

# SEGURIDAD DE LA INFORMACIÓN

TEMA 5

**SEGURIDAD EN REDES TCP/IP**

## Índice del tema

- Seguridad en la Capa de Transporte
- Seguridad en la Capa de Internet
- Firewalls en Redes
- Seguridad en la Capa de Acceso a Red: el caso de las Redes Inalámbricas

## SEGURIDAD EN LA CAPA DE TRANSPORTE





- Existen diversos tipos de amenazas que pueden aparecer al usar la Web, y los posibles puntos de ataque son:
  - el cliente Web,
  - el servidor Web,
  - la propia información entre cliente y servidor

	Threats	Consequences	Countermeasures
<b>Integrity</b>	<ul style="list-style-type: none"><li>• Modification of user data</li><li>• Trojan horse browser</li><li>• Modification of memory</li><li>• Modification of message traffic in transit</li></ul>	<ul style="list-style-type: none"><li>• Loss of information</li><li>• Compromise of machine</li><li>• Vulnerability to all other threats</li></ul>	Cryptographic checksums
<b>Confidentiality</b>	<ul style="list-style-type: none"><li>• Eavesdropping on the Net</li><li>• Theft of info from server</li><li>• Theft of data from client</li><li>• Info about network configuration</li><li>• Info about which client talks to server</li></ul>	<ul style="list-style-type: none"><li>• Loss of information</li><li>• Loss of privacy</li></ul>	Encryption, web proxies
<b>Denial of Service</b>	<ul style="list-style-type: none"><li>• Killing of user threads</li><li>• Flooding machine with bogus requests</li><li>• Filling up disk or memory</li><li>• Isolating machine by DNS attacks</li></ul>	<ul style="list-style-type: none"><li>• Disruptive</li><li>• Annoying</li><li>• Prevent user from getting work done</li></ul>	Difficult to prevent
<b>Authentication</b>	<ul style="list-style-type: none"><li>• Impersonation of legitimate users</li><li>• Data forgery</li></ul>	<ul style="list-style-type: none"><li>• Misrepresentation of user</li><li>• Belief that false information is valid</li></ul>	Cryptographic techniques

- WebGoat: [https://www.owasp.org/index.php/Proyecto\\_WebGoat\\_OWASP](https://www.owasp.org/index.php/Proyecto_WebGoat_OWASP)
  - Es una aplicación Web que contiene y muestra intencionalmente muchos agujeros de seguridad, diseñada por OWASP para mostrar vulnerabilidades de seguridad y enseñar seguridad en aplicaciones Web

**Tema 5: Seguridad en Redes TCP/IP**

5

# WebGoat



The screenshot shows the WebGoat application's user interface. On the left, there is a sidebar with a red header containing a white goat logo and the text "WEBGOAT". Below the header, the sidebar lists various security categories: General, Access Control Flaws, AJAX Security, Authentication Flaws, Buffer Overflows, Code Quality, Concurrency, Cross-Site Scripting (XSS), Improper Error Handling, Injection Flaws, Denial of Service, Insecure Communication, Insecure Configuration, Insecure Storage, Malicious Execution, Parameter Tampering, Session Management Flaws, and Web Services. Each category has a small arrow icon indicating it can be expanded.

- 28 lecciones y cuatro laboratorios
- Requiere un proxy funcionando en el puerto 8080:

The screenshot shows the OWASP ZAP 2.5.0 interface. At the top, the menu bar includes Archivo, Editar, Ver, Analizar, Reporte, Herramientas, Online, Ayuda. The main window shows a tree view of contexts and sites, with "WebGoat" selected. A context "GET:all" is also expanded. On the right, a large dialog box titled "Opciones" (Options) is open, specifically the "Proxy local" tab. It contains fields for "Address (eg localhost, 127.0.0.1)" set to "localhost" and "Puerto (ej. 8080)" set to "8080". There is a note: "Establece la configuración. Los puertos http y https tienen que ser los mismos que los de arriba." Below the address fields are two checked checkboxes: "Modify/Remove 'Accept-Encoding' request-header" and "Always unzip gzipped content". Under "Security Protocols", several checkboxes are present, with "SSLv2Hello" unchecked and others ("SSL 3", "TLS 1", "TLS 1.1", "TLS 1.2") checked. At the bottom of the dialog are "Cancelar" (Cancel) and "Aceptar" (Accept) buttons. The status bar at the bottom of the ZAP window shows "Alertas 0 1 3 0" and "Escaneo actual 0 0 0 0 0 0 0 0".

# SQL en WebGoat



WebGoat - Mozilla Firefox

WebGoat Problem loading page Preferences

localhost/WebGoat/start.mvc

Most Visited ▾ Offensive Security Kali Linux Kali Docs Kali Tools Exploit-DB Aircrack-ng

Buffer Overflows >

Code Quality >

Concurrency >

Cross-Site Scripting (XSS) >

Improper Error Handling >

Injection Flaws >

Command Injection

Numeric SQL Injection

Log Spoofing

XPATH Injection

LAB: SQL Injection

Stage 1: String SQL Injection

Stage 2: Parameterized Query #1

Stage 3: Numeric SQL Injection

Stage 4: Parameterized Query #2

String SQL Injection

Modify Data with SQL Injection

Add Data with SQL Injection

strar aplicaciones

Blind Numeric SQL Injection

Blind String SQL Injection

Denial of Service >

THIS LESSON ONLY WORKS WITH THE DEVELOPER VERSION OF WEBGOAT

Implement a fix to block SQL injection into the fields in question on the Login page. Repeat stage 1. Verify that the attack is no longer effective.

\* Login failed

Goat Hills Financial Human Resources

Please Login

Neville Bartholomew (admin) ▾

Password  ••••

Login

Params

ParamValue

Screen 6

menu 1100

• Vamos a intentar entrar como admin

# SQL en WebGoat



- Activar el proxy para capturar las solicitudes

The screenshot shows the OWASp ZAP (Zed Attack Proxy) interface. A large blue arrow points to the red 'Stop' button in the toolbar at the top center. The main window displays a captured HTTP request for 'GET:attack(Screen.menu)'. The request details show a GET request to 'http://localhost/WebGoat/service/cookie.mvc' with various headers and a cookie. The response pane is empty. The bottom navigation bar includes tabs like 'Historia', 'Buscar', 'Alertas', 'Salida', 'Escaneo Activo', 'Puntos de interrupción', and 'Spider(Araña)'. The status bar at the bottom indicates 'Alertas 0 1 3' and 'Escaneo actual 0 0 0 0 0 0 0 0 0'.

# SQL en WebGoat



WebGoat - Mozilla Firefox  
localhost/WebGoat/start.mvc  
THIS LESSON ONLY WORKS WITH THE DEVELOPER VERSION OF WEBGOAT  
Buffer Overflows >  
Code Quality >  
Concurrency >  
Cross-Site Scripting (XSS) >  
Improper Error Handling >  
Injection Flaws >  
Command Injection  
Numeric SQL Injection  
Log Spoofing  
XPath Injection  
LAB: SQL Injection  
Stage 1: String SQL Injection  
Stage 2: Parameterized Query #1  
Stage 3: Numeric SQL Injection  
Stage 4: Parameterized Query #2  
String SQL Injection  
Modify Data with SQL Injection  
Add Data with SQL Injection  
Star aplicaciones cors  
Blind Numeric SQL Injection  
Blind String SQL Injection  
Denial of Service >

\* Login failed  
Goat Hills Financial Human Resources  
Please Login  
Neville Bartholomew (admin)  
Password: \*\*\*\*  
Login

domain: httpOnly: false, maxAge: -1, name: JSESSIONID, path: /, secure: false, value: D944781E27E237FBDEE2C75224; version: 0  
Params: ParamValue: Screen: 6, menu: 1100

- Introducir una clave cualesquiera

Busca el siguiente breakpoint  
(la solicitud)

Sesión sin Nombre - OWASP Test Runner

Archivo Editar Ver Analizar Reporte Herramientas Online Ayuda

Modo estándar

Sitios Scripts

Contextos Default Context Sitos http://localhost WebGoat GET: start.mvc GET: attack(Screen,menu) POST: attack(Screen,menu)(action,employee\_id, password)

Método Header: Vista Raw Cuerpo: Vista Raw

POST http://localhost/WebGoat/attack?Screen=6&menu=1100 HTTP/1.1  
Host: localhost  
User-Agent: Mozilla/5.0 (X11; Linux x86\_64; rv:45.0) Gecko/20100101 Firefox/45.0  
Accept: \*/\*  
Accept-Language: en-US,en;q=0.5  
Content-Type: application/x-www-form-urlencoded; charset=UTF-8  
X-Requested-With: XMLHttpRequest  
Referer: http://localhost:8080/WebGoat/start.mvc  
employee\_id=112&password=hola&action=Login

Navegación Predefinida Parámetros HTTP Sessions Resultados Zest WebSockets AJAX Spider Fuzzer Historia Buscar Alertas Salida Escaneo Activo Puntos de interrupción Spider(Araña)

New Scan Progreso: 0: Attack Mode Scanner 0% Escaneo actual: 0 Num requests: 0

ID	Req. Timestamp	Resp. Timestamp	Método	URL	Code	Reason	RTT	Size Resp. Header	Size Resp. Body

# SQL en WebGoat



The screenshot shows the OWASP ZAP 2.5.0 interface during a penetration test. The left sidebar displays the 'Contexts' tree, with 'Default Context' and 'Sistios' expanded, showing 'localhost' and 'WebGoat' contexts. The 'Sistios' context contains 'start.mvc', 'attack(Screen,menu)', 'lessons', and 'service' sub-folders. The main pane shows a 'Punto de interrupción' (Breakpoint) at 'GET:attack(Screen,menu)' with a raw POST request. The request payload includes 'employee\_id=112&password=hola&action=Login'. A blue arrow points from this payload area down to a browser window below. The browser window shows the URL 'http://localhost/WebGoat/start.mvc' and the modified query string 'employee\_id=112&password=' OR '1'='1&action=Login', where the password field has been altered to include the SQL injection condition.

- Realizar el ataque

# SQL en WebGoat



**THIS LESSON ONLY WORKS WITH THE DEVELOPER VERSION OF WEBGOAT**

Implement a fix to block SQL injection into the fields in question on the Login page. Repeat stage 1. Verify that the attack is no longer effective.

The screenshot shows a web application interface for 'Goat Hills Financial Human Resources'. At the top, there is a logo of a goat and the text 'Goat Hills Financial Human Resources'. Below this, a navigation bar says 'Welcome Back Neville - Staff Listing Page'. A dropdown menu is open, listing several staff members with their roles: Larry Stooge (employee), Moe Stooge (manager), Curly Stooge (employee), Eric Walker (employee), Tom Cat (employee), Jerry Mouse (hr), David Giambi (manager), Bruce McGuirre (employee), Sean Livingston (employee), Joanne McDougal (hr), and John Wayne (admin). The 'Larry Stooge (employee)' entry is highlighted with a blue background. To the right of the dropdown menu is a vertical column of buttons: 'SearchStaff', 'ViewProfile', 'CreateProfile', 'DeleteProfile', and 'Logout'. A large blue arrow points from the text '• Se accede en modo administrador' to the 'John Wayne (admin)' entry in the dropdown menu.

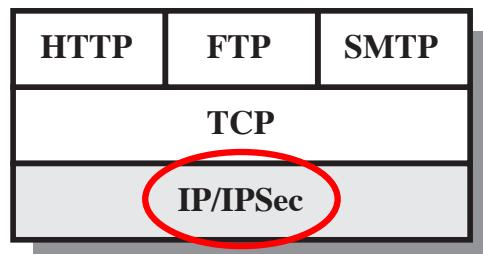
- Se accede en modo administrador

- XXE (XML External Entity)
  - Ataque de injection contra un sistema que gestiona entradas XML
  - Este ataque puede liderar a lecturas o modificaciones no autorizadas, o DoS

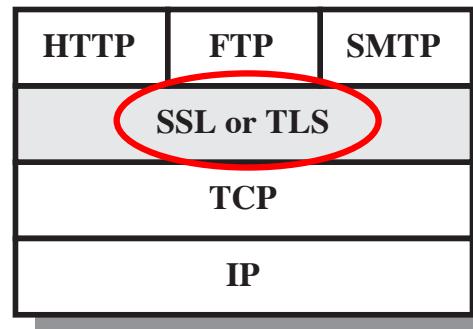
The screenshot shows the WebGoat application interface for the XXE challenge. The left sidebar menu is visible, showing various security flaws: Introduction, General, Injection Flaws (SQL Injection, SQL Injection [advanced], SQL Injection [mitigations], XXE), Authentication Flaws, Cross-Site Scripting (XSS), Access Control Flaws, Insecure Communication, Request Forgeries, Vulnerable Components - A9, Client side, and Challenges. The 'XXE' item under 'Injection Flaws' is highlighted.

The main content area is titled 'XXE'. It contains a navigation bar with 'Show hints' and 'Reset lesson' buttons, and a sequence of numbered steps from 1 to 8. Below this is a section titled 'Let's try' with the instruction: 'In this assignment you will add a comment to the photo, when submitting the form try to execute an XXE injection with the comments field. Try listing the root directory of the filesystem.' A photo of a cat is displayed, with the caption 'John Doe uploaded a photo. 24 days ago'. The photo itself has text overlaid: 'HUMAN' at the top and 'I REQUEST YOUR ASSISTANCE' at the bottom. Below the photo is a comment input field labeled 'Add a comment' and a 'Submit' button. A previous comment by 'webgoat' is shown: '2017-10-18, 23:04:18 Silly cat....'

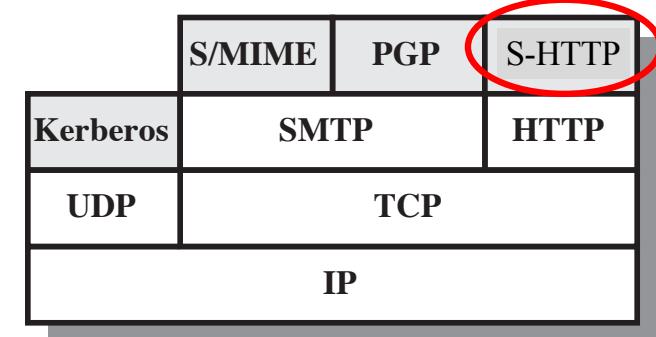
- De los tres puntos de ataques mencionados anteriormente (**cliente Web, servidor Web y tráfico**), nosotros centramos en la gestión del tráfico de red
- Así, para dar una solución al problema del tráfico Web, existen diferentes soluciones, dependiendo de la capa TCP/IP a considerar



(a) Network Level



(b) Transport Level



(c) Application Level

- El IETF formó a mediados de los 90 un Grupo de Trabajo denominado *Web Transaction Security (WTS)*
  - su objetivo: desarrollar los requisitos y las especificaciones para la provisión de servicios de seguridad en transacciones Web

## **Web Transaction Security (wts)** **(concluded WG)**

[Documents](#) | [Charter](#) | [History](#) | [List Archive »](#) | [Tools WG Page »](#)

### **Description of Working Group**

The goal of the Web Transaction Security Working Group is to develop requirements and a specification for the provision of security services to Web transaction, e.g., transactions using HyperText Transport Protocol

(HTTP). This work will proceed in parallel to and independently of the development of non-security features in the HTTP Working Group. The working group will prepare two documents for submission as Internet Drafts; an HTTP Security Requirements Specification, and an HTTP Security Protocol Specification. The latter will be submitted as a Standards Track RFC.

