authentication Mechanisms

- Username/Password approach: The username and password combination is one of the basic authentication mechanisms used, and is analogous to HTTP Digest and Basic based authentication methods. The username token element is used to pass user credentials for authentication. The password can be transported as plain text or in digest format. When the digest approach is used, the password is encrypted using the SHA1 hashing technique.
- X.509 approach: This approach identifies the user by a public key infrastructure which maps the X.509 certificate to a particular user. More security can be added by using a public key and a private key to encrypt and decrypt the X.509

certificate to ensure that messages are not replayed, a time limit can be set to decline messages which arrive after a certain elapsed duration.

- Kerberos: The concept of a ticket forms the underlying mechanism of Kerberos: The client needs to authenticate with a key distribution centre (KDC) using a username/password combination or an X.509 certificate. On successful authentication, the user is granted a ticket granting ticket (TGT). Using the TGT, the client tries to access a ticket granting service (TGS). At this step, the first two roles of identification and authorization are over. The client then requests a service ticket (ST) to acquire a particular resource from the TGS and is granted the ST. The client uses the ST to access the service.
- Digital signature: XML signatures are used to protect the message from modification and misinterpretation. The signing must be performed by a reliable party or the real sender.
- Encryption: XML encryption is used to protect data from misinterpretation by making it unreadable to an unauthorized third party. Both symmetric and asymmetric approaches can be used.

WS-Security allows existing security

mechanisms to be leveraged appropriately to prevent any overhead in incorporating new mechanisms.

SAML

Security Assertion Markup Language (SAML) is a standard for logging users into applications based on their sessions in another context. This single sign-on (SSO) login standard has significant advantages over logging in using a username/password:

- No need to type in credentials
- No need to remember and renew passwords
- No weak passwords

Most organizations already know the identity of users because they are logged in to their Active Directory domain or intranet. It makes sense to use this information to log users in to other applications, such as web-based applications; and one of the more elegant ways of doing this is by using SAML. SAML is very powerful and flexible, but the specification can be quite a handful.

Working of SAML

SAML SSO works by transferring the user's identity from one place (the identity provider) to another (the

service provider). This is done through an exchange of digitally signed XML documents.

Consider the following scenario:- A user is logged into a system that acts as an identity provider. The user wants to log in to a remote application, such as a support or accounting application (the service provider). The following happens:

1) The user accesses the remote application using a link on an intranet, a bookmark, or similar and the application loads.

2) The application identifies the user's origin (by application subdomain, user IP address, or similar) and redirects the user back to the identity provider, asking for authentication. This is the authentication request.

3) The user either has an existing active browser session with the identity provider or establishes one by logging into the identity provider.

4) The identity provider builds the authentication response in the form of an XML-document containing the user's username or email address, signs it using an X.509 certificate, and posts this information to the service provider.

5) The service provider, which already knows the identity provider and has a certificate fingerprint, retrieves

the authentication response and validates it using the certificate fingerprint.
6) The identity of the user is established and the user is provided with app access.

Benefits of SAML Authentication

1) Standardization: SAML is a standard format that allows seamless interoperability between systems, independent of implementation. It takes away the common problems associated with vendor and platform-specific architecture and implementation.

2) Improved User Experience: Users can access multiple service providers by signing in just once, without additional authentication, allowing for a faster and better experience at each service provider. This eliminates password issues such as reset and recovery.

