Protocols Security

HTTPS, SSL, SSH, IPSEC

Issues

Need to secure web communications

Qualities of communication security

- Need for confidentiality: not read by a third party
- Need for integrity: no modification
- Need for authentication: identity of an object
- Need for non-repudiation: cannot deny one's actions

HTTPS

HTTPS stands for Hypertext Transfer Protocol Secure, which is an encapsulation of the HTTP protocol through the SSL protocol.

HTTP → SSL → HTTPS

Operation of the HTTP protocol

The HTTP protocol describes how a web browser queries a web server and requests the content of a URL.

→ Method + URL +Version http

Request Header

Request Body

Format of HTTP requests

Method

GET: Request to retrieve a resource located at the specified URL.

HEAD: This method only requires information about the resource, without requesting the resource itself.

POST: Sending data to the programme located at the specified URL (the body of the request can be used).

PUT: This method allows adding a resource on the server (sending data to the server).

DELETE: Deletion of the resource located at the specified URL.

URL: identifies the resource (Uniform Resource Locator), that is, the address of the page on the server.

Version:

version of the HTTP protocol (HTTP/1.1)

Request header:

to provide more information. It consists of field:value pairs.

Connection: Keep-Alive: do not close the connection after the response.

Accept: text/html, image/jpeg, image/png, text/*, image/*, */*: the client can receive all

Format of HTTP requests

Accept-Encoding: the client can receive all these encodings

Xgzip, xdeflate, gzip, deflate, identity

Accept-Language: the client can receive these languages

fr, en

Host: www.site.com

Body of the request:

To complete the request, the body of the request is sent. It can contain, for example, the content of an HTML form submitted via POST.

Format of HTTP requests

Example:

So the following request is:

http ://hypothetical.ora.com/

This causes the following message to be sent by the browser:

GET / HTTP/1.0

Connection: Keep Alive

User-Agent: Mozilla/3.0Gold (WinNT;I)

Host: hypothetical.ora.com

Accept: image/gif,image/xxbitmap,

image/jpeg,text/html

HTTP Response

An HTTP response is a set of lines sent to the browser by the server. It includes:

a status line:

it is a line specifying the version of the protocol used and the state of the request processing with a code and an explanatory text.

The line includes three elements that must be separated by a space:

The version of the protocol used

The status code which specifies whether the request was successful or if there was an error

The meaning of the code

Ex: HTTP/1.1 200 OK

HTTP/1.1 400 NOT FOUND

The version of the protocol is HTTP/1.1

HTTP Request Format

The most commonly used status codes are:

200: The HTTP request has been processed successfully.

201: The request has been processed correctly and resulted in the creation of a new resource.

202: The request has been accepted for processing, but the processing may not have completed successfully.

204: The HTTP server has successfully processed the request but there is no information to send back.

301: The requested resource has a new address (URL).

302: The requested resource resides temporarily at a different address (URI).

400: The web browser has made a conditional GET request and access is allowed, but the document has not been modified.

401: The request requires user authentication.

403 : The HTTP server understood the request, but refuses to process it.

Response header

This is a set of optional lines that provide additional information about the response.

Ex:

Date:

Tue, 22 Jun 2004 13:18:15 GMT

Server:

Apache/1.3.26 (Unix) Debian GNU/Linux PHP/4.1.2

mod_ssl/2.8.9 OpenSSL/0.9.6g DAV/1.0.3

information about the server

LastModified:

Tue, 22 Jun 2004 13:15:43 GMT

date of the last modification of the requested resource

The body of the response

The body of the response: composed of the lines of the requested document

Exp:

<html>

<body>

<h1> html page </h1>

 containing an image

and only one

<img src="balle.gif">

</body>

<html>

HTTPS Protocol

The purpose of HTTPS is to secure access to a web service in order to preserve its confidentiality.

Principle:

HTTP offers no guarantee of confidentiality; it is easy for a hacker to intercept requests and responses during these accesses.

Thus, we do not have certainty that we are consulting the site we believe.

HTTPS Protocol

HTTP:

HTTPS

The 'http' protocol is not subject to any encryption.

This is why the 'https' protocol for 'http' 'secured' was invented. When you browse a site or certain pages of sites, particularly transaction pages of merchant sites or banking sites, you will notice that the root of the page begins with 'https'.

HTTPS Protocol

In order to address these drawbacks, HTTPS is implemented.

It allows encapsulating and encrypting HTTP traffic, so it will be impossible for a hacker who intercepts this data to decrypt it.

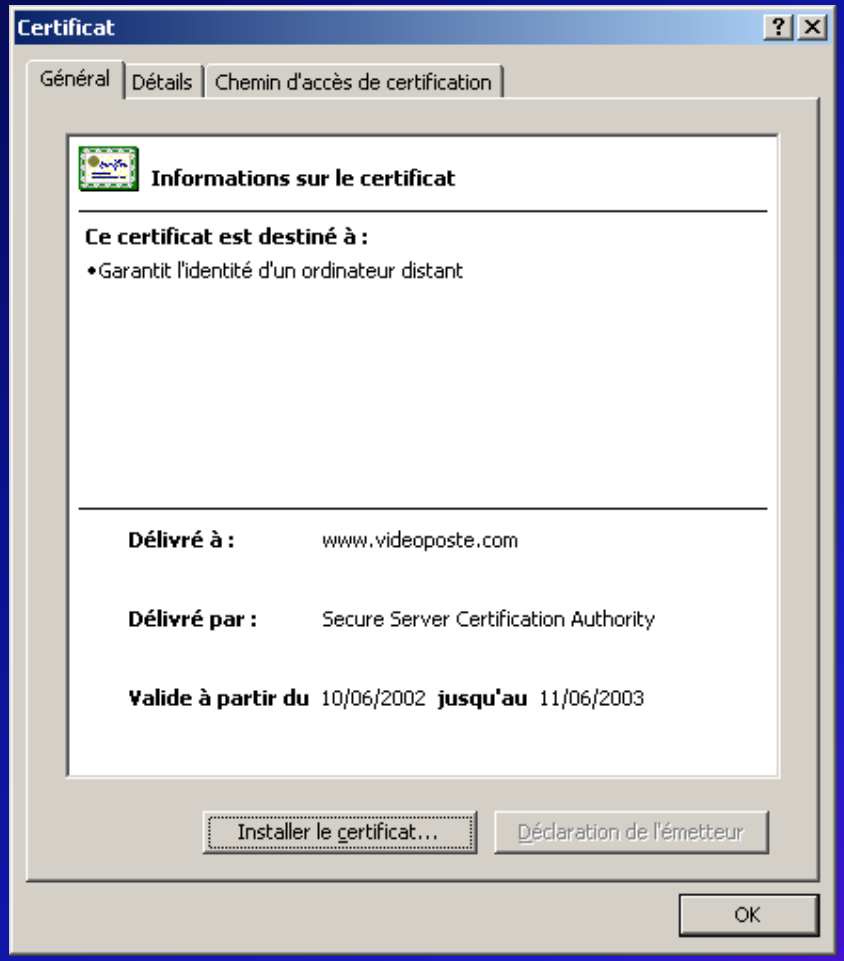
It ensures that the server you are accessing is indeed the one you think it is.

The exchanges are encrypted and decrypted using a pair of cryptographic keys that are unique to a W3 server.

Private key: known only by this server

Public key: known to the whole world

X509 certificates launch certmgr.msc



Advantages and disadvantages

The main advantages that HTTPS provides over HTTP:

- Data integrity
- Data confidentiality

Guarantee of having a trusted receiving host

The only real disadvantage is the requirement to encrypt all the data on the web page.

When it is said that the vast majority of browsers support HTTPS, this means that these browsers support SSL, indicated by a padlock in the address bar.

The SSL protocol (Secure Socket Layer)

General information: SSL = Secure Socke

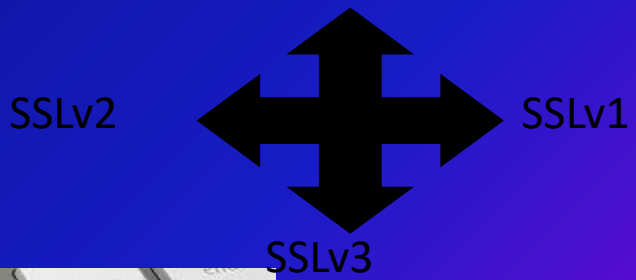
SSL is a system that allows information to be exchanged between two computers in a secure manner.