### **Goals and Milestones**

Jul 1995	HTTP Security Requirements finalized at the Stockholm IETF. Submit HTTP Security Specification proposal(s) as Internet-Drafts.
Dec 1995	HTTP Security Specification finalized at the Dallas IETF, submit to IESG for consideration as a Proposed Standard.
Done	HTTP Security Requirements submitted as Internet-Draft.

**Note:** The data for concluded WGs is occasionally incorrect.

#### **Group**

Name: Web Transaction Security  
 Acronym: wts  
 Area: Security Area (sec)  
 State: Concluded  
 Charter: [charter-ietf-wts-01](#) (Approved)

#### **Personnel**

Chair: [Charlie Kaufman <charlie\\_kaufman@notesdev.ibm.com>](#)  
 Area Director: ?

#### **Mailing List**

Address: [www-security@nsmx.rutgers.edu](mailto:www-security@nsmx.rutgers.edu)  
 To Subscribe: [www-security-request@nsmx.rutgers.edu](mailto:www-security-request@nsmx.rutgers.edu)  
 Archive: <http://www-ns.rutgers.edu/www-security>

- Este grupo se centró en el desarrollo de una solución en la capa de aplicación, y diseñó el protocolo **SHTTP - Secure HyperText Transfer Protocol**, especificado en los documentos RFC que se observan abajo

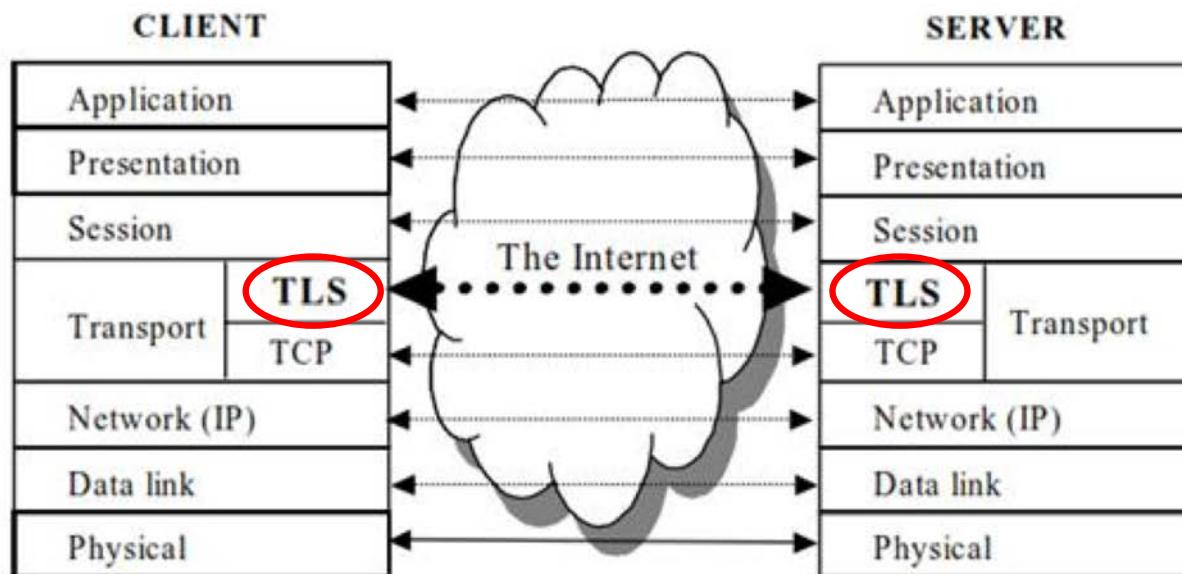
## Web Transaction Security (wts) (concluded WG)

[Documents](#) | [Charter](#) | [History](#) | [List Archive »](#) | [Tools WG Page »](#)

Document	Title	Date	Status	IPR	Area Director
<b>RFCs</b>					
<a href="#">RFC 2084 (draft-ietf-wts-requirements)</a>	Considerations for Web Transaction Security	1997-01	RFC 2084 (Informational)		
<a href="#">RFC 2659 (draft-ietf-wts-shtml)</a>	Security Extensions For HTML	1999-08	RFC 2659 (Experimental)		
<a href="#">RFC 2660 (draft-ietf-wts-shttp)</a>	The Secure HyperText Transfer Protocol	1999-08	RFC 2660 (Experimental)		
<b>Related Documents</b>					

- S-HTTP es una solución alternativa a HTTPS pero obsoleta
  - S-HTTP sólo cifra los datos enviados al servidor sin requerir un proceso de negociación y funcionando en el mismo puerto que HTTP
  - HTTPS, por el contrario, funciona sobre SSL protegiendo el dato antes y después de todo el proceso de transmisión de la misma – nota: SSL se describe a continuación

- Por otro lado, en las mismas fechas, los desarrolladores de Netscape abordaron el problema, pero desde la capa de transporte
  - como una solución intermedia (ni en la capa alta ni en las bajas)
- El resultado fue el protocolo **SSL - Secure Sockets Layer**, una subcapa entre la de aplicación y la de transporte
  - más concretamente, SSL se sitúa por encima de TCP dado que este es orientado a la conexión y proporciona fiabilidad



- el objetivo del protocolo SSL es, por tanto, **crear conexiones seguras y transmitir datos a través de esas conexiones**
- la última versión producida fue la v3.0

# SSL - Secure Sockets Layer

- El protocolo SSL es un **protocolo cliente/servidor** que proporciona los siguientes servicios de seguridad entre los puntos que se comunican:
  - Autenticación de entidades y de origen de datos
  - Confidencialidad de la conexión
  - Integridad de la conexión
- Más concretamente, SSL emplea:
  - criptografía de **clave secreta** para la autenticación de los datos (mensajes) y para el cifrado de los mismos
  - criptografía de **clave pública** para la autenticación de las entidades y para el establecimiento de clave
    - básicamente, hay tres algoritmos de intercambio de clave en la especificación de SSL: RSA, Diffie-Hellmann y Fortezza
    - a pesar de la utilización de criptografía de clave pública, **no proporciona el servicio de no-repudio (ni no-repudio de origen ni no-repudio de entrega)**



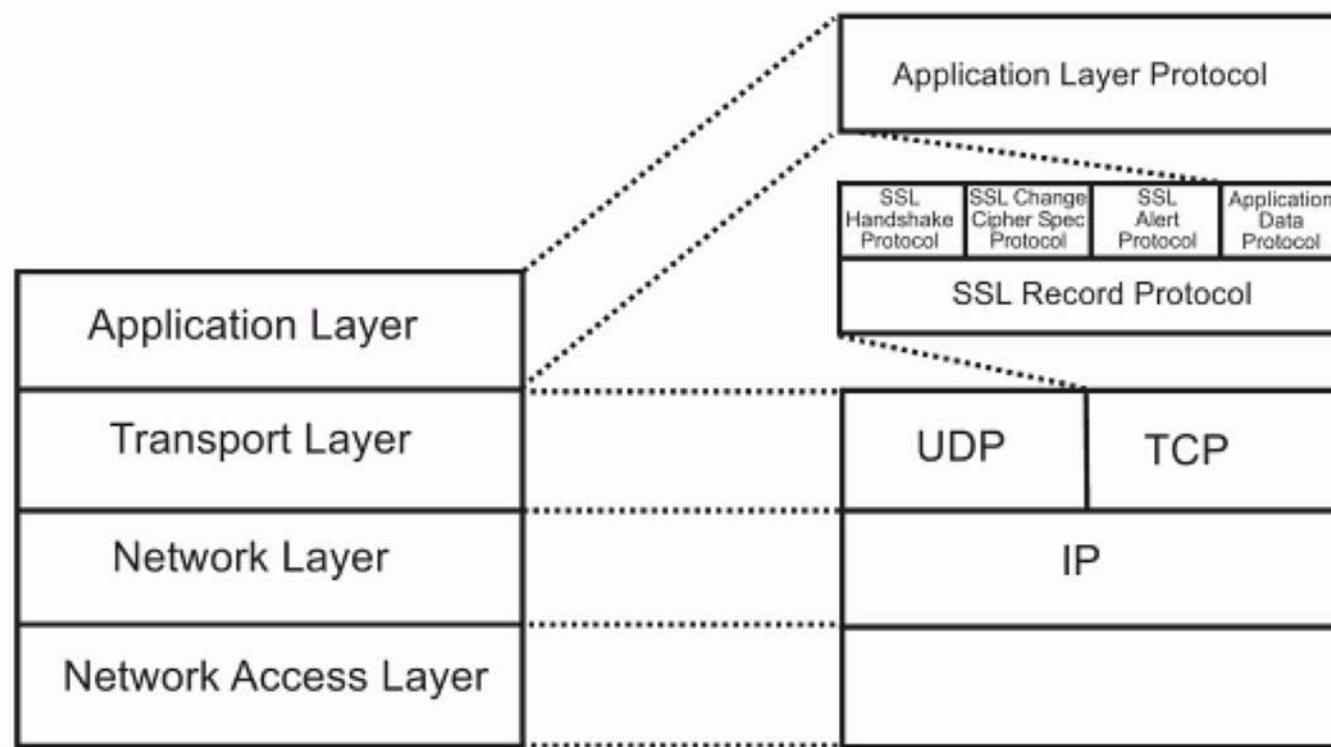
- Una ventaja del protocolo SSL es que es independiente del protocolo de la capa de aplicación
  - es decir, cualquier protocolo de aplicación basado en TCP se puede beneficiar de SSL (éste le dota de los servicios de seguridad mencionados)

Port Numbers Reserved for Application Protocols Layered over SSL/TLS

Protocol	Description	Port #
nsiops	IIOP Name Service over SSL/TLS	261
https	HTTP over SSL/TLS	443
nntps	NNTP over SSL/TLS	563
ldaps	LDAP over SSL/TLS	636
ftps-data	FTP Data over SSL/TLS	989
ftps	FTP Control over SSL/TLS	990
telnets	Telnet over SSL/TLS	992
imaps	IMAP4 over SSL/TLS	993
ircs	IRC over SSL/TLS	994
pop3s	POP3 over SSL/TLS	995
tftps	TFTP over SSL/TLS	3713
sip-tls	SIP over SSL/TLS	5061
...	...	...

- El protocolo SSL emplea, entre otros, estos dos conceptos:
  - **Sesión SSL**: asociación entre el cliente y el servidor en la que se negocian los parámetros de seguridad para todas las conexiones de esa sesión
  - **Conexión SSL**: realización de la transmisión de datos entre el cliente y el servidor, protegida criptográficamente según lo negociado en la sesión
- El ámbito de la funcionalidad de SSL es doble, como se desprende de lo anterior:
  1. Establecer una conexión segura (confidencial y autenticada) entre los puntos que se comunican
  2. Utilizar esa conexión para transmitir de forma segura los datos del nivel de aplicación entre el emisor y el receptor. Esta transmisión requiere a su vez de:
    - Dividir los datos en fragmentos más manejables
    - Procesarlos de forma individual
      - cada fragmento tratado se denomina **SSL record**

- Para llevar a cabo esa doble funcionalidad, SSL consta de dos subcapas y varios subprotocolos, como se observa en la siguiente figura:



- La subcapa alta contiene:
  - **SSL Handshake Protocol**: permite que los puntos de comunicación se autentiquen mutuamente, y que además, negocien un **cipher suite** y (opcionalmente) un método de compresión
  - **SSL Change Cipher Spec Protocol**: permite a los puntos de comunicación activar el cipher suite
  - **SSL Alert Protocol**: permite a los puntos de comunicación indicar posibles problemas potenciales e intercambiar los correspondientes mensajes de alerta
  - **SSL Application Data Protocol**: es el propio protocolo de la capa de aplicación (ej: HTTP) y alimenta al SSL Record Protocol
- La subcapa baja contiene:
  - **SSL Record Protocol**: fragmenta los datos de la capa de aplicación y los procesa de forma individual

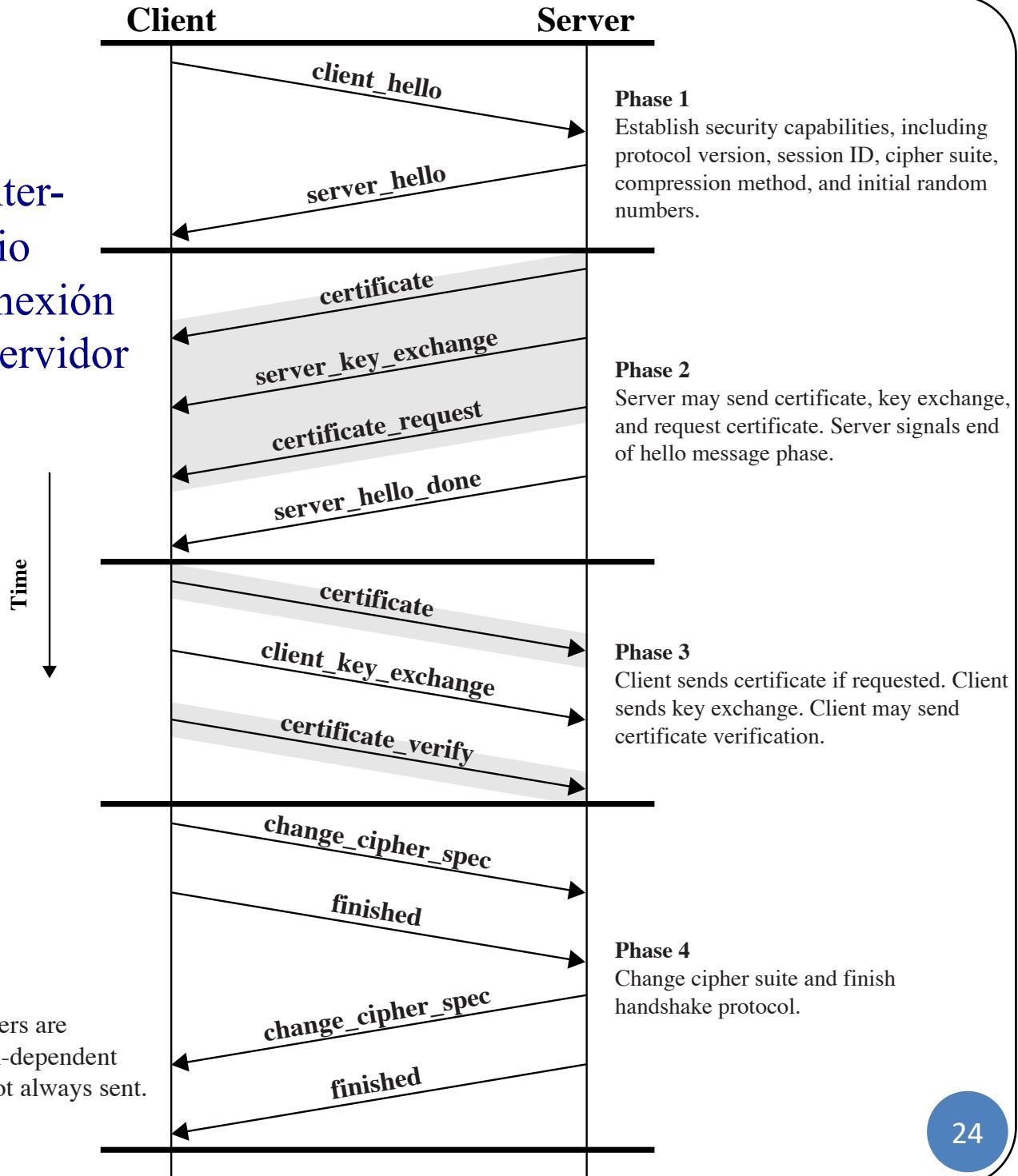
- SSL Handshake Protocol
  - Se utiliza antes de transmitir ningún dato de la capa de aplicación
  - Es la parte más compleja de SSL porque permite al servidor y al cliente:
    - autenticarse mutuamente
    - negociar un algoritmo de cifrado y una función MAC
    - así como las claves a usar para proteger los datos del SSL record
  - Consta de una serie de mensajes intercambiados entre el cliente y el servidor, con el formato:

	1 byte	3 bytes	$\geq 0$ bytes
Type	Length	Content	

- Por lo tanto, cada mensaje tiene 3 campos:
  - *Type* (1 byte): indica uno de 10 posibles mensajes (ver siguiente tabla)
  - *Length* (3 bytes): longitud del mensaje en bytes
  - *Content* ( $\geq 0$  bytes): parámetros asociados con el mensaje (ver también siguiente tabla)

<b>Message Type</b>	<b>Parameters</b>
hello_request	null
client_hello	version, random, session id, cipher suite, compression method
server_hello	version, random, session id, cipher suite, compression method
certificate	chain of X.509v3 certificates
server_key_exchange	parameters, signature
certificate_request	type, authorities
server_done	null
certificate_verify	signature
client_key_exchange	parameters, signature
finished	hash value

- La figura muestra el intercambio inicial necesario para establecer una conexión lógica entre cliente y servidor
- Se observan 4 fases

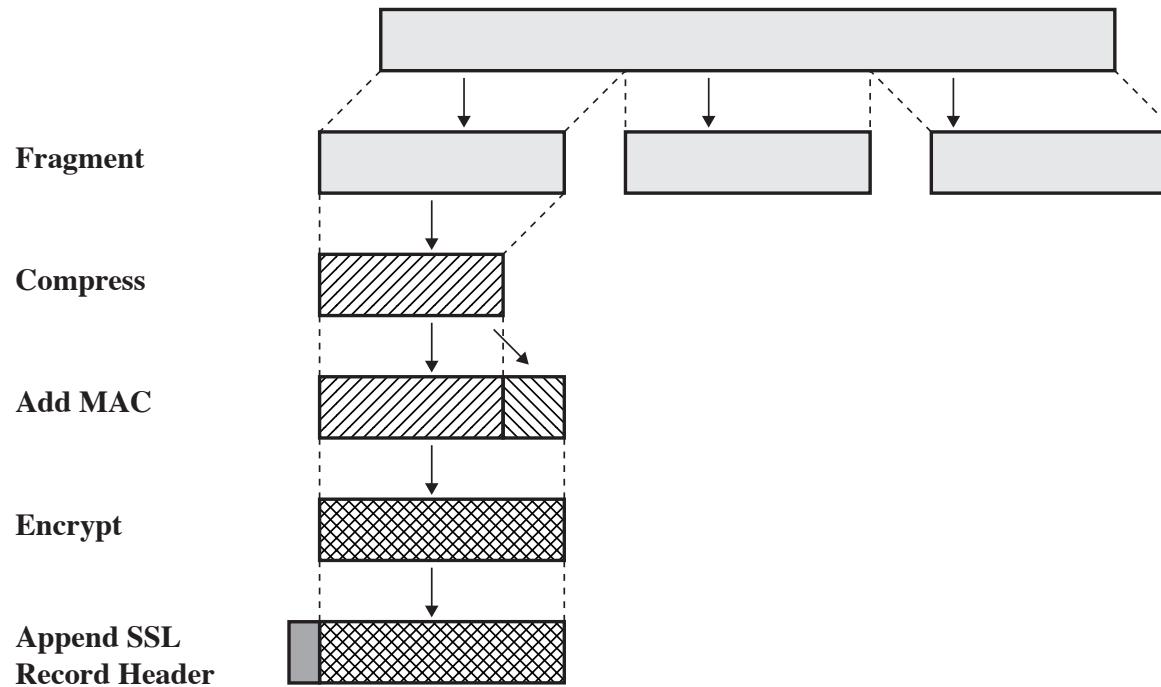


- DHE (Diffie Helman efímero): Tanto el cliente como el servidor generan sus valores secretos ( $x$ ,  $y$ ,  $cli\_sec$ ,  $srv\_sec$ ) en cada negociación
  - Proporciona ***forward secrecy*** (FS) en la creación del secreto compartido
    - FS protege las comunicaciones y las sesiones establecidas en el pasado
      - Es decir, claves de sesión antiguas no serán derivadas
  - Negociación de Claves: DHE/RSA
    - **Server Key Exchange:** Servidor → Cliente:  $p, g, pubKey = g^y \text{ mod } p$ 
      - Estos parámetros están firmados por el Servidor
    - **Client Key Exchange:** Cliente → Servidor:  $pubKey = g^x \text{ mod } p$ 
      - $\text{premaster\_secret} = (g^x)^y \text{ mod } p = (g^y)^x \text{ mod } p = g^{xy} \text{ mod } p$
  - **Negociación de Claves: ECDHE (Curvas elípticas) - TLS**
    - **Server Key Exchange:** Servidor → Cliente:  $a, b, p, G, pubKey = G * srv\_sec$ 
      - Estos parámetros están firmados por el Servidor
    - **Client Key Exchange:** Cliente → Servidor:  $pubKey = G * cli\_sec$ 
      - $\text{premaster\_secret} = \text{server\_pubKey} * \text{cli\_sec} = \text{client\_pubKey} * \text{srv\_sec}$   
 $= G * \text{cli\_sec} * \text{srv\_sec}$

- SSL Change Cipher Spec Protocol
  - Es un protocolo muy simple que consta de un solo mensaje de un solo byte con valor 1 que permite activar el cipher suite
- SSL Alert Protocol
  - Se usa para comunicar al otro punto de comunicación las alertas relacionadas con SSL, y cada mensaje de este protocolo consta de 2 bytes
    - Estos mensajes también se comprimen y se cifran de acuerdo con lo establecido en la sesión
    - El primer byte toma el valor 1 (warning) o 2 (fatal) para informar de la severidad del mensaje. Si el nivel es fatal SSL termina la conexión de forma inmediata
      - Otras conexiones de la misma sesión pueden continuar pero no se producen nuevas conexiones dentro de la misma sesión
    - El segundo byte contiene un código que indica la alerta específica
      - Ejemplos: `unexpected_message`, `bad_record_mac`, `decompression_failure`, `illegal_parameter`, ...

- SSL Record Protocol

- Toma los datos de la subcapa alta, los fragmenta en bloques manejables, los comprime de forma opcional, añade el MAC, cifra, y añade una cabecera
- El resultado final se transmite en un segmento TCP
- En recepción, los datos recibidos son descifrados, verificados, descomprimidos y reensamblados antes de entregarlos a la capa de aplicación



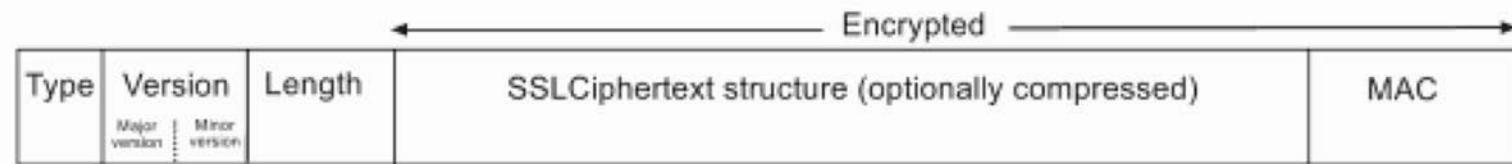
- Por lo tanto, este subprotocolo proporciona:
  - *Confidencialidad*:
    - el SSL Handshake Protocol define una **clave secreta compartida** que es utilizada para el cifrado de los datos
  - *Integridad de datos*:
    - el SSL Handshake Protocol también define **una clave secreta compartida** que es utilizada para formar un **MAC**
- En lo que a fragmentación se refiere, cada mensaje de la capa de aplicación se fragmenta en bloques de longitud  $2^{14}$  bytes o menor
- En lo que respecta al algoritmo de compresión, SSL no especifica ninguno
- En cuanto al código de autenticación de mensajes, se utiliza uno similar a HMAC



- El mensaje comprimido y el valor MAC se cifran utilizando **criptografía simétrica**. Los algoritmos que se pueden utilizar se muestran en la tercera columna de la tabla:

CipherSuite	Key Exchange	Cipher	Hash
<i>SSL_NULL_WITH_NULL_NULL</i>	NULL	NULL	NULL
<i>SSL_RSA_WITH_NULL_MD5</i>	RSA	NULL	MD5
<i>SSL_RSA_WITH_NULL_SHA</i>	RSA	NULL	SHA
<i>SSL_RSA_EXPORT_WITH_RC4_40_MD5</i>	RSA_EXPORT	RC4_40	MD5
<i>SSL_RSA_WITH_RC4_128_MD5</i>	RSA	RC4_128	MD5
<i>SSL_RSA_WITH_RC4_128_SHA</i>	RSA	RC4_128	SHA
<i>SSL_RSA_EXPORT_WITH_RC2_CBC_40_MD5</i>	RSA_EXPORT	RC2_CBC_40	MD5
<i>SSL_RSA_WITH_IDEA_CBC_SHA</i>	RSA	IDEA_CBC	SHA
<i>SSL_RSA_EXPORT_WITH_DES40_CBC_SHA</i>	RSA_EXPORT	DES40_CBC	SHA
<i>SSL_RSA_WITH_DES_CBC_SHA</i>	RSA	DES_CBC	SHA
<i>SSL_RSA_WITH_3DES_EDE_CBC_SHA</i>	RSA	3DES_EDE_CBC	SHA
<i>SSL_DH_DSS_EXPORT_WITH_DES40_CBC_SHA</i>	DH_DSS_EXPORT	DES40_CBC	SHA
<i>SSL_DH_DSS_WITH_DES_CBC_SHA</i>	DH_DSS	DES_CBC	SHA
<i>SSL_DH_DSS_WITH_3DES_EDE_CBC_SHA</i>	DH_DSS	3DES_EDE_CBC	SHA
<i>SSL_DH_RSA_EXPORT_WITH_DES40_CBC_SHA</i>	DH_RSA_EXPORT	DES40_CBC	SHA
<i>SSL_DH_RSA_WITH_DES_CBC_SHA</i>	DH_RSA	DES_CBC	SHA
<i>SSL_DH_RSA_WITH_3DES_EDE_CBC_SHA</i>	DH_RSA	3DES_EDE_CBC	SHA
<i>SSL_DHE_DSS_EXPORT_WITH_DES40_CBC_SHA</i>	DHE_DSS_EXPORT	DES40_CBC	SHA
<i>SSL_DHE_DSS_WITH_DES_CBC_SHA</i>	DHE_DSS	DES_CBC	SHA
<i>SSL_DHE_DSS_WITH_3DES_EDE_CBC_SHA</i>	DHE_DSS	3DES_EDE_CBC	SHA
<i>SSL_DHE_RSA_EXPORT_WITH_DES40_CBC_SHA</i>	DHE_RSA_EXPORT	DES40_CBC	SHA
<i>SSL_DHE_RSA_WITH_DES_CBC_SHA</i>	DHE_RSA	DES_CBC	SHA
<i>SSL_DHE_RSA_WITH_3DES_EDE_CBC_SHA</i>	DHE_RSA	3DES_EDE_CBC	SHA
<i>SSL_DH_anon_EXPORT_WITH_RC4_40_MD5</i>	DH_anon_EXPORT	RC4_40	MD5
<i>SSL_DH_anon_WITH_RC4_128_MD5</i>	DH_anon	RC4_128	MD5
<i>SSL_DH_anon_EXPORT_WITH_DES40_CBC_SHA</i>	DH_anon	DES40_CBC	SHA
<i>SSL_DH_anon_WITH_DES_CBC_SHA</i>	DH_anon	DES_CBC	SHA
<i>SSL_DH_anon_WITH_3DES_EDE_CBC_SHA</i>	DH_anon	3DES_EDE_CBC	SHA
<i>SSL_FORTEZZA_KEY_WITH_NULL_SHA</i>	FORTEZZA_KEY	NULL	SHA
<i>SSL_FORTEZZA_KEY_WITH_FORTEZZA_CBC_SHA</i>	FORTEZZA_KEY	FORTEZZA_CBC	SHA
<i>SSL_FORTEZZA_KEY_WITH_RC4_128_SHA</i>	FORTEZZA_KEY	RC4_128	SHA