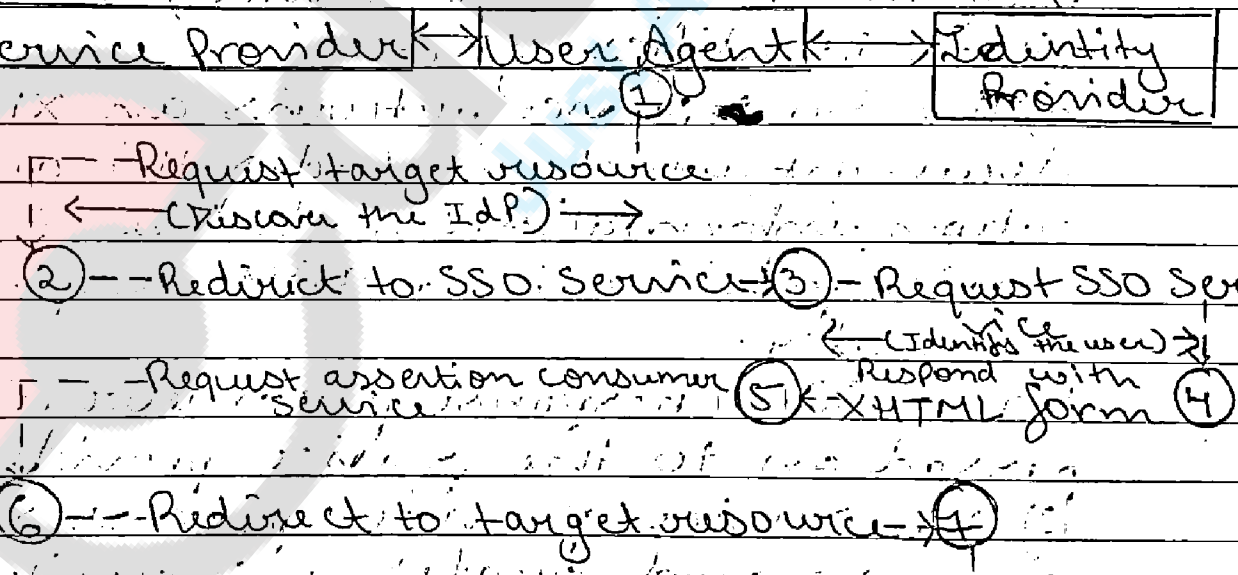
3) Increased Security: Security is a key aspect of software development, and when it comes to enterprise applications, it is extremely important. SAML provides a single point of authentication, which happens

at a secure identity provider. Then, SAML transfers the identity to service providers. This form of authentication ensures that credentials don't leave the firewall boundary.

4) Loose Coupling of Directories: SAML doesn't require user information to be maintained and synchronized between directories.

5) Reduced costs for service Providers: With SAML, you don't have to maintain account information across multiple services. The identity provider bears this burden.

Example:



The user agent would in most cases be a web browser

Imagine you're the user in an environment with single sign-on and you're trying to get access to some resource on a server. The sequence of events goes like this:

1) You try to access the resource on the server, which in SAML terminology is a service provider. The service provider in turn checks to see if you're already authenticated within the system. If you are, you skip to step 4; if you're not, the service provider starts the authentication process.

2) The service provider determines the appropriate identity provider for you and redirects you to that provider—in this case, the single sign-on service.

3) Your browser sends an authentication request to the SSO service; the service then identifies you.

4) The SSO service returns an XHTML document, which includes the authentication information needed by the service provider in a SAMLResponse parameter.

5) The SAMLResponse parameter is passed on to the service provider.