SSL ensures three things:

Confidentiality:

Integrity:

Authentication:



Characteristics

Independent of the communication protocol

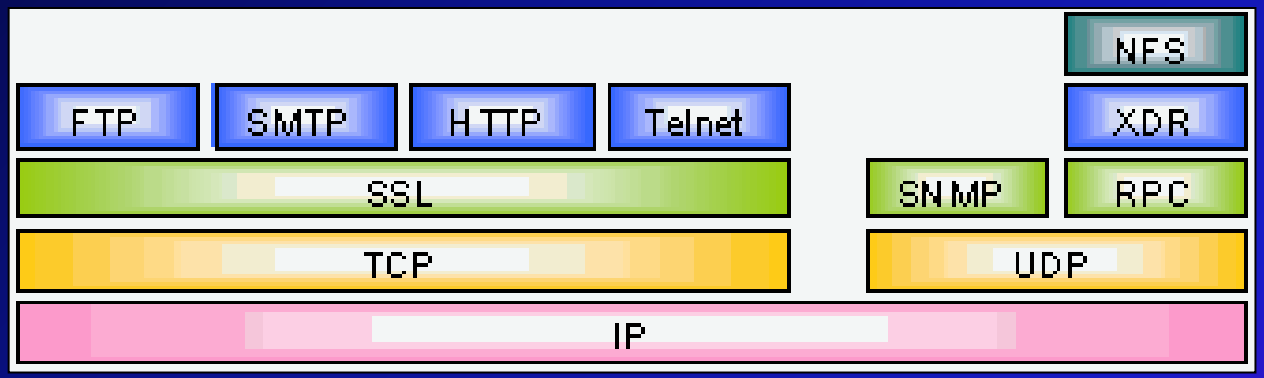
Follows the client/server model

Private and secure connection

Authenticated endpoints

Architecture

SSL is not a single protocol but it is located between the application layer and the transport layer



SSL operates over several TCP protocols: HTTP, LDAP, POP3, FTP.....

SSL consists of:

- Key generators
- Hash functions
- Encryption algorithms
- Negotiation and session management protocols
- Certificates

Why use SSL rather than another system?

- **SSL is standardised.**
  - There is a free version of SSL: OpenSSL (http://www.openssl.org) that you can use in your programs without paying royalties.
  - **OpenSSL is open source: anyone can audit and verify the source code (The secret lies in the encryption keys, not in the algorithm itself).**
- **SSL has been cryptanalysed: this system has been analysed more than all its competitors. SSL has been reviewed by numerous cryptographic experts. Therefore, it can be considered secure.**
- **It is widely used: it is easy to create programs that will communicate with other programs using SSL.**

History

https 1994 http Netscape

SSL 2.0 February 1995 Netscape The SSL Protocol Version 2.0: using a server certificate in X.509 v3 format.

SSL 3.0 November 1996 Netscape The SSL Protocol Version 3.0

the client must be able to exploit its private key and, on the other hand, provide its certificate in X.509 v3 format.

TLS 1.0 January 1999

IETF RFC 2246 (Internet Engineering Task Force)

TLS Extensions June 2003 IETF RFC 3546

TLS Extensions April 2006 IETF RFC 4366

Ports and applications using SSL and TLS

Protocol: NSIIOPS TCP Port: 261 Description: IIOP Name Service over SSL and TLS

Protocol: HTTPS TCP Port: 443 Description: HTTP over SSL and TLS

Protocol: DDM-SSL TCP Port: 448 Description: DDM-SSL

Protocol: SMTPS TCP Port: 465 Description: SMTP over SSL and TLS

Protocol: NNTPS TCP Port: 563 Description: NNTP over SSL and TLS

Protocol: SSHELL TCP Port: 614 Description: SSL Shell

Protocol: LDAPS TCP Port: 636 Description: LDAP over SSL and TLS

Protocol: FTPS-DATA TCP Port: 989 Description: FTP data over SSL and TLS

Protocol: FTPS TCP Port: 990 Description: FTP control over SSL and TLS

Protocol: TELNETS TCP Port: 992 Description: Telnet over SSL and TLS

Protocol: IMAPS TCP Port: 993 Description: IMAP4 over SSL and TLS

Protocol: IRCS TCP Port: 994 Description: IRC over SSL and TLS

Protocol: POP3S TCP Port: 995 Description: POP3 over SSL and TLS

Implementations of SSL and TLS

Implementations in web browsers

The majority of SSL and TLS implementations are found in web browsers and servers. The Apache server, in particular, can utilise SSL through an implementation based on OpenSSL.

OpenSSL

Implemented in C, OpenSSL is a cryptographic toolkit comprising two libraries (one for general cryptography and one implementing the SSL protocol), as well as a command-line tool. OpenSSL supports SSL 2.0, SSL 3.0, and TLS 1.0. OpenSSL is distributed under an Apache-style licence.

GnuTLS

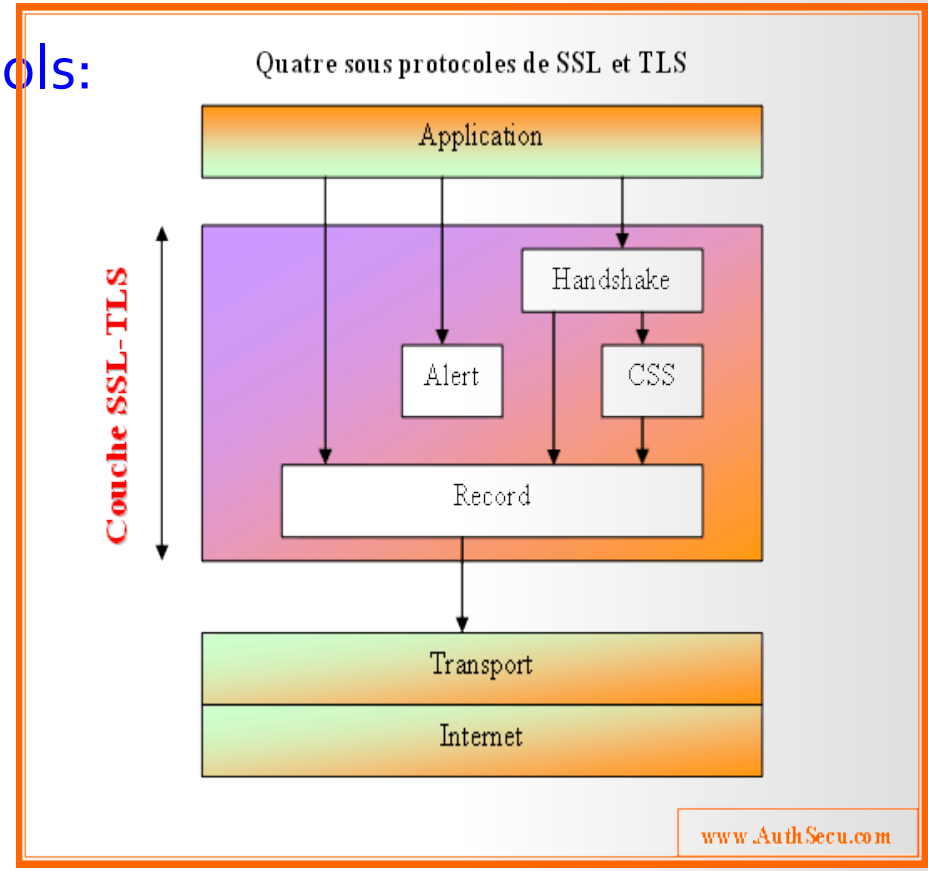
The GnuTLS project provides an implementation of the TLS protocol conforming to IETF specifications. GnuTLS supports TLS 1.1, TLS 1.0, SSL 3.0, and TLS extensions. It allows authentication via X509 and PGP certificates. Unlike OpenSSL, GnuTLS is compatible with GPL licences.