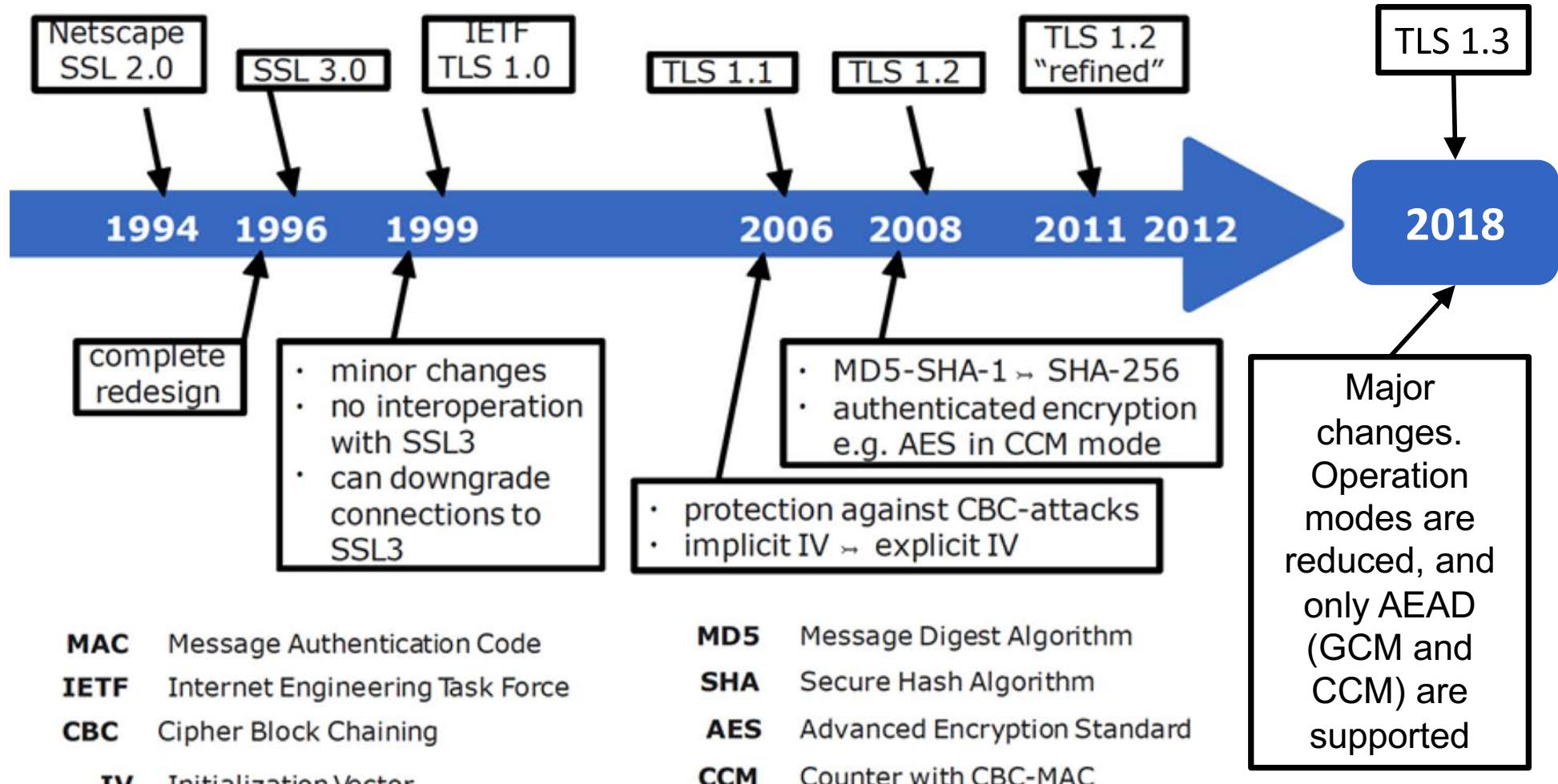
- El paso final del SSL Record Protocol es preparar una **cabecera** que consta de los siguientes campos:
  - *Content Type* (8 bits): protocolo de la subcapa alta de SSL de la que procede el fragmento:
    - 20 → SSL Change Cipher Spec Protocol
    - 21 → SSL Alert Protocol
    - 22 → SSL Handshake Protocol
    - 23 → SSL Application Data Protocol
  - *Major Version* (8 bits): versión de SSL en uso (para SSLv3, el valor es 3)
  - *Minor Version* (8 bits): versión menor en uso (para SSLv3, el valor es 0)
  - *Compressed Length* (16 bits): longitud en bytes del fragmento de texto en claro (o del fragmento comprimido si se ha utilizado compresión)



# Transport Layer Security

- TLS es una iniciativa de IETF para estandarizar SSL
- Se definió por primera vez (TLS 1.0) en 1999, en el RFC 2246.  
Como se menciona en ese RFC:

*“the differences between this protocol and SSL 3.0 are not dramatic, but they are significant to preclude interoperability between TLS 1.0 and SSL 3.0.”*
- Versiones posteriores han sido TLS 1.1, publicada en 2006 en el RFC 4346, TLS 1.2, publicada en 2008 en el RFC 5246, y TLS 1.3, publicada en 2018 en el RFC 8446
  - TLS 1.2 introduce cambios muy significativos, como la inclusión de **AES** en el cipher suite, la consideración de **SHA-256**, así como la consideración de **criptografía de clave pública basada en curvas elípticas**
    - Lo hemos visto anteriormente en el uso de ECDHE para la negociación
  - TLS 1.3 **reduce el tiempo de “handshake”**, y reduce el número de modos de operación soportados, limitándolo a **GCM** y **CCM**



Transport Layer Security (tls) - Charter

datatracker.ietf.org/wg/tls/charter/

NICS Journals EasyChair Mail-Yahoo IEEE Mac UMA Prensa Cine

Ssl and Tls: Theory and Practice - Rolf Oppliger, Ph.D. – Google Libros

Transport Layer Security (tls) - Charter

Sign In

**datatracker.ietf.org**

**Transport Layer Security (tls)**

Documents | Charter | History | List Archive » | Tools WG Page »

**Description of Working Group**

The TLS Working Group was established in 1996 to standardize a 'transport layer' security protocol. The working group began with SSL version 3.0. The TLS Working Group has completed a series of specifications that describe the Transport Layer Security protocol versions 1.0, 1.1, and 1.2, extensions to the protocol, and new ciphersuites to be used with TLS.

The primary goals of the WG are to maintain:

- The TLS protocol, RFC 5246;
- The DTLS protocol, draft-ietf-tls-rfc4347bis.

Significant changes to the protocol, such as a new version 1.3, are not within scope of the working group unless they are explicitly added to the charter.

The secondary goals of the WG are to publish:

- Guidelines for Specifying the Use of TLS/DTLS;
- Recommendations for use of TLS (e.g., server ID);
- Extensions to TLS and DTLS; and,
- Cipher suites.

**Goals and Milestones**

Done	Agreement on charter and issues in current draft.
Done	Final draft for Secure Transport Layer Protocol ('STLP')
Done	Working group 'Last Call'
Done	Submit to IESG for consideration as a Proposed Standard.
Done	First revised draft of TLS specification
Done	TSL 1.1 Specification
Done	First draft of TLS 1.2 specification, including CTR mode cipher suites
Done	First draft of specification for cipher suites with combined encryption/authentication modes
Dec 2011	Heartbeat Extension Sent to IESG

Group

Name: Transport Layer Security  
Acronym: tls  
Area: Security Area (sec)  
State: Active  
Charter: [charter-ietf-tls-04 \(Approved\)](#)

Personnel

Chairs: [Eric Rescorla <ekr@networkresonance.com>](#)  
[Joseph Salowey <jsalowey@cisco.com>](#)  
[Eric Rescorla <ekr@rfcinator.com>](#)  
Area Director: [Sean Turner <turners@ieca.com>](#)  
Tech Advisor: [Allison Mankin <mankin@psg.com>](#)

Mailing List

Address: [tls@ietf.org](#)  
To Subscribe: <https://www.ietf.org/mailman/listinfo/tls>  
Archive: <http://www.ietf.org/mail-archive/web/tls/>

Jabber Chat

Room Address: [xmpp:tls@jabber.ietf.org](#)  
Logs: <http://jabber.ietf.org/logs/tls/>

Version 4.36, 2012-11-07  
Report a bug

Transport Layer Security (tls) - Documents

Ssl and Tls: Theory and Practice – Rolf Oppiger, Ph.D. – Google Libros

Transport Layer Security (tls) - Documents

**Transport Layer Security (tls)**

Documents | Charter | History | List Archive » | Tools WG Page »

Document	Title	Date	Status	IPR	Area Director
<a href="#">draft-ietf-tls-cached-info-13</a>	Transport Layer Security (TLS) Cached Information Extension	2012-09-12	I-D Exists WG Document		
<a href="#">draft-ietf-tls-multiple-cert-status-extension-02</a>	The TLS Multiple Certificate Status Request Extension	2012-10-19	I-D Exists WG Document		
<a href="#">draft-ietf-tls-oob-pubkey-06</a>	Out-of-Band Public Key Validation for Transport Layer Security (TLS)	2012-10-22	I-D Exists WG Document		
<b>RFCs</b>					
<a href="#">RFC 2246 (draft-ietf-tls-protocol)</a>	The TLS Protocol Version 1.0	1999-01	RFC 2246 (Proposed Standard) Obsoleted by <a href="#">RFC 4346</a> Updated by <a href="#">RFC 3546</a> , <a href="#">RFC 5746</a> , <a href="#">RFC 6176</a>		
<a href="#">RFC 2712 (draft-ietf-tls-kerb-cipher-suites)</a>	Addition of Kerberos Cipher Suites to Transport Layer Security (TLS)	1999-10	RFC 2712 (Proposed Standard)		
<a href="#">RFC 2817 (draft-ietf-tls-http-upgrade)</a>	Upgrading to TLS Within HTTP/1.1	2000-05	RFC 2817 (Proposed Standard) <a href="#">Errata</a>		
<a href="#">RFC 2818 (draft-ietf-tls-https)</a>	HTTP Over TLS	2000-05	RFC 2818 (Informational) Updated by <a href="#">RFC 5785</a> <a href="#">Errata</a>		
<a href="#">RFC 3268 (draft-ietf-tls-ciphersuite)</a>	Advanced Encryption Standard (AES) Ciphersuites for Transport Layer Security (TLS)	2002-07	RFC 3268 (Proposed Standard) Obsoleted by <a href="#">RFC 5246</a>		
<a href="#">RFC 3546 (draft-ietf-tls-extensions)</a>	Transport Layer Security (TLS) Extensions	2003-06	RFC 3546 (Proposed Standard) Obsoleted by <a href="#">RFC 4366</a>	Steven Bellovin	
<a href="#">RFC 3749 (draft-ietf-tls-compression)</a>	Transport Layer Security Protocol Compression Methods	2004-05	RFC 3749 (Proposed Standard)	Steven Bellovin	
<a href="#">RFC 4132 (draft-ietf-tls-camellia)</a>	Addition of Camellia Cipher Suites to Transport Layer Security (TLS)	2005-07	RFC 4132 (Proposed Standard) Obsoleted by <a href="#">RFC 5932</a>	Russ Housley	
<a href="#">RFC 4279 (draft-ietf-tls-psk)</a>	Pre-Shared Key Ciphersuites for Transport Layer Security (TLS)	2005-12	RFC 4279 (Proposed Standard)	Russ Housley	
<a href="#">RFC 4346 (draft-ietf-tls-rfc2246-bis)</a>	The Transport Layer Security (TLS) Protocol Version 1.1	2006-04	RFC 4346 (Proposed Standard) Obsoleted by <a href="#">RFC 5246</a> Updated by <a href="#">RFC 4366</a> , <a href="#">RFC 4680</a> , <a href="#">RFC 4681</a> , <a href="#">RFC 5746</a> , <a href="#">RFC 6176</a> <a href="#">Errata</a>	Russ Housley	
<a href="#">RFC 4366 (draft-ietf-tls-rfc3546bis)</a>	Transport Layer Security (TLS) Extensions	2006-04	RFC 4366 (Proposed Standard) Obsoleted by <a href="#">RFC 5246</a> , <a href="#">RFC 6086</a> Updated by <a href="#">RFC 5746</a> <a href="#">Errata</a>	Russ Housley	
<a href="#">RFC 4492 (draft-ietf-tls-ecc)</a>	Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS)	2006-05	RFC 4492 (Informational) Updated by <a href="#">RFC 5246</a> <a href="#">Errata</a>	2 Russ Housley	
<a href="#">RFC 4785 (draft-ietf-tls-psk-null)</a>	Pre-Shared Key (PSK) Ciphersuites with NULL Encryption for Transport Layer Security (TLS)	2007-01	RFC 4785 (Proposed Standard)	Russ Housley	
<a href="#">RFC 5054 (draft-ietf-tls-srp)</a>	Using the Secure Remote Password (SRP) Protocol for TLS Authentication	2007-11	RFC 5054 (Informational)	Tim Polk	
<a href="#">RFC 5081 (draft-ietf-tls-openpgp-keys)</a>	Using OpenPGP Keys for Transport Layer Security (TLS) Authentication	2007-11	RFC 5081 (Experimental) Obsoleted by <a href="#">RFC 6091</a>	Russ Housley	
<a href="#">RFC 5246 (draft-ietf-tls-rfc4346-bis)</a>	The Transport Layer Security (TLS) Protocol Version 1.2	2008-08	RFC 5246 (Proposed Standard) Updated by <a href="#">RFC 5746</a> , <a href="#">RFC 5878</a> , <a href="#">RFC 6176</a> <a href="#">Errata</a>	1 Tim Polk	
<a href="#">RFC 5288 (draft-ietf-tls-rsa-aes-gcm)</a>	AES Galois Counter Mode (GCM) Cipher Suites for TLS	2008-08	RFC 5288 (Proposed Standard)	Pasi Eronen	
<a href="#">RFC 5289 (draft-ietf-tls-ecc-new-mac)</a>	TLS Elliptic Curve Cipher Suites with SHA-256/384 and AES Galois Counter Mode (GCM)	2008-08	RFC 5289 (Informational)	Pasi Eronen	
<a href="#">RFC 5469 (draft-ietf-tls-des-idea)</a>	DES and IDEA Cipher Suites for Transport Layer Security (TLS)	2009-02	RFC 5469 (Informational)	1 Tim Polk	
<a href="#">RFC 5487 (draft-ietf-tls-psk-new-mac-aes-gcm)</a>	Pre-Shared Key Cipher Suites for TLS with SHA-256/384 and AES Galois Counter Mode	2009-03	RFC 5487 (Proposed Standard)	Pasi Eronen	