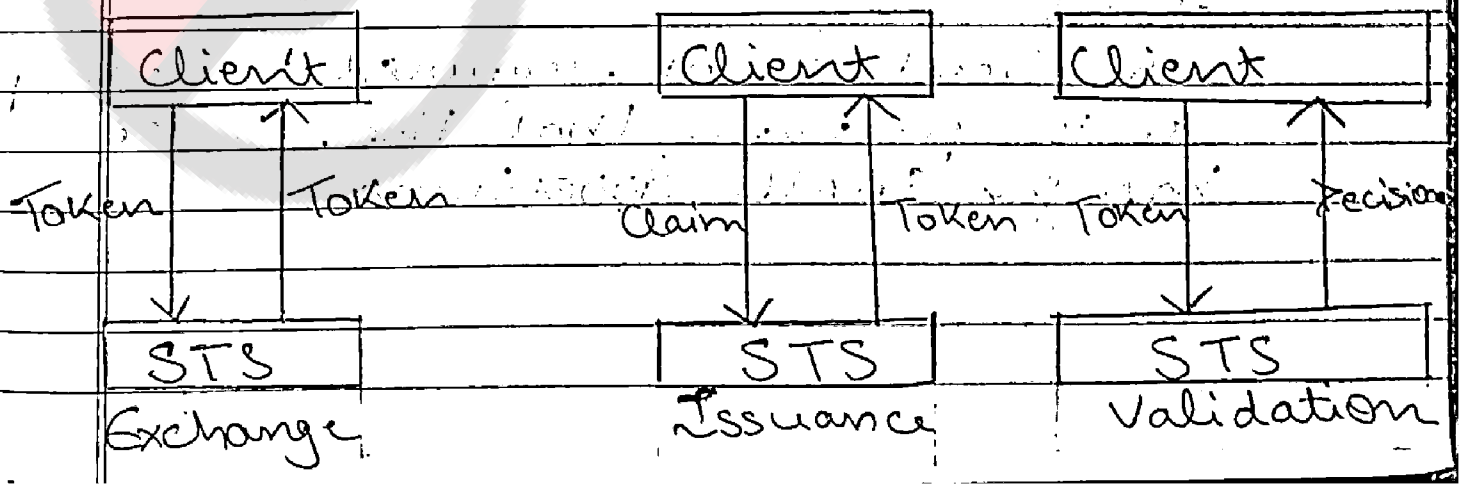
6) The service provider processes this response and creates a security context for you—basically, it logs you in—and then tells your requested resource it's

7) With this information, you can now request the resource you're interested in again.
8) The resource is finally returned to you!

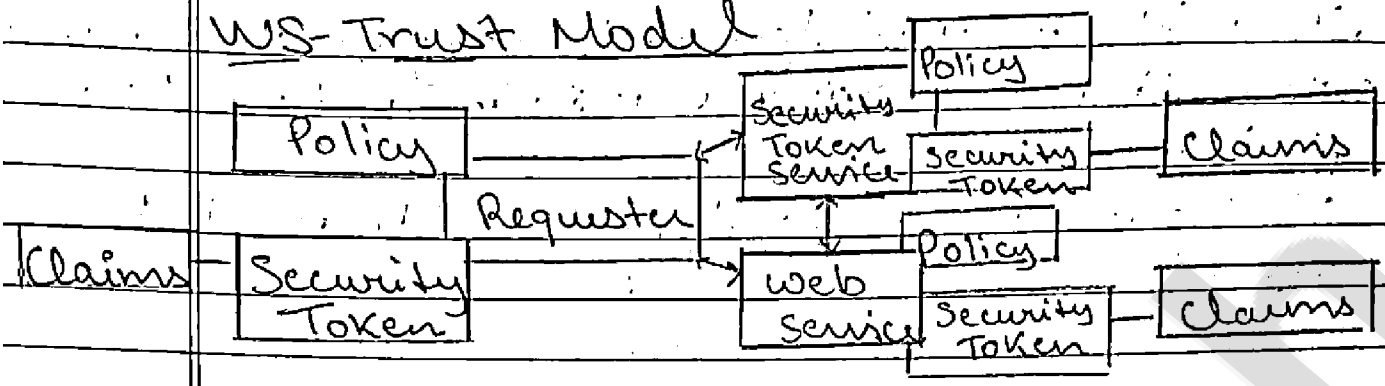
Q] WS-Trust
A SOAP message protected by WS-Security presents three possible issues with regards to security tokens:
1) Security token format incompatibility
2) Security token trust
3) Namespace differences

WS-Trust addresses these issues by:
Defining a request/response protocol
- Client sends RequestSecurityToken
- Client receives RequestSecurityTokenResponse