SSL Operation

The SSL and TLS protocols

are subdivided into four sub-protocols:

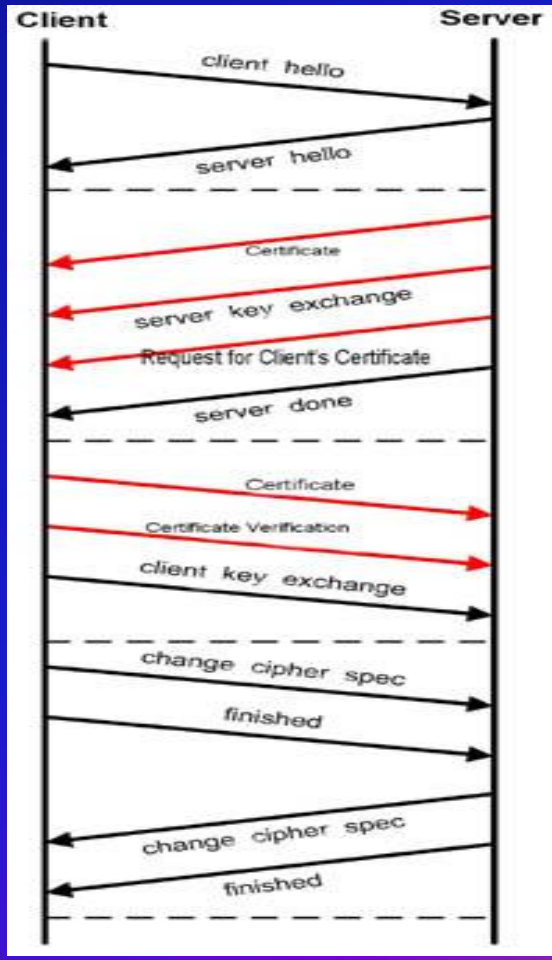
- Handshake Protocol
- Change Cipher Spec Protocol
- Alarm Protocol
- Record Protocol -



SSL Handshake protocol

- Client send: ClientHello
  Highest SSL version supported
  32-byte random number
  SessionID
  List of supported encryption methods
  List of supported compression methods

  ●Server sends ServerHello
  SSL version that will be used
  32-byte random number
  SessionID
  Encryption method that will be used
  Compression method that will be used
  ●**Server authentication,**
  sending of the Certificate
  public key certificate
  Issuing authority's root certificate
  ●When the client receives the certificate, he decides whether it is a trusted server or not.

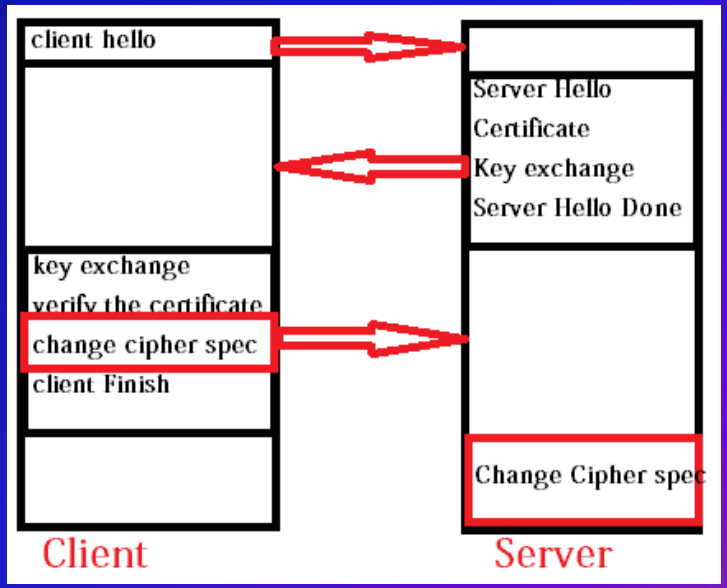


ChangeCipherSpec Protocol

a special protocol with a single message

Upon sending the ChangeCipherSpec message, all subsequent messages will be encrypted

ChangeCipherSpec is always followed by a Finished message



SSL communication ('record')

SSL Record protocol: Once negotiated, they encrypt all information exchanged and perform various checks.

With SSL, the sender of data:

– splits the data into packets

– compresses the data

– cryptographically signs the data

– encrypts the data

– sends them

SSL communication ('record')

The one who receives the data:

– decrypts the data

– verifies the signature of the data

– decompresses the data

– reassembles the data packets

The Alert protocol

The Alert protocol can be invoked:

- by the application, for example to signal the end of a connection
- by the Handshake protocol following an issue that occurred during its process

by the Record layer directly, for example if the integrity of a message is in question

How does SSL protect communications?

It uses:

– a …….system to generate the session key.

– a …….using the session keys to encrypt the data.

– a …….system to ensure that messages are not corrupted.

SSL must ensure confidentiality, integrity, and authentication: what algorithms are used in each function?

The role of certificates:

During an SSL negotiation, it is essential to verify the identity of the person you are communicating with. How can you be sure that the server you are speaking to is indeed

This is where certificates come into play.

When you connect to a secure web server, it will send you a certificate containing its public key, the company name, its address, etc. It is a kind of identification.

How can you verify the authenticity of this identification?

It is the PKIs (Public Key Infrastructures), external companies (which you implicitly trust), that will verify the authenticity of the certificate. (The list of these PKIs is included in your browser. There are generally VeriSign, Thawte, etc.)

These PKIs cryptographically sign the certificates of companies (and they charge for this).

The attacks and weaknesses of SSL:

- SSL is theoretically vulnerable to brute force attacks when using 40-bit keys, so it is advisable to use 128-bit keys.
- SSL is very vulnerable to man-in-the-middle attacks:

  the attacker intercepts (physically) the client's request and pretends to be the server to the client, while also pretending to be a client to the legitimate server. They therefore receive the entirety of the supposedly protected flow.
- Supporting deprecated protocols (SSLv2/SSLv3/TLS 1.0)

Conclusion

– The SSL protocol is currently the only security protocol deployed and used on a large scale, its main advantage being its transparency in relation to the TCP protocol.

It guarantees authentication, confidentiality, and data integrity.

– With its modular architecture, it is not limited to traditional applications, as it integrates wireless networks like WAP (Wireless Application Protocol).

H protocol (Secure Shell) Secure Remote Access

- SSH is a protocol for securing remote access to a machine over a network.
- SSH is the replacement for: telnet, rsh, rlogin and can replace ftp.
- Use encryption.
- – It allows commands to be executed remotely securely.
- - The transmission is compressed.

Origin

telnet issues:

The data sent is clear.

A third party can inspect the traffic.

Use of cryptography in application flows!

SSH (Secure Shell)

History

Created by Tatu Ylönen in 1995, a student at Helsinki University of Technology

Free SSH1 version(1995)

developed by SSH Communications Security Corp., Finland

SSH 2 version (2006)

Open SSH

Two distributions are available:

Commercial version

freeware (www.openssh.com)

Advantages

- session encryption and one-time password (OTP)
- the SSH client is available on a very large number of platforms
- security is ensured from the client to the server (end-to-end security)

Functions

Secure command shell

Port forwarding

Secure file transfer

1.Secure Command Shell

Allows editing of files.

View the contents of directories.