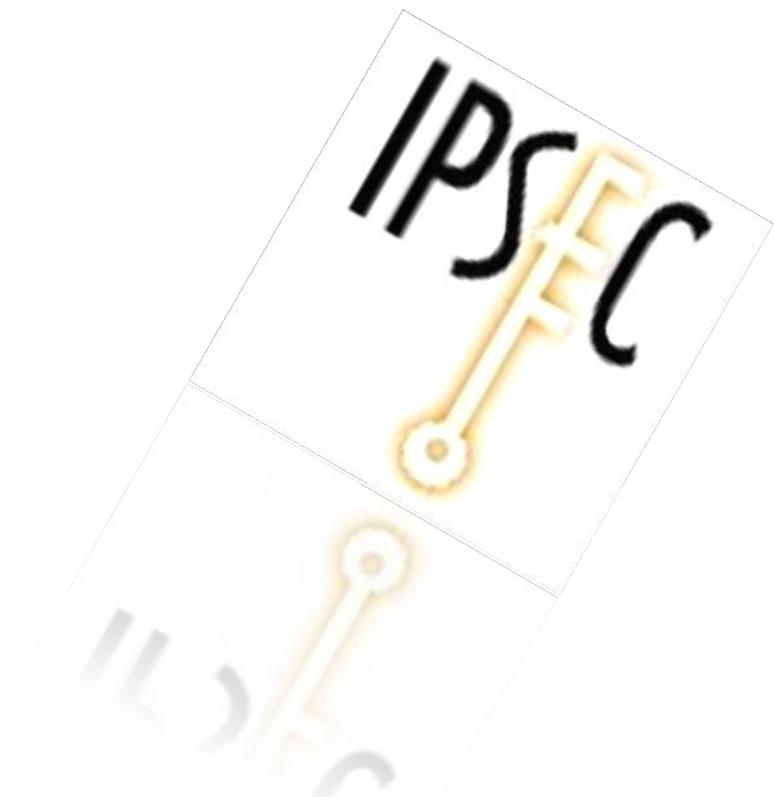
- Es conveniente comentar que existe un protocolo llamado **DTLS** (Datagram Transport Layer Security) definido en el RFC 6347
    - Se utiliza para los protocolos basados en datagramas
      - Es decir, para los que se ejecutan por encima de UDP
    - Se creó en 2006, aunque la última versión es de Enero de 2012
    - Esta tomando un papel relevante en entornos restringidos (IoT)
- <https://datatracker.ietf.org/wg/dice/documents/>

The screenshot shows the IETF Datatracker interface. At the top, there's a navigation bar with links for IETF, Datatracker, Groups, Documents, Meetings, Other, User, and a Document search input field. Below the navigation bar, the title "DTLS In Constrained Environments (dice)" is displayed, followed by a "Concluded WG" badge. A horizontal menu bar below the title includes links for About, Documents, Meetings (which is currently selected), History, Photos, Email expansions, List archive, and Tools. The main content area features a table with the following data:

Document	Date	Status	IPR	AD / Shepherd
RFC 7925 (was draft-ietf-dice-profile) Transport Layer Security (TLS) / Datagram Transport Layer Security (DTLS) Profiles for the Internet of Things	2016-07 61 pages	Proposed Standard RFC		Stephen Farrell Zach Shelby

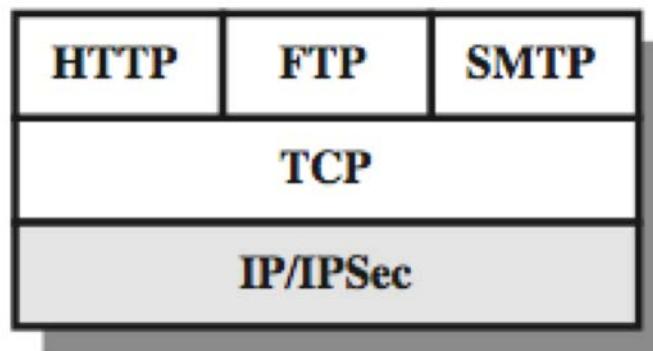
At the bottom of the page, there are links for Atom feed, All changes, Significant, Subscribe to changes, Export as CSV, and footer links for ISOC, IETF Trust, RFC Editor, IRTF, IESG, IETF, IAB, IASA & IAOC, IETF Tools, and IANA.

## SEGURIDAD EN LA CAPA DE INTERNET

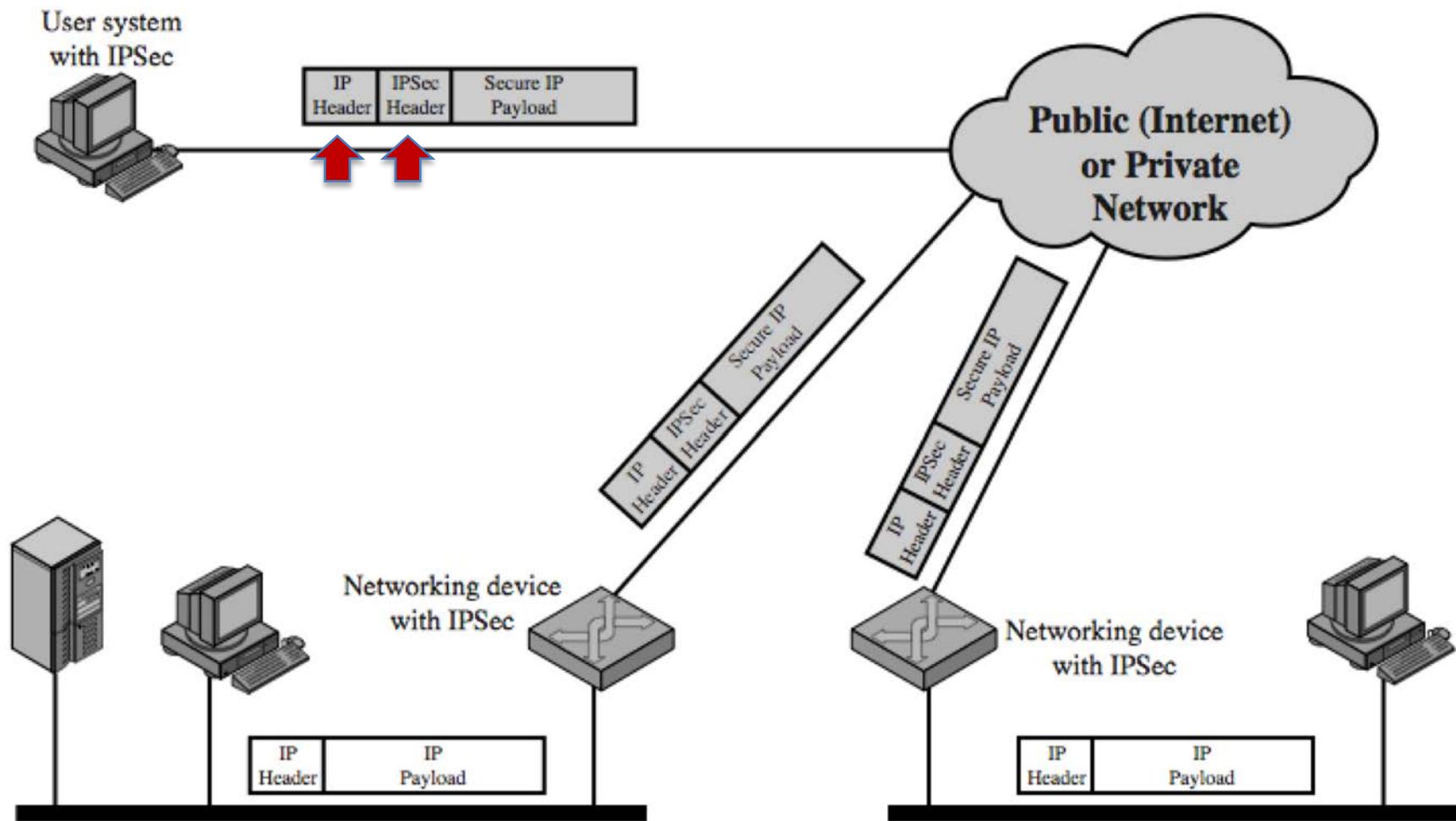


- En 1994, la *Internet Architecture Board (IAB)* publicó un informe titulado “Security in the Internet Architecture” (RFC1636)
- Entre otras, este informe identificaba la necesidad de proporcionar seguridad a la **infraestructura de red**
  - aconsejando la incorporación de mecanismos de **cifrado y autenticación** para la siguiente versión de IP (IPv6)
- A partir de ese momento se elaboraron, bajo el nombre **IPSec** (RFC 4301), las especificaciones y las funcionalidades de seguridad en la capa de Internet para el modelo TCP/IP
  - no sólo teniendo en cuenta IPv6, sino **también para** que sirviera para la propia **IPv4**

- Implementando la seguridad al nivel de IP, una empresa garantiza la protección de **todas sus aplicaciones**, necesiten éstas seguridad o no



- Por ello, se puede usar en muy distintos escenarios:
  - Conectividad segura entre sucursales a través de internet
  - Acceso remoto seguro sobre Internet
  - Establecimiento de conectividad extranet e intranet con socios
  - Aplicaciones de comercio electrónico
  - etc.

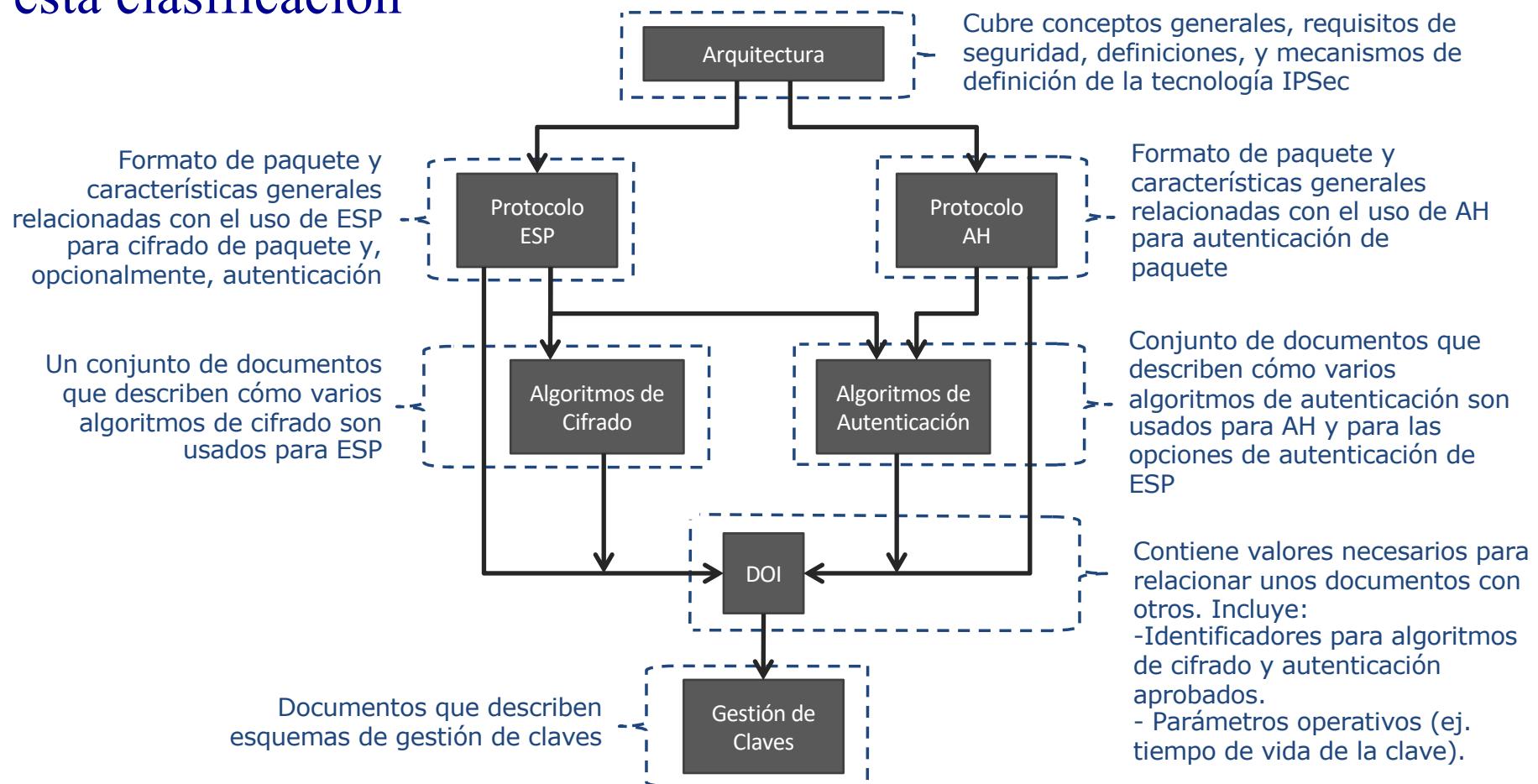


- La seguridad en IPSec se centra en autenticación, confidencialidad e intercambio de claves entre los puntos que se comunican
- Al funcionar por debajo del nivel de transporte, es transparente a las aplicaciones
  - Por lo tanto, es transparente a los usuarios finales:
    - no es necesario entrenar a los usuarios en el uso de mecanismos de seguridad
    - no hace falta que gestionen claves
- IPSec no proporciona servicios de no-repudio
- No proporciona protección frente a ataques DoS



- Para la comunicación segura entre dos puntos, IPSec utiliza los siguientes protocolos:
  - **ESP (Encapsulating Security Payload)**
    - Cabecera para confidencialidad, integridad y autenticación del origen de datos
    - También incluye un número de secuencia que proporciona una forma de protección ante ataques de repetición
    - Además, proporciona una protección parcial ante análisis de tráfico (**sólo en modo túnel**)
  - **AH (Authentication Header)**
    - Cabecera para integridad y autenticación del origen de datos
    - También incluye un número de secuencia que proporciona una forma de protección ante ataques de repetición (mismo motivo que ESP)
  - **IKE (Internet Key Exchange)**
    - Protocolo para generar y distribuir claves criptográficas para ESP y AH
    - También autentica la identidad del sistema remoto
- Antes de que dos puntos se comuniquen de forma segura, tienen que acordar qué parámetros de seguridad se van a aplicar

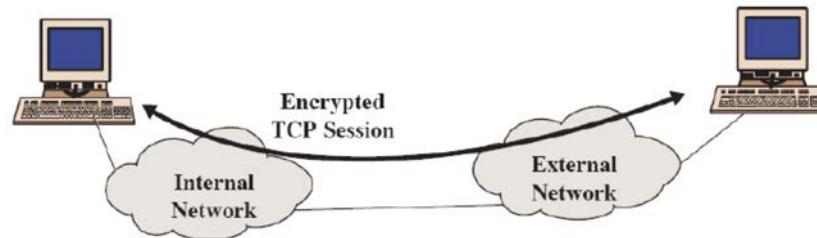
- De hecho, los documentos de IETF al respecto de IPSec siguen esta clasificación