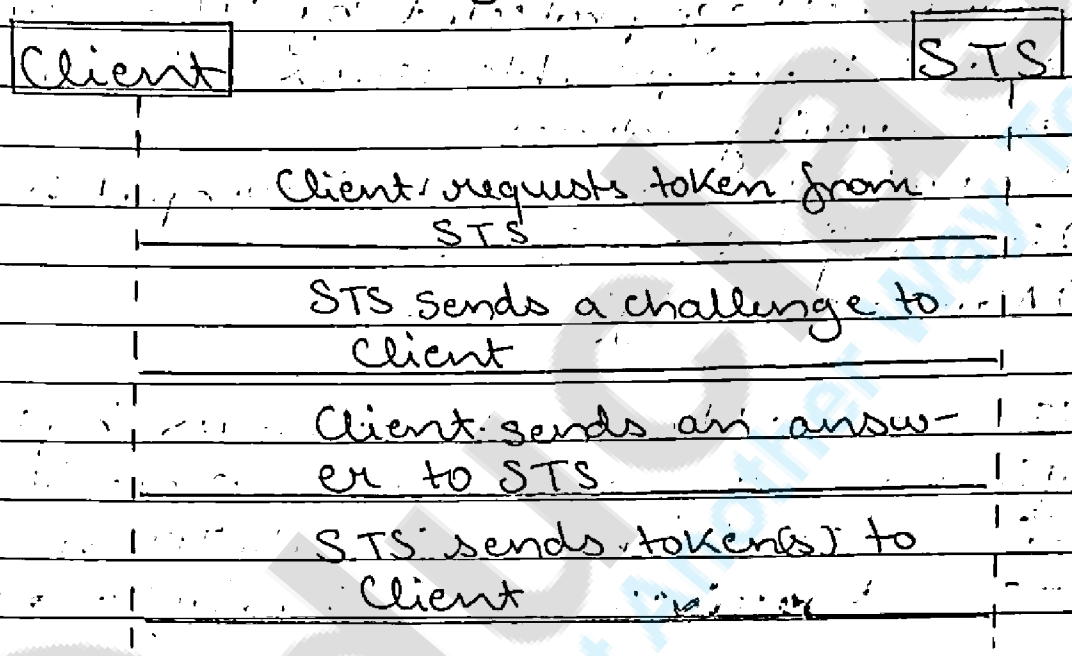
STS Functions
A Security Token Service allows:
1) Token Exchange
2) Token Issuance
3) Token Validation



WS-Trust Model

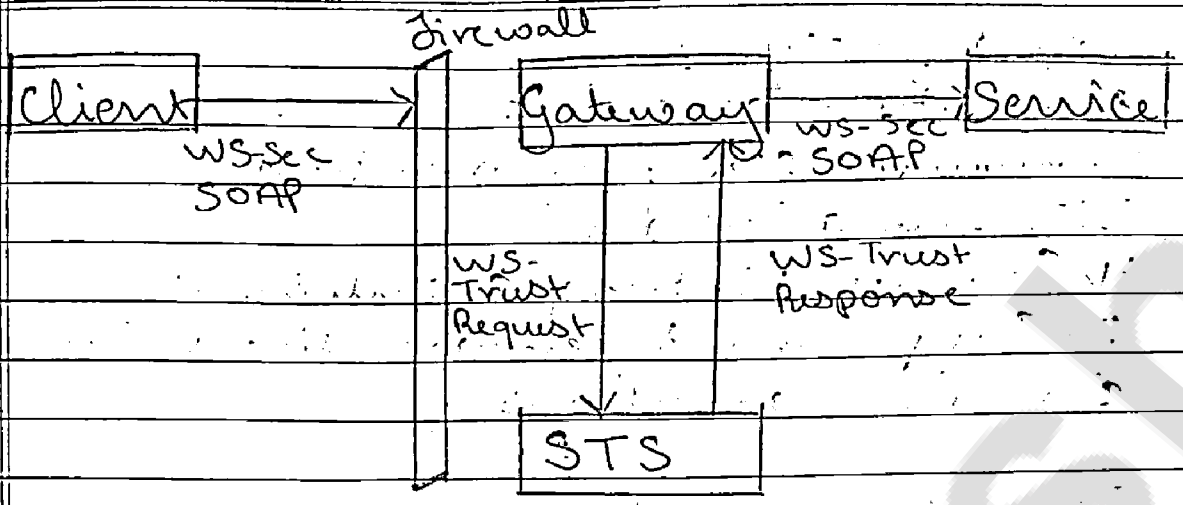


Request-Challenge Operation



WS-Trust Example

- Client understands X.509 certificates only
- Service understands SAML only
- The service does not directly trust the client
- The client is not required to anticipate the preference that the service has for SAML Assertions