Create a user account

Change permission

any command prompt operation can be safely performed remotely

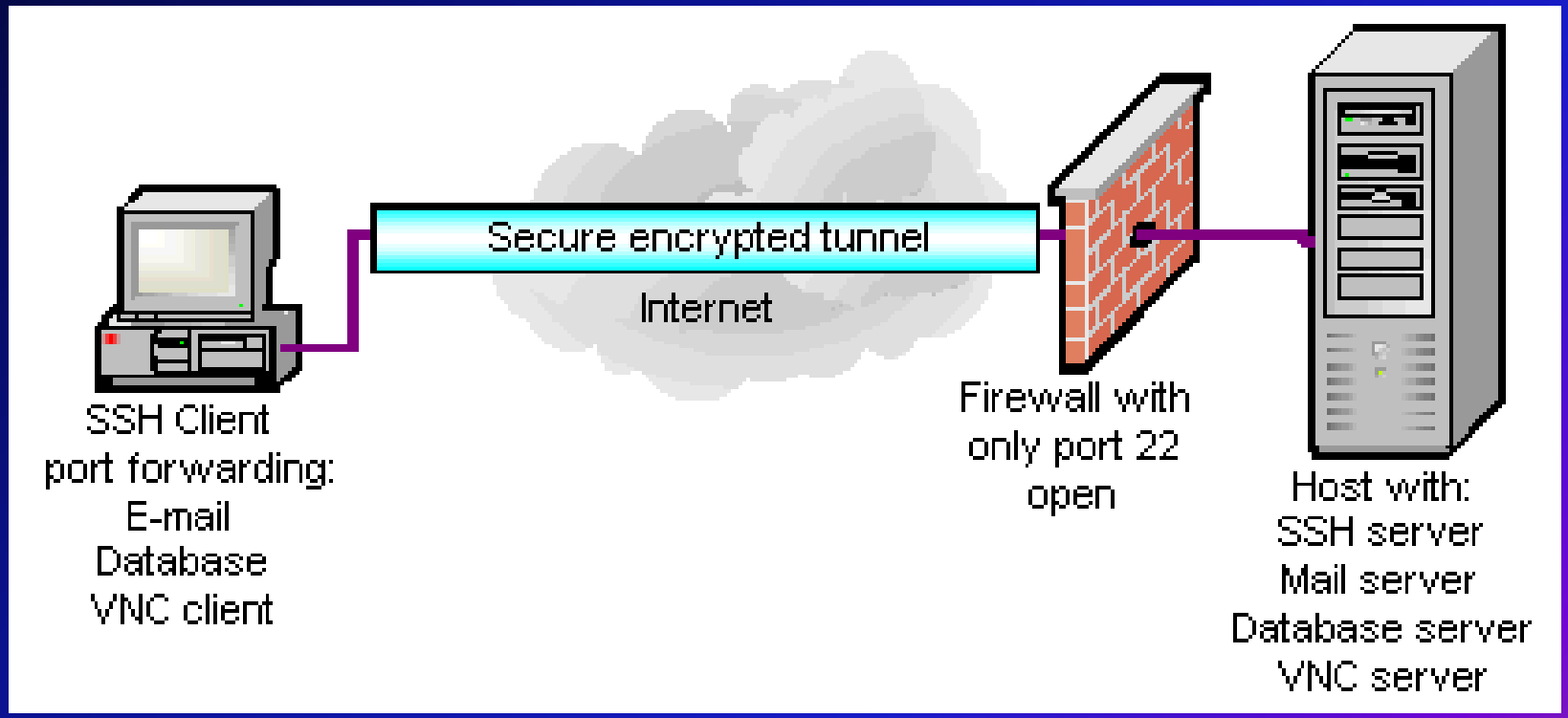
2.SSH: port forwarding

Port Forwarding allows the use of an SSH connection to transport insecure protocols (POP, NNTP, ...).

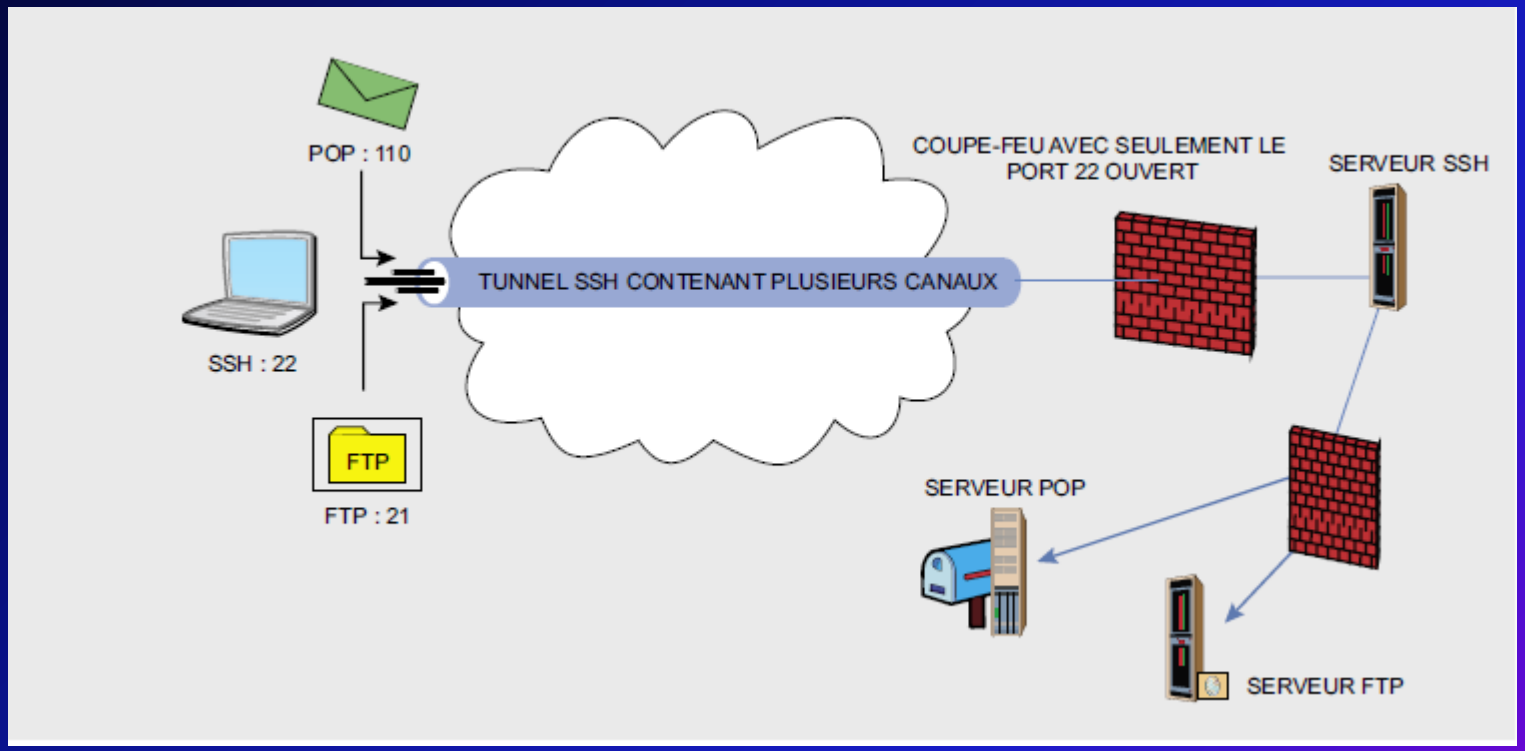
We can then build a VPN based on SSH connections.