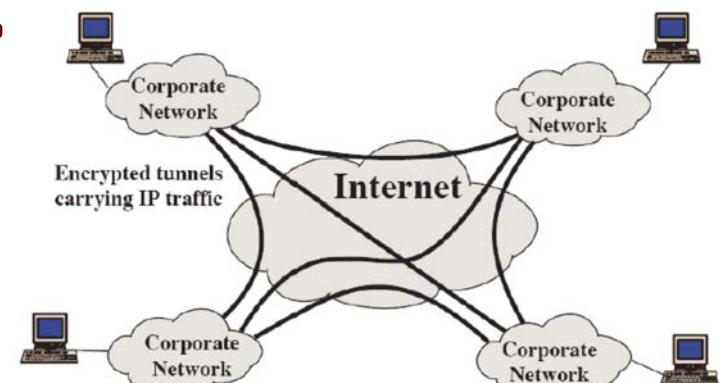
DOI: Domain of Interpretation

# Modos de IPSec

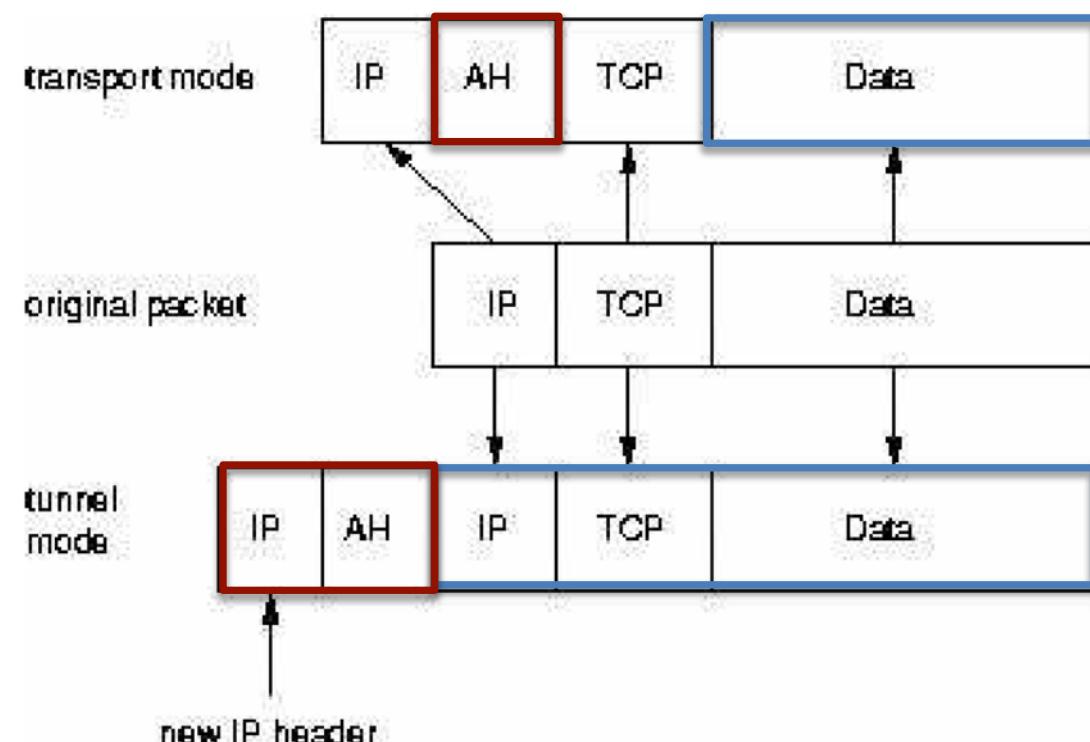
- Modo transporte:
  - Se usa normalmente para comunicaciones punto a punto entre dos hosts
  - Proporciona protección a la carga útil del paquete IP (**IP payload**)
    - es decir, a los protocolos de la capa superior - TCP, UDP, ICMP



- Modo túnel:
  - Se suele usar cuando los puntos a comunicar son gateways de seguridad o bien routers
  - Proporciona **protección a todo el paquete IP**



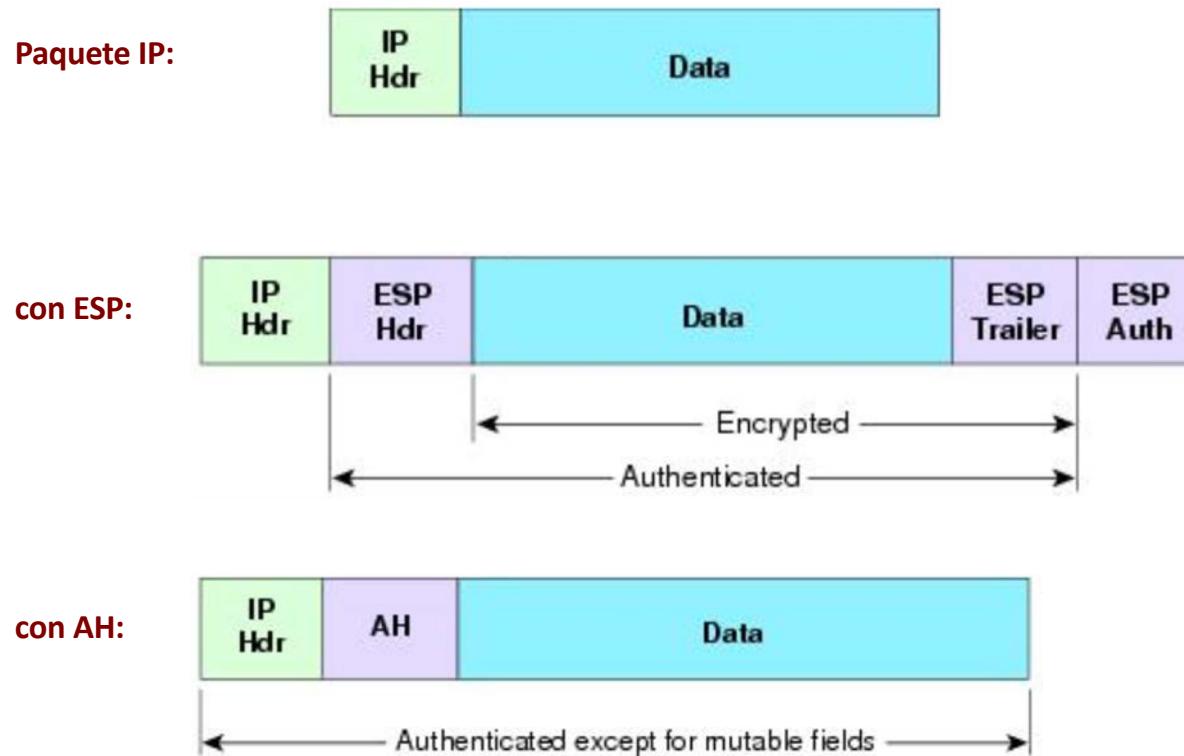
- Es decir, el modo túnel encapsula el datagrama IP dentro de un nuevo datagrama IP que emplea el protocolo IPsec. En cambio, el modo transporte IPsec sólo maneja la carga del datagrama IP, insertándose la cabecera IPsec entre la cabecera IP y la cabecera del protocolo de capas superiores



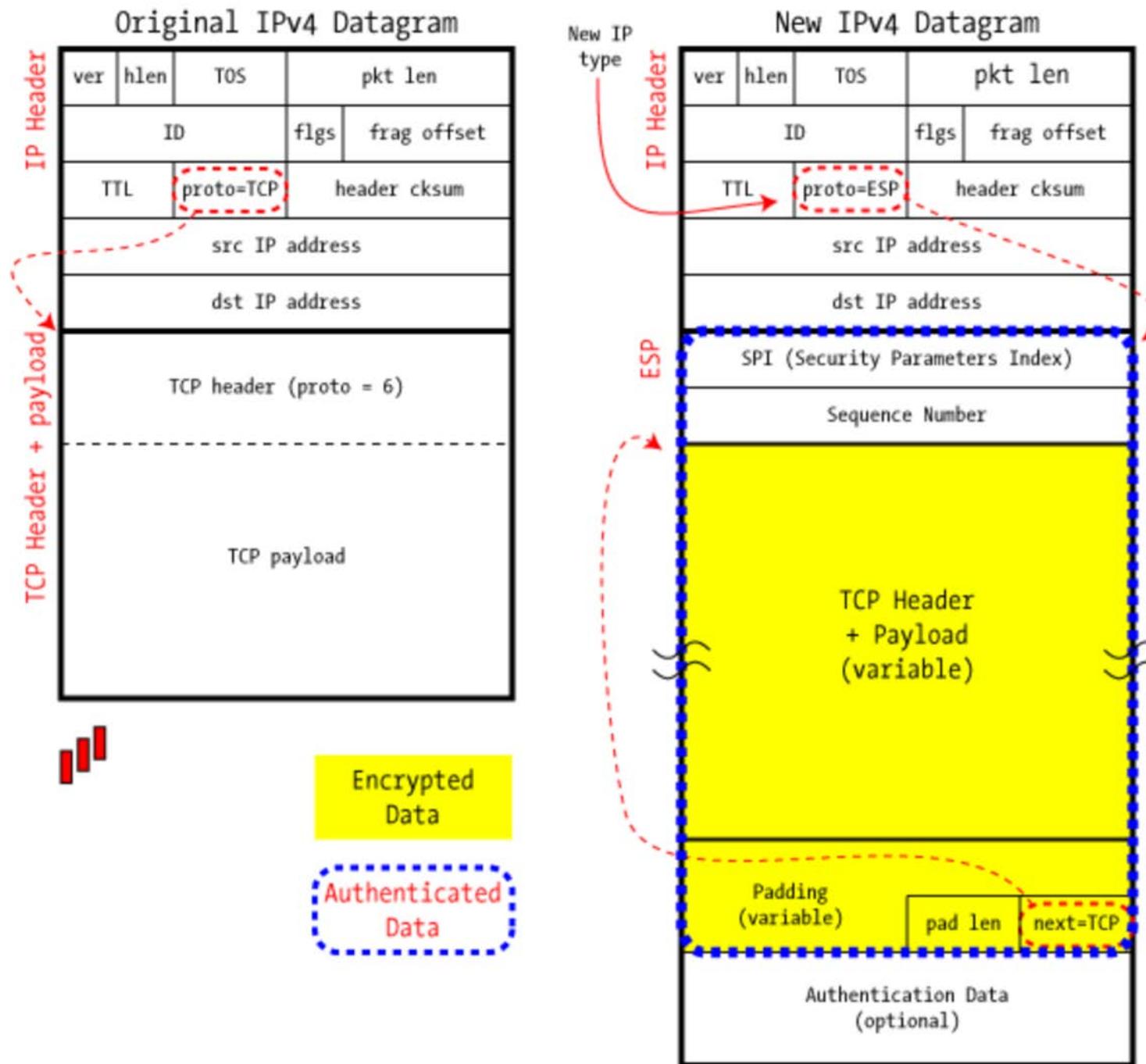
- Como puede verse en la figura anterior, la protección a todo el paquete IP del modo túnel se puede lograr:
  - i. añadiendo las cabecera AH o ESP al paquete IP original
  - ii. creando una cabecera IP nueva  
- De esta forma el paquete IP original (paquete interno) se “**encapsula**” y viaja por el túnel sin que ninguno de los routers intermedios pueda saber **ni el origen ni el destino final de los datos**

# Modo transporte

- En este modo de uso:
  - si se utiliza ESP: se cifra y opcionalmente autentica el payload, pero no la cabecera
  - si se utiliza AH: se autentica el payload y algunas porciones de la cabecera

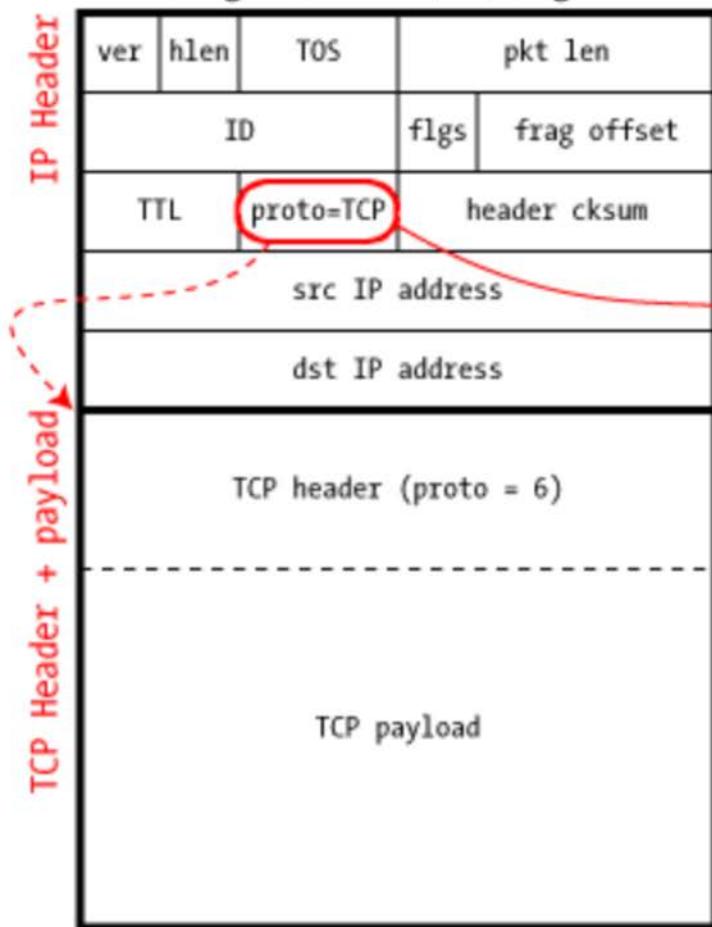


## IPSec in ESP Transport Mode

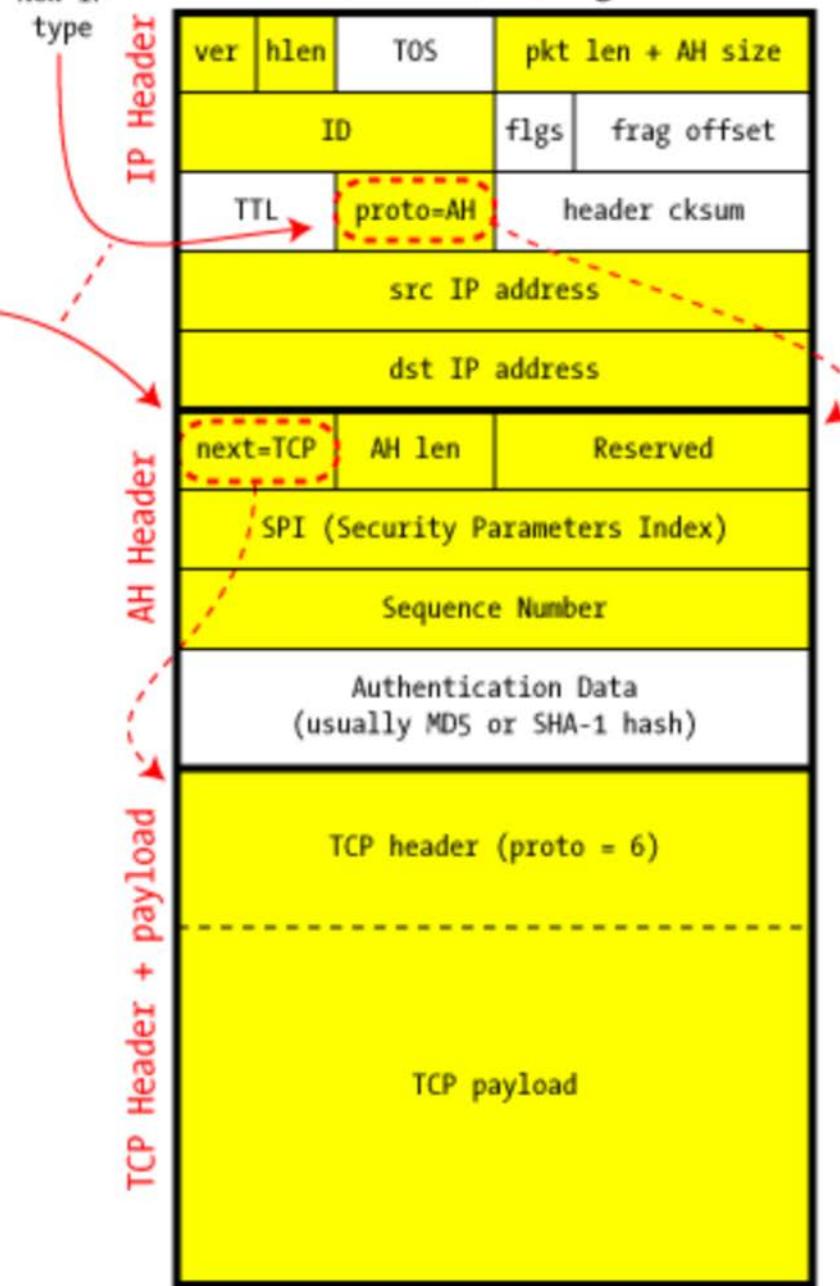


## IPSec in AH Transport Mode

Original IPv4 Datagram



New IPv4 Datagram



Protected by  
AH Auth Data

- La siguiente tabla muestra algunos de los códigos de protocolos de Internet, asignados por IANA (Internet Assigned Numbers Authority)