The Security Assertions Markup language (SAML) is an XML-based framework for Web services that enables the exchange of authentication and authorization information among business partners. X.509 is a digital certificate standard, specifying ~~certe~~ certificate structure. Main fields are ID, subject field, validity dates, public key and CA signature.

Q) WS-Security Policy
- Based on WS-Policy
- Various groups of policy assertions
- Expressed in WSDL
- Defines six policy assertions that apply to WS-Security
- To express security requirements of a web service according to the WS-Policy spec
  - what needs to be protected
  - what token to use
  - Algorithms, references types etc

Assertion Types

1) Protection assertions
2) Required Elements Assertion
3) Token assertions
4) Security Binding assertions
5) Supporting token assertions
6) Protocol assertions

1) Protection Assertions

Specify what needs to be protected
- Integrity Assertions
- Confidentiality Assertions

SignedParts
This assertion specifies the parts of the message that need integrity protection

SignedElements
This assertion is used to specify arbitrary elements in the message that require integrity protection

EncryptedParts
This assertion specifies the parts of the message that need confidentiality protection

EncryptedElements
This assertion is used to specify arbitrary elements in the message that require confidentiality protection

2) Required Elements Assertion
This assertion is used to specify header elements that the message MUST contain

3) Token Assertions
Token assertions specify the type of tokens to use to protect or bind tokens and claims to the message
- Username Token
- X.509
- Issued Token

Username Token :- This element represents a requirement to include a Username Token
Issued Token :- This element represents a requirement for an issued token, that is one issued by some token issuer.
X.509 :- This element represents a requirement for a binary security token carrying an X509 token.

4) Security Bindings
A set of properties that together provide enough information to secure a given message exchange.
The bindings are identified primarily by the style of protection encryption used to protect the message exchange.
A binding defines the following security characteristics:
- The minimum set of tokens that will be used and how they are bound to messages.

- Any necessary key transfer mechanisms
- Any required message elements (eg timestamps)
- The content and ordering of elements in the wsse:Security header. Elements not specified in the binding are not allowed.
- How correlation of messages is performed securely (if applicable to the message pattern)

Security Binding Assertion
- AlgorithmSuite Assertion
- Layout Assertion
- TransportBinding Assertion
- SymmetricBinding Assertion
- AsymmetricBinding Assertion

Algorithm Suite :- This assertion indicates a requirement for an algorithm suite

Layout :- This assertion indicates a requirement for a particular security header layout

Transport Binding :- Indicates that the transport layer is used to satisfy the security requirements

Symmetric Binding :- Indicates that the message layer is used to satisfy the security requirements
Defines "Encryption Token" and "Signature Token" properties
When multiple messages are exchanged the tokens perform the same func-

tions for all messages.

Asymmetric Binding :- Indicates that the message layer is used to satisfy the security requirements.

Defines "~~Init~~ Initiator Token" and "Recipient Token" properties

The initiator token is used for the message signature from initiator to recipient, and encryption from recipient to initiator.

The Recipient Token is used for encryption from initiator to recipient, and for the message signature from recipient to initiator.

When multiple messages are exchanged the tokens perform different functions.

5) Supporting tokens

Services may require multiple sets of claims to be presented.

Corresponds to additional tokens in a message

Supporting tokens are included in the security header and may optionally include additional message parts to sign and/or encrypt.

6) Protocol :- Request/Response pairs for obtaining assertions.