Allows the securing of TCP/IP application data

2.SSH: port forwarding



2.SSH: port forwarding

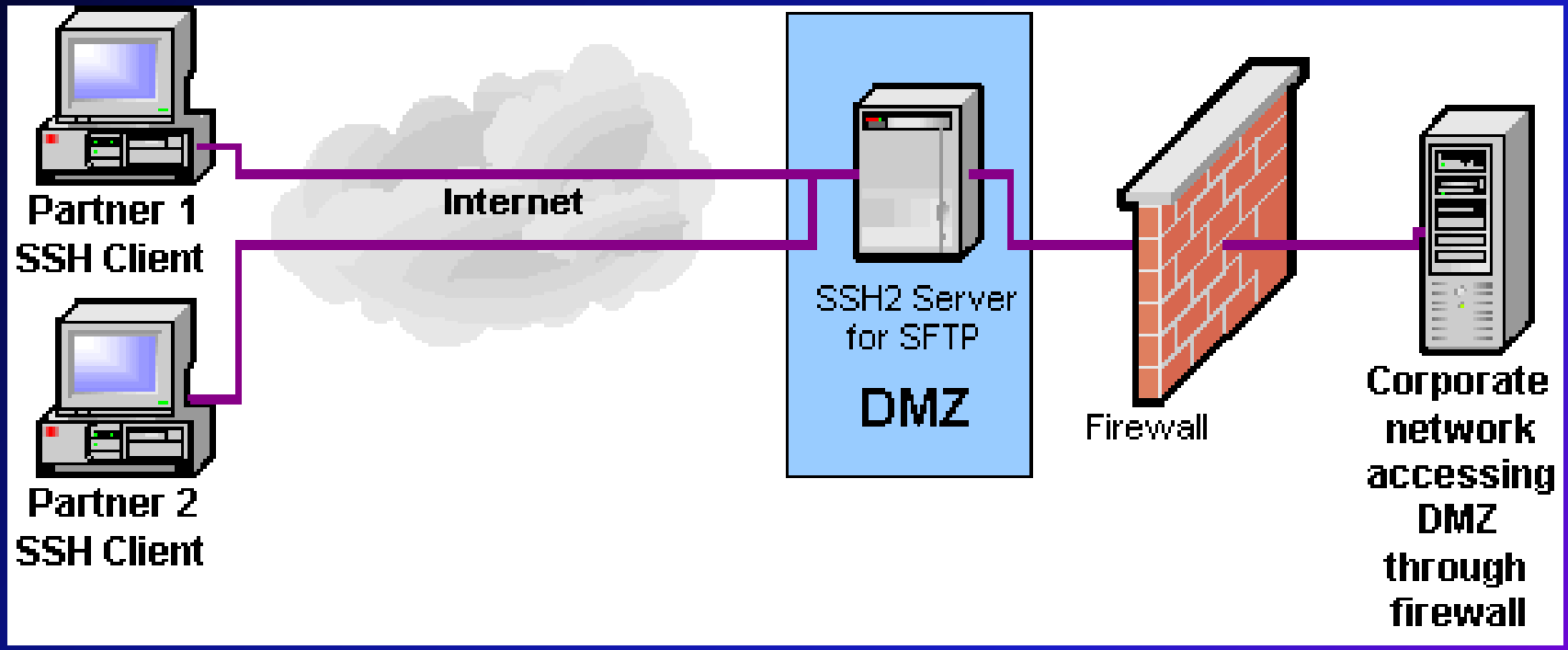


3.Secure file transfer

Secure File Transfer Protocol (SFTP) is a subsystem of the SSH protocol.

SFTP encrypts the username/password and the data to be transferred.

Uses the same SSH port of the server, there is no need to open any other ports on the firewall or router.


Partner 1
SSH Client
Partner 2
SSH Client
Internet
SSH2 Server
for SFTP
DMZ
Firewall
Corporate
network
accessing
DMZ
through
firewall

SSH Versions

The SSH protocol exists in two major versions:

– Version 1 -> Shell or command line.

- This version suffered from security problems in data integrity verification.

– Version 2: -> more secure cryptographically,

What algorithms are used in both versions?

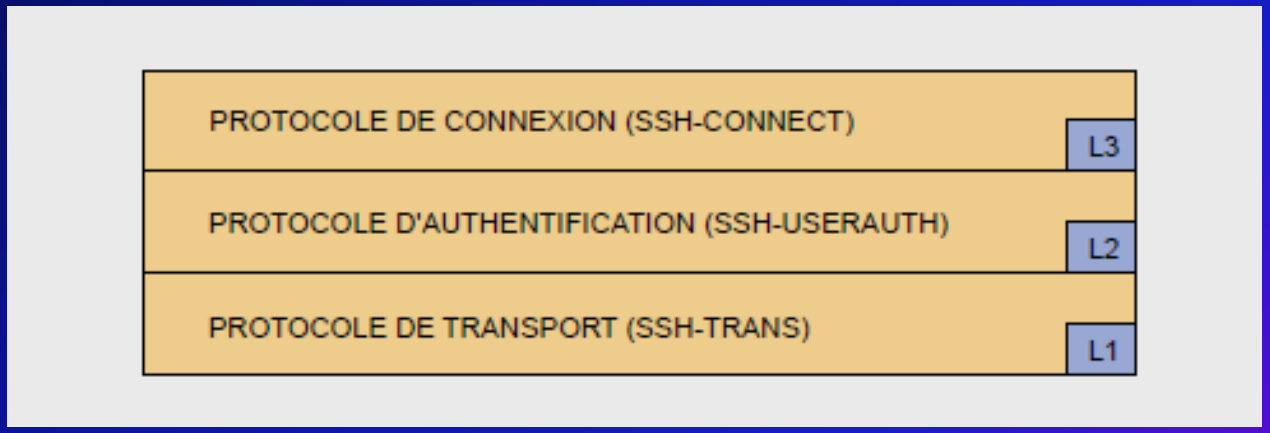
Architecture:

SSH uses a client-server architecture to ensure

authentication,

encryption and

the integrity of data transmitted over a network.



Operation

1. the client and the server exchange in clear their SSH protocol version numbers.

2. The server begins by sending the client:

the list of supported encryption methods,

the list of supported authentication methods

protocol extension indicators (for example, the compression method, etc.) and a 64-bit cookie that the client must return. This cookie is designed to protect the server against denial of service attacks.

3. The client in turn sends the list of supported encryption methods, the list of supported authentication methods, and a copy of the server's cookie.

4. The client and the server choose the best algorithms supported by both.

5. The client and the server separately calculate a session identifier from the Diffie-Hellman values exchanged between the two entities.

Operation

6. The server sends its public key to the client and signs the previously exchanged values with its private key.

7. The client verifies the server's signature and then switches to encrypted mode.

8. The server responds to the client with an encrypted confirmation message.

9. Both parties, the client and the server, are now in encrypted mode using the selected algorithm and key.

10. The client now sends a request for a service (e.g., ssh-userauth or ssh-connection).

11. The server specifies the authentication methods it can accept (public key, password, etc.).

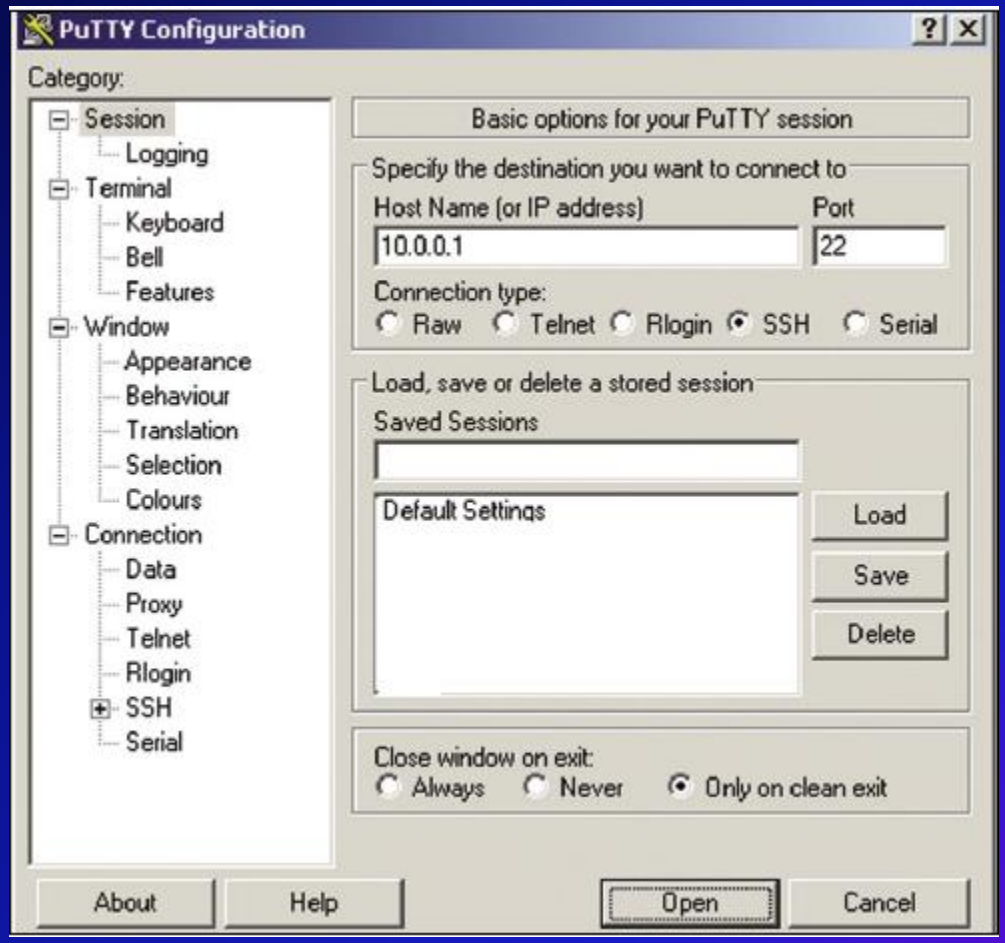
12. Finally, the client sends its authentication method that the server accepts or rejects. In most SSH implementations and according to the server policy, the client is allowed three attempts to authenticate.