**Some IP protocol codes**

<b>Protocol code</b>	<b>Protocol Description</b>
1	ICMP — Internet Control Message Protocol
2	IGMP — Internet Group Management Protocol
4	IP within IP (a kind of encapsulation)
6	TCP — Transmission Control Protocol
17	UDP — User Datagram Protocol
41	IPv6 — next-generation TCP/IP
47	GRE — Generic Router Encapsulation (used by PPTP)
50	IPsec: ESP — Encapsulating Security Payload
51	IPsec: AH — Authentication Header



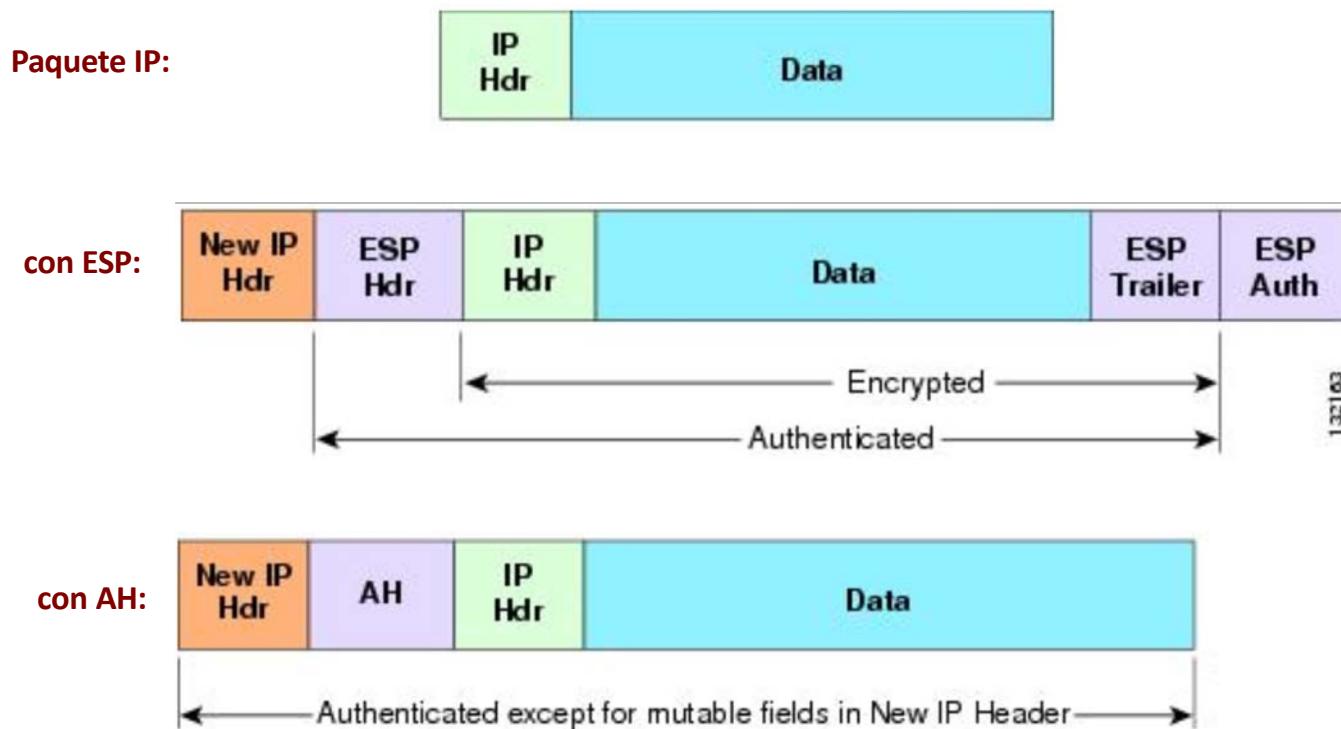
- Lista completa en:

<http://www.iana.org/assignments/protocol-numbers/protocol-numbers.xhtml>

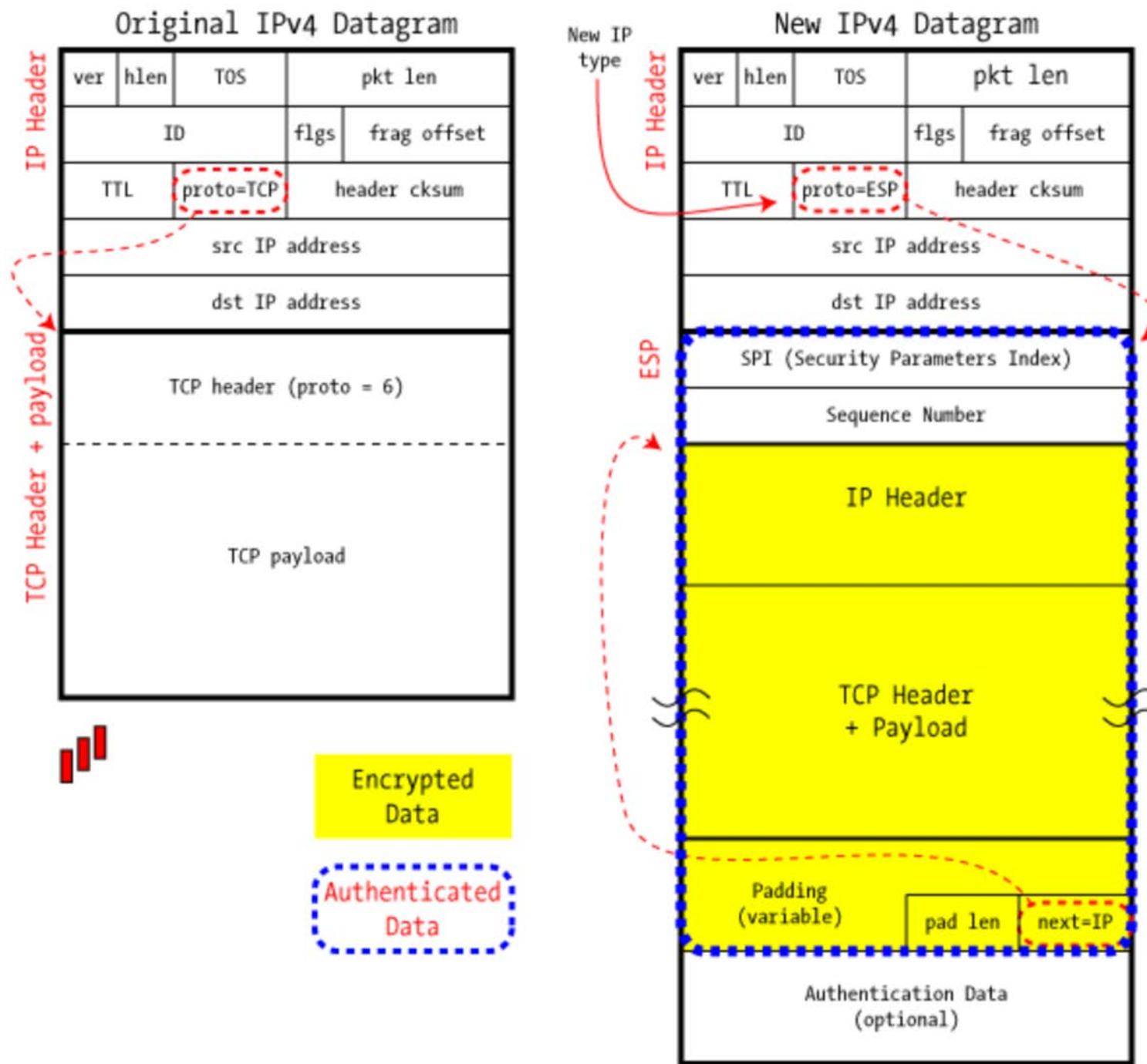
# Modo túnel

- En este modo de uso:

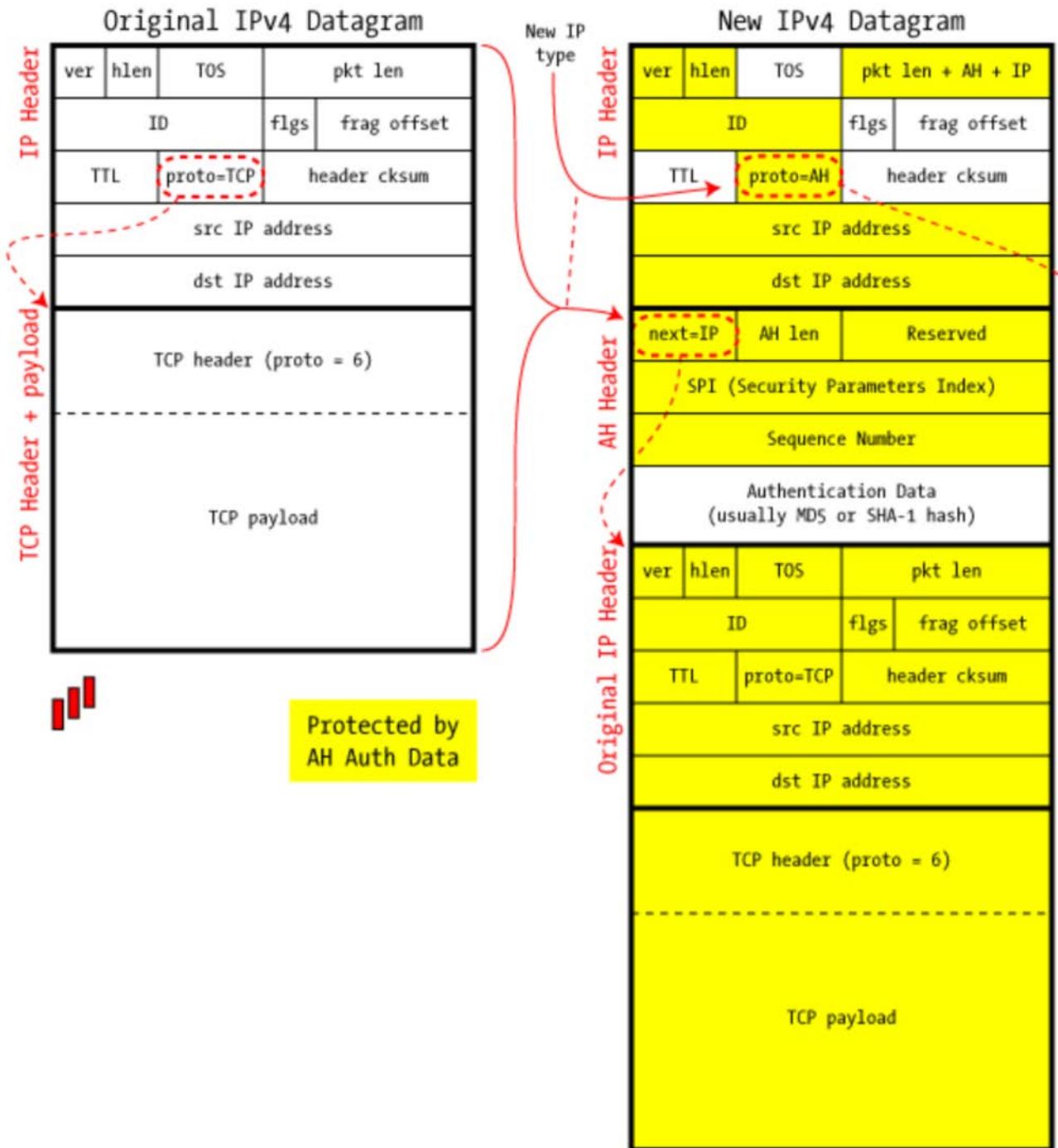
- si se usa ESP: se cifra y opcionalmente autentica todo el paquete IP original (paquete interno), incluyendo la cabecera de ese paquete original
- si se usa AH: se autentica todo el paquete original y algunas partes de la cabecera externa



## IPSec in ESP Tunnel Mode



## IPSec in AH Tunnel Mode



- Con IPSec se tiene que:
  - Asegurar la integridad de los mensajes por incluir en la cabecera del protocolo IPSec, el HMAC basados en MD5 o SHA
  - Permitir el uso general de algoritmos de cifrados estándar, como DES, 3DES, AES y Blowfish
  - Controlar un tipo de ataque de DoS basado en replay por aplicar un número de secuencia única de paquetes (sólo se aceptan paquetes que tienen un número actual de secuencia o posterior, las anteriores se descartan)
  - Encapsular y desencapsular paquetes IPSec. Para ello, se requiere el uso de algún mecanismo que almacene las claves secretas, los algoritmos de cifrado y autenticación, y las direcciones IP involucradas en la comunicación



## Asociaciones de seguridad (SA – Security Associations)

- Por tanto, cada SA define:
  - La dirección IP origen y destino
  - Determina el protocolo IPsec (AH o ESP)
    - A veces, se permite compresión de paquetes
  - El algoritmo y la clave secreta empleados por el protocolo IPsec
  - El índice de parámetro de seguridad (**SPI - Security Parameter Index**)
    - Es un número de 32 bits que identifica la asociación de seguridad
  - Sólo se protege un sentido
    - El emisor y el receptor deben aplicar la misma SA pero teniendo en cuenta el destino y el origen