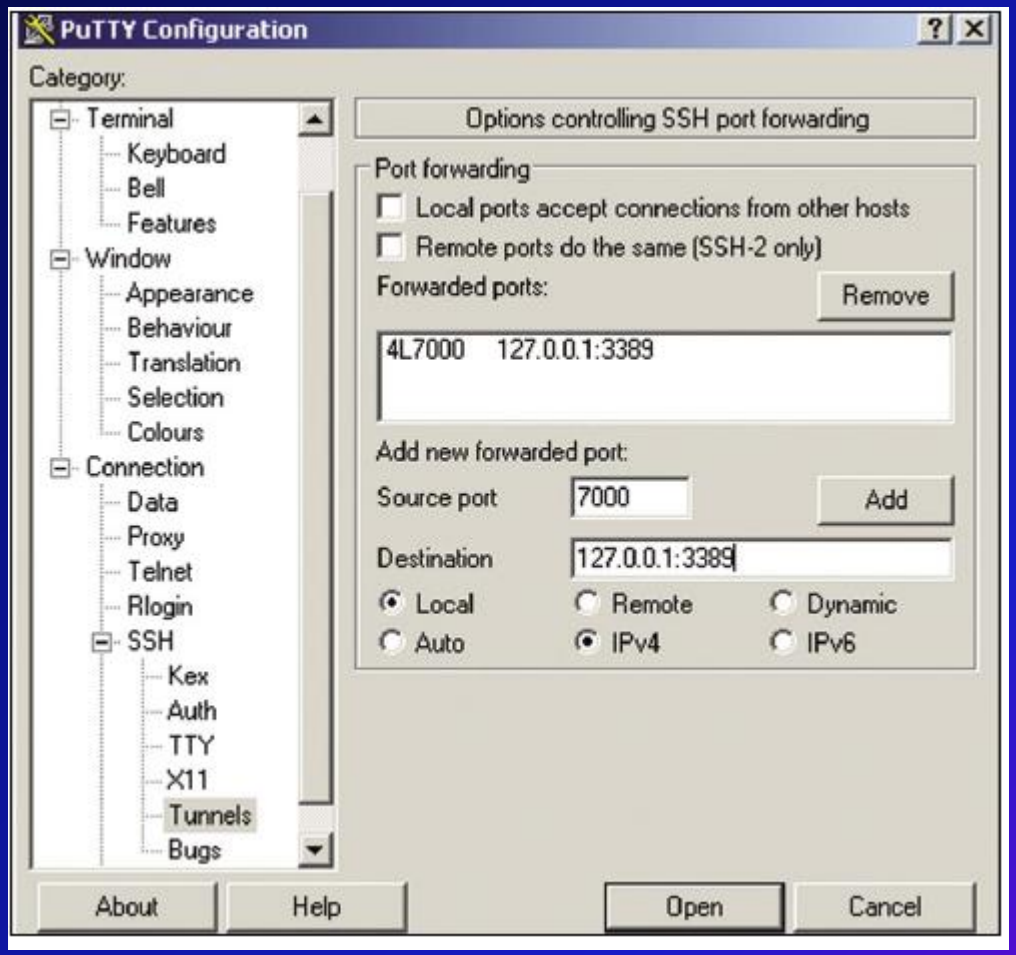
Advanced implementation and use of SSH in a Windows environment

SSH Server:

Copssh: http://www.itefix.no

SSH client: The most well-known is undoubtedly Putty


PuTTY Configuration
Category:
Session
Logging
Terminal
Keyboard
Bell
Features
Window
Appearance
Behaviour
Translation
Selection
Colours
Connection
Data
Proxy
Telnet
Rlogin
SSH
Serial
Basic options for your PuTTY session
Specify the destination you want to connect to
Host Name (or IP address)
Port
10.0.0.1
22
Connection type:
Raw
Telnet
Rlogin
SSH
Serial
Load, save or delete a stored session
Saved Sessions
Default Settings
Load
Save
Delete
Close window on exit:
Always
Never
Only on clean exit
About
Help
Open
Cancel


PuTTY Configuration
Category:
Terminal
Keyboard
Bell
Features
Window
Appearance
Behaviour
Translation
Selection
Colours
Connection
Data
Proxy
Telnet
Rlogin
SSH
Kex
Auth
TTY
X11
Tunnels
Bugs
Options controlling SSH port forwarding
Port forwarding
Local ports accept connections from other hosts
Remote ports do the same (SSH-2 only)
Forwarded ports:
Remove
4L7000 127.0.0.1:3389
Add new forwarded port:
Source port
7000
Add
Destination
127.0.0.1:3389
Local
Remote
Dynamic
Auto
IPv4
IPv6
About
Help
Open
Cancel

The advantages:

SSH constitutes a powerful and practical approach to securing communications over a network of computers.

Through its authentication mechanism, SSH allows for secure tunnelling of remote connections, file transfers, TCP/IP port forwarding, and other important features. Thus, it has the following advantages:

The creation of a VPN at the exchange level

The concept of Tunnelling

The disadvantages

SSH is vulnerable to denial of service attacks, thereby inheriting the weaknesses of TCP/IP on which it relies. Furthermore, depending on the environment, SSH is susceptible to certain attack methods, such as traffic analysis and interception.

The IPSEC protocol

1. Introduction

IPsec is a standard that defines a security extension for the internet protocol (IP):

Packet listening

Address spoofing

to enable the securing of networks based on this protocol.

Confidentiality:

Authentication:

Integrity:

Protection against replay

The security is implemented at the IP level,

IPsec can be deployed on all network devices and provide a unique means of protection for all data exchanges.

IPsec = Internet Protocol Security

History

- Standard developed at the IETF since 1992
- First version in 1995
- Enhanced version, with dynamic security parameter management, in November 1998
- Optional in IPv4
- Mandatory in IPV6

IPSEC

Security protocols:

- Authentication Header (AH)
- Encapsulation Security Payload (ESP)

Key exchange protocol:

- Internet Key Exchange (IKE)

Internal databases:

- Security Policy Database (SPD)
- Security Association Database (SAD)

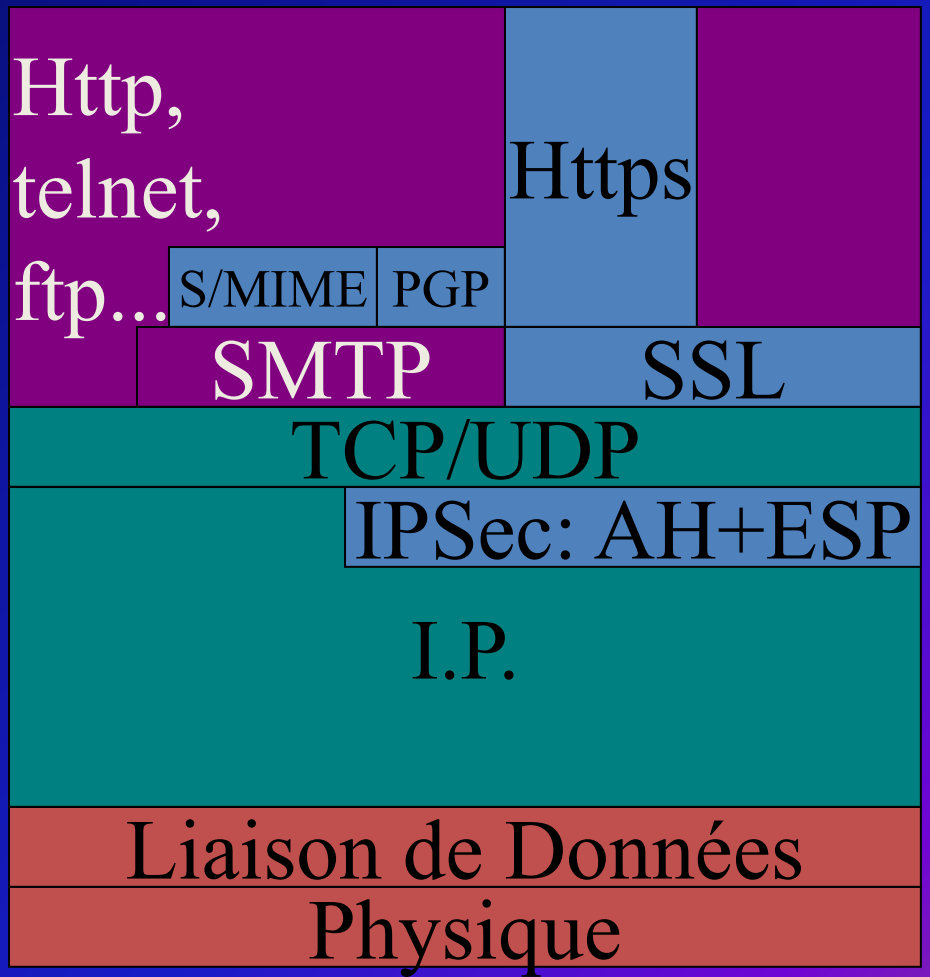
IPSEC: positioning

« application »

« déplacement de bout en bout »

« routage »

« déplacement de point en point »



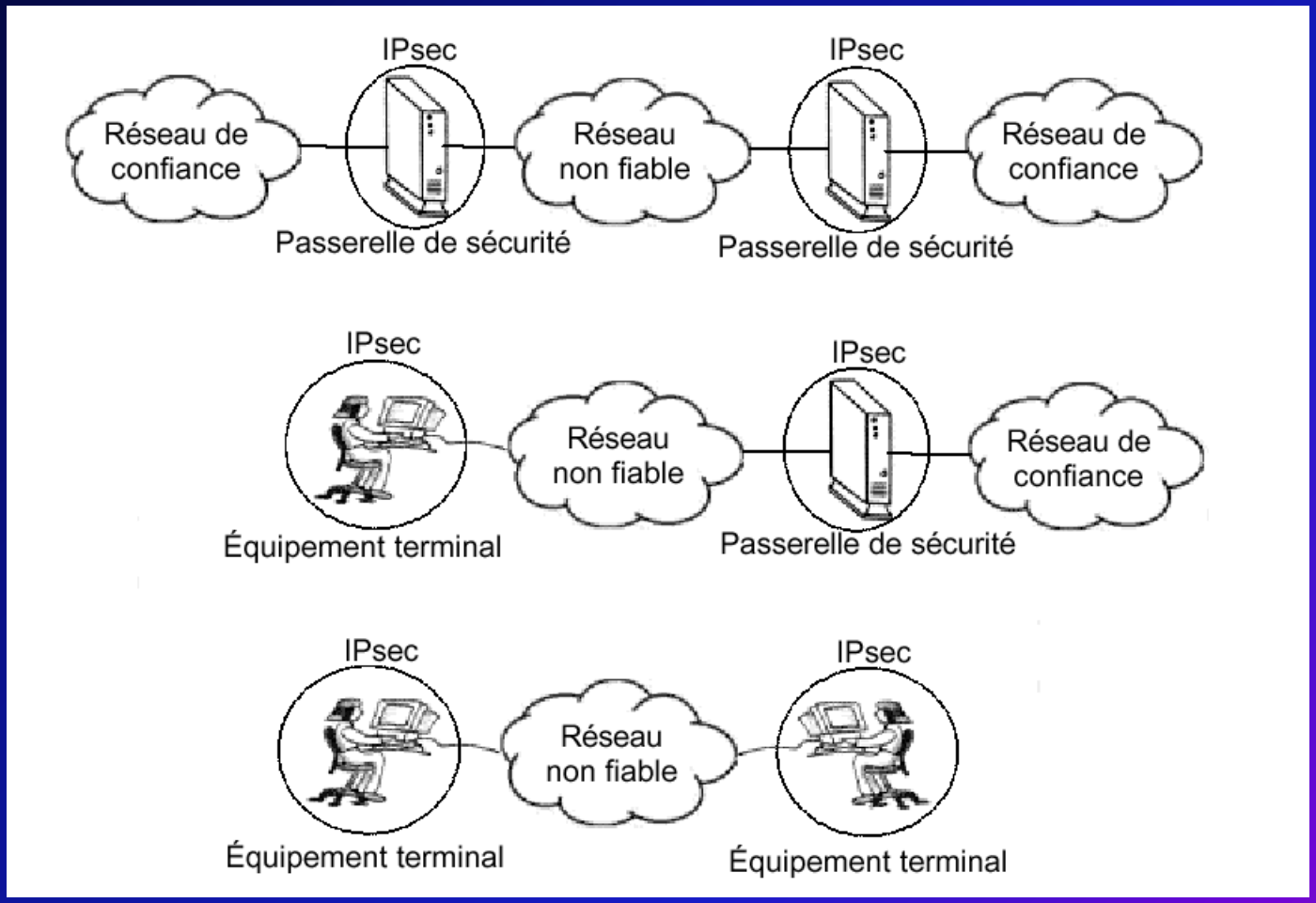
2. Securing exchanged data

Where does IPsec fit in?

IPsec is integrated into the TCP/IP protocol stack at the IP level.

IPsec can be implemented on all devices using the network and provide either end-to-end protection, between communicating parties, or link-by-link protection, across network segments.

Where does IPsec come into play?



What security services are provided?

IPsec aims to prevent various attacks made possible by the IP protocol.

IPsec can provide the following security services:

Data confidentiality and partial protection against traffic analysis.

Data authenticity and continuous access control.

Protection against replay attacks.

Advantage:

Ability to use it only on specific communications (without disrupting other communications).

IPSec is below the transport layer (TCP, UDP); it is therefore transparent to applications (allowing for increased security without modifying higher-level applications).

Once established, IPSec is transparent to users.

How are these services provided?

Security services are provided through two extensions of the IP protocol:

AH (Authentication Header)

is used to validate the integrity of messages and to prove the identity of the sender, as well as to prevent replay attacks.

ESP (Encapsulating Security Payload):

has the primary role of ensuring confidentiality but can also ensure data authenticity.

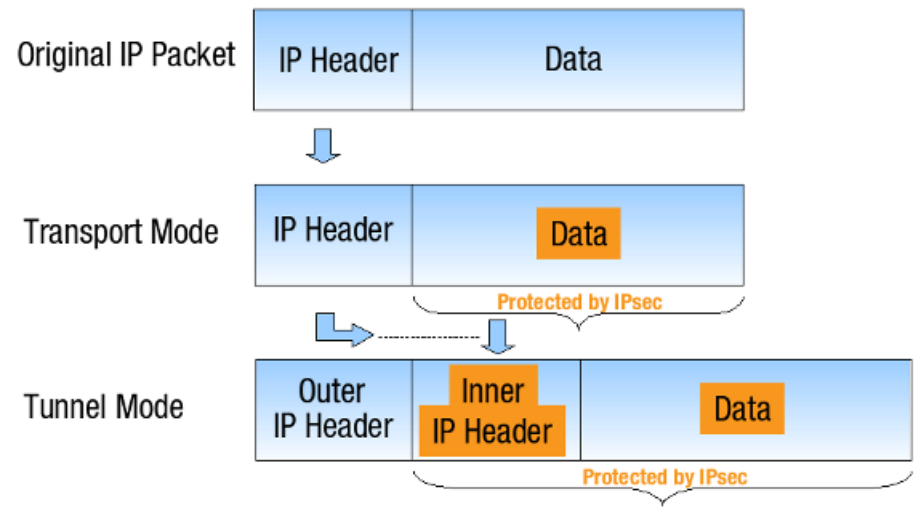
3. The two modes of operation of IPSec

The transport mode: protects only the content of the IP packet without touching the header; this mode can only be used on end devices (client machines, servers). AH, ESP, or both.

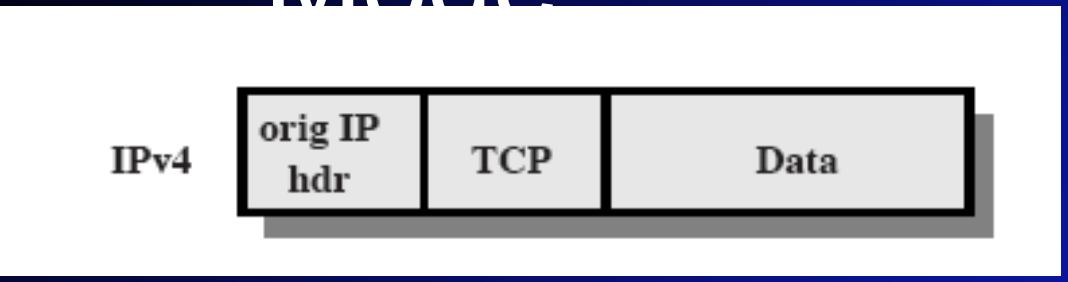
The tunnel mode: allows the creation of tunnels by 'encapsulating' each IP packet in a new packet. Thus, the protection covers all fields of the incoming IP packets at the entrance of a tunnel, including the fields of the headers (source and destination addresses).

AH or ESP.

The two operating modes of IPSec



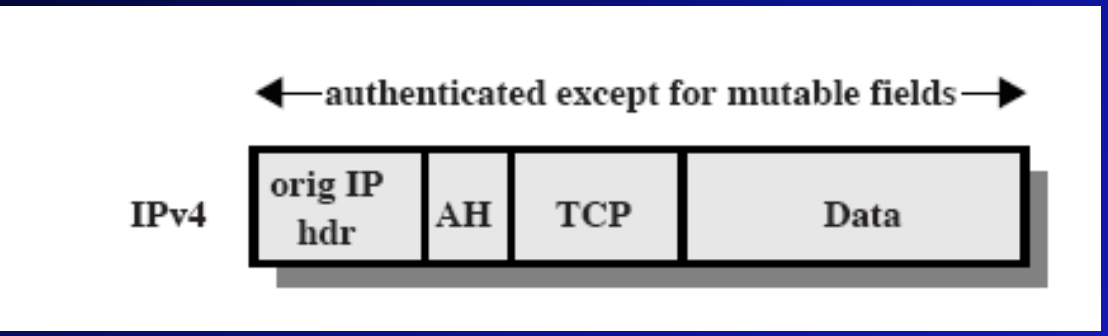
AH: Tunnel and Transport Mode



Original

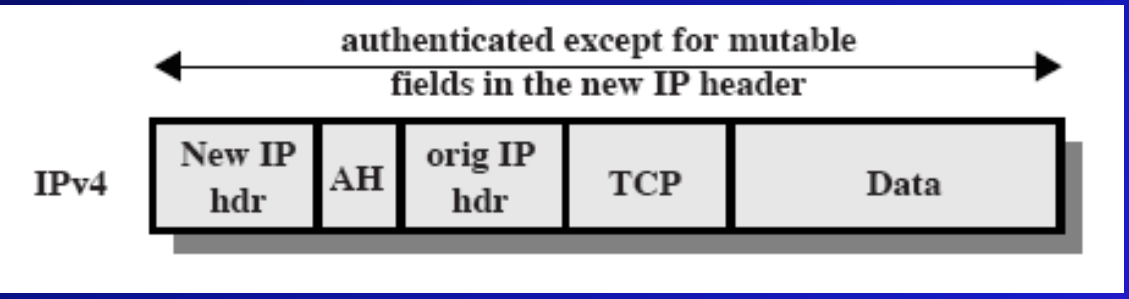
Transport Mode

Cover most of the original packet

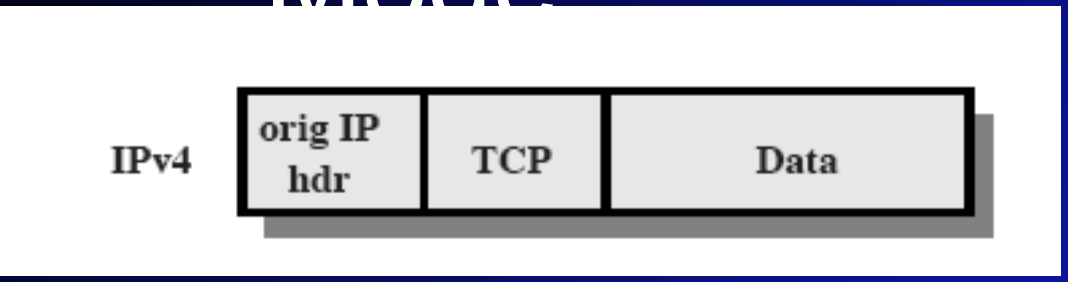


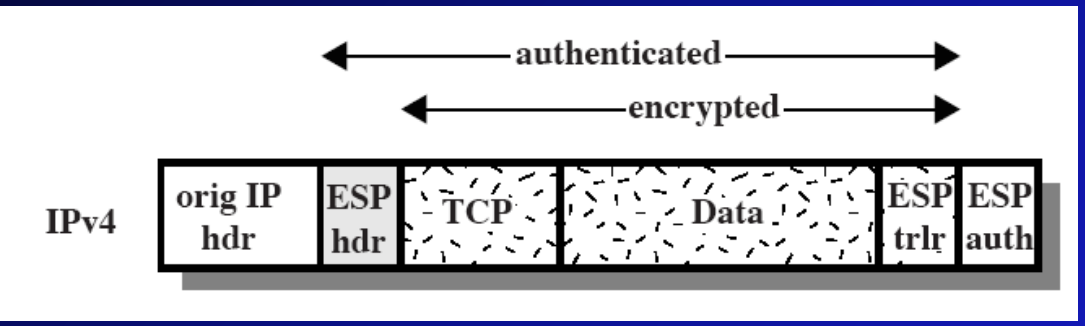
Tunnel Mode

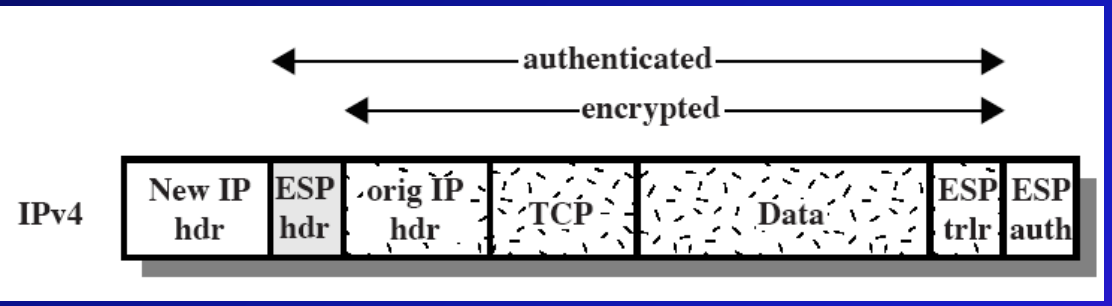
Cover entire original packet



ESP: Tunnel and Transport Mode







Transport Mode

Good for host to host traffic

Tunnel Mode

Good for VPNs, gateway to gateway security

Course Questions 10

- Q1: Provide the use cases for each security protocol
- Q2: What types of encryption algorithms are used
- Q3: What security functions are provided by the SSL protocol
- Q4: What security functions are provided by the SSH protocol
- Q5: What is the role of PKI: Public Key Infrastructure
- Q6: What is the use of a certificate?
- Q7: A browser displays an invalid certificate, why?