- Las SAs se almacenan en una **base de datos de asociaciones de seguridad (SAD)**. Para cada entrada en la SAD existen varios campos. Entre ellos:
  - *Security Parameter Index (SPI)*: Es un valor de 32 bits para identificar a una SA particular
  - *AH Information*: algoritmo de autenticación, claves y otros parámetros relacionados con AH
  - *ESP Information*: algoritmo de cifrado y autenticación, claves y otros parámetros relacionados con ESP
  - *Lifetime of the SA*: Un intervalo o un contador después del cual habrá que reemplazar la SA
- Algunas SAD también definen:
  - El tipo de modo (túnel o transporte)

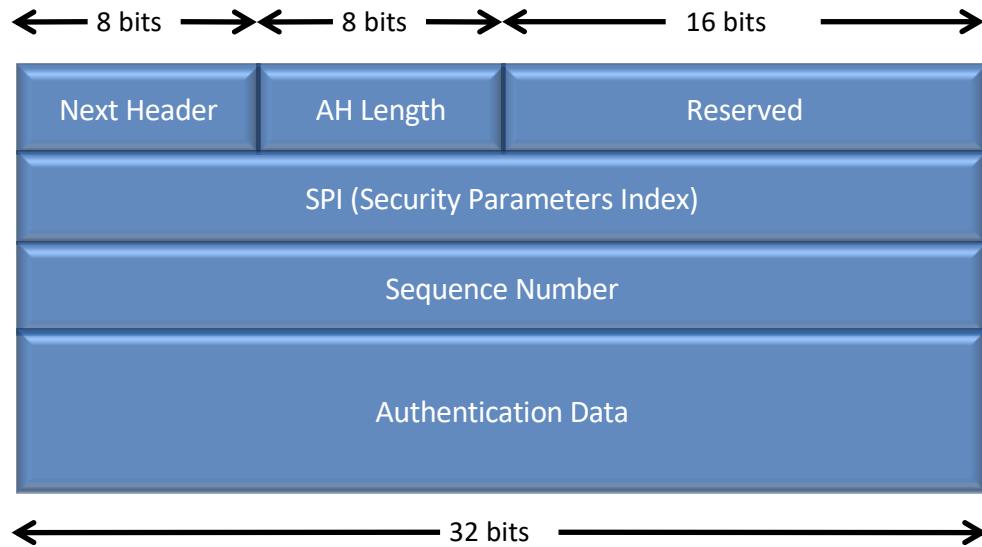
- Sin embargo, SA sólo especifica “el modo en el que se protegerá el dato IPSec”. Para definir “el modo en cómo va a viajar el tráfico entre dos puntos”, se requiere de una política de seguridad (**SP – Security Policy**) que se almacena en una **SPD (Security Policy Database)**
- Un SP define:
  - Las direcciones de origen y destino a proteger
    - En modo transporte, éstas serán las mismas direcciones que aquellas definidas en la SA
    - En modo túnel no tienen que ser las mismas
  - Los protocolos y puertos a proteger
    - Algunas implementaciones no permiten la definición de protocolos específicos a proteger. En este caso, se protege todo el tráfico
  - El modo de protección: túnel o transporte

- Un ejemplo de SPD es el siguiente:

<b>Protocol</b>	<b>Local IP</b>	<b>Port</b>	<b>Remote IP</b>	<b>Port</b>	<b>Action</b>	<b>Comment</b>
UDP	1.2.3.101	500	*	500	BYPASS	IKE
ICMP	1.2.3.101	*	*	*	BYPASS	Error messages
*	1.2.3.101	*	1.2.3.0/24	*	PROTECT: ESP intransport-mode	Encrypt intranet traffic
TCP	1.2.3.101	*	1.2.4.10	80	PROTECT: ESP intransport-mode	Encrypt to server
TCP	1.2.3.101	*	1.2.4.10	443	BYPASS	TLS: avoid double encryption
*	1.2.3.101	*	1.2.4.0/24	*	DISCARD	Others in DMZ
*	1.2.3.101	*	*	*	BYPASS	Internet

# Cabeceras AH y ESP

- Authentication Header - AH



Next Header (8 bits): Identifica el tipo de cabecera inmediatamente posterior a esta cabecera

AH Length (8 bits): Longitud de la Cabecera de Autenticación en palabras de 32 bits, menos 2 palabras

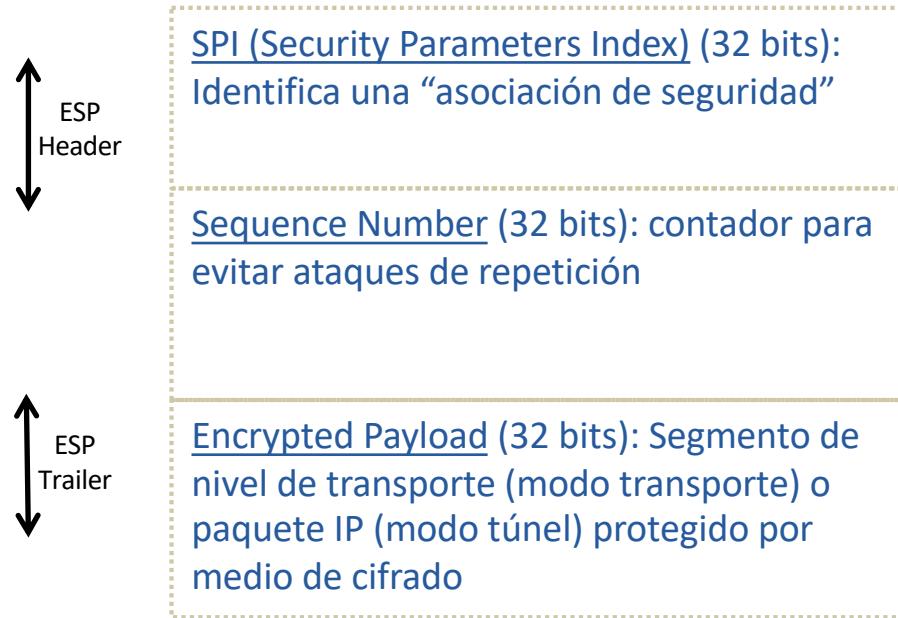
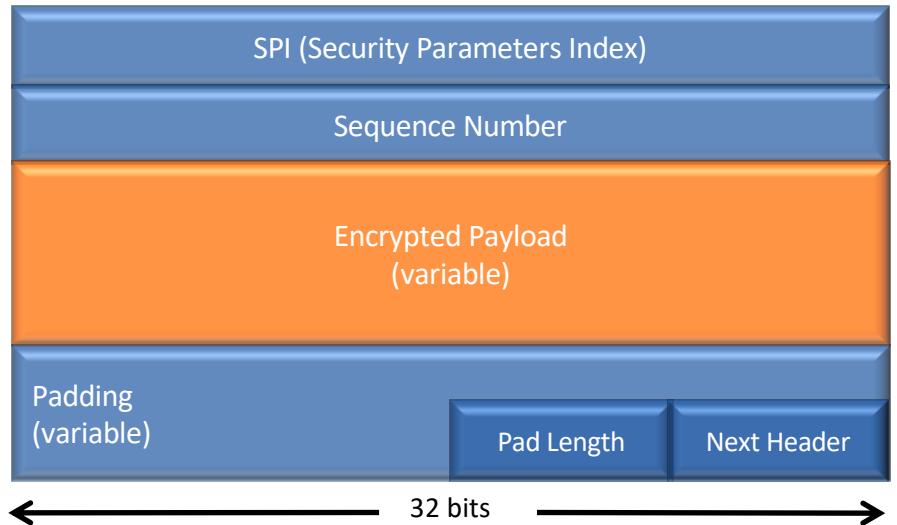
Reserved (16 bits): Para uso futuro

SPI (Security Parameters Index) (32 bits): Identifica una “asociación de seguridad”

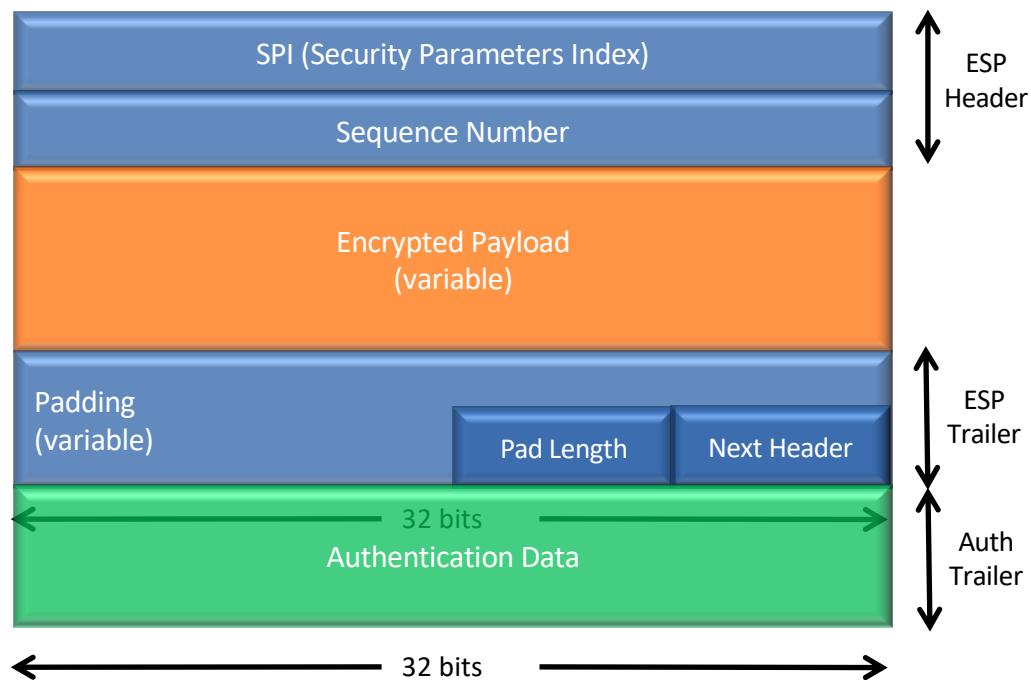
Sequence Number (32 bits): contador para evitar ataques de repetición

Authentication Data (variable): Campo de longitud variable (debe ser un número de palabras de 32 bits) que contiene el valor de comprobación de integridad (valor MAC) para este paquete

- Encapsulating Security Payload – ESP (sólo cifrado)



- Encapsulating Security Payload – ESP (cifrado + autenticación):



- Las distintas opciones de qué cabeceras usar y qué servicios queramos proporcionar:

	AH	ESP (encryption only)	ESP (encryption plus authentication)
Connectionless integrity	✓		✓
Data origin authentication	✓		✓
Rejection of replayed packets	✓	✓	✓
Confidentiality		✓	✓
Limited traffic flow confidentiality		✓	✓

## Ejemplo: modo transporte



192.168.1.100



192.168.2.100

- setkey es el comando para establecer la comunicación segura. Concretamente, lee las órdenes asociadas al SA o SAD de un fichero cuando **se invoca** con:
  - # `setkey -f /etc/ipsec.conf` (el fichero)
- Se comprueba la viabilidad de la acción setkey por lanzar:
  - # `setkey -D`
  - # `setkey -DP`

```

#!/usr/sbin/setkey -f

# Configuración para 192.168.1.100

# Vaciar las SAD y SPD
flush;
spdflush;

# Atención: Utilice estas claves sólo para pruebas
# ¡Debería generar sus propias claves!

# SAs para AH empleando claves largas de 128 bits
add 192.168.1.100 192.168.2.100 ah 0x200 -A hmac-md5 \
0xc0291ff014dccdd03874d9e8e4cdf3e6;
add 192.168.2.100 192.168.1.100 ah 0x300 -A hmac-md5 \
0x96358c90783bbfa3d7b196ceabe0536b;

# SAs para ESP empleando claves largas de 192 bits (168 + 24 paridad)
add 192.168.1.100 192.168.2.100 esp 0x201 -E 3des-cbc \
0x7aeaca3f87d060a12f4a4487d5a5c3355920fae69a96c831;
add 192.168.2.100 192.168.1.100 esp 0x301 -E 3des-cbc \
0xf6ddb555acfd9d77b03ea3843f2653255afe8eb5573965df;

# Políticas de seguridad
spdadd 192.168.1.100 192.168.2.100 any -P out ipsec
    esp/transport//require
    ah/transport//require;

spdadd 192.168.2.100 192.168.1.100 any -P in ipsec
    esp/transport//require
    ah/transport//require;

```

SPI



Se limpia el sistema de SAs y SPs antiguas (el SAD y SPD)

AH



SAs: AH y la clave (en ASCII, "", hexadecimal) para el HMAC

- A: alg. de autenticación
- E: alg. de cifrado
- C: alg. de comprensión

ESP



SAs: ESP y la clave para el cifrado

Protocolo y puerto



Dirección de la acción de la política

```

#!/usr/sbin/setkey -f

# Configuración para 192.168.2.100

# Vaciar las SAD y SPD
flush;
spdflush;

# Atención: Emplee estas claves sólo para pruebas
# ;Debería generar sus propias claves!

# SAs para AH empleando claves largas de 128 bits
add 192.168.1.100 192.168.2.100 ah 0x200 -A hmac-md5 \
0xc0291ff014dccdd03874d9e8e4cdf3e6;

add 192.168.2.100 192.168.1.100 ah 0x300 -A hmac-md5 \
0x96358c90783bbfa3d7b196ceabe0536b;

# SAs para ESP empleando claves largas de 192 bits (168 + 24 paridad)
add 192.168.1.100 192.168.2.100 esp 0x201 -E 3des-cbc \
0x7aeaca3f87d060a12f4a4487d5a5c3355920fae69a96c831;
add 192.168.2.100 192.168.1.100 esp 0x301 -E 3des-cbc \
0xf6ddb555acfd9d77b03ea3843f2653255afe8eb5573965df;

# Políticas de seguridad
spdadd 192.168.1.100 192.168.2.100 any -P in ipsec
    esp/transport//require
    ah/transport//require;

spdadd 192.168.2.100 192.168.1.100 any -P out ipsec
    esp/transport//require
    ah/transport//require;

```

## Ejemplo: modo túnel



192.168.1.100



172.16.1.0/24



192.168.2.100



172.16.2.0/24

```
#!/usr/sbin/setkey -f

# Vaciar las SAD y SPD
flush;
spdflush;

# SAs para ESP realizando cifrado con claves largas de 192 bit (168 + 24 paridad)
# y autenticación empleando claves largas de 128 bits
add 192.168.1.100 192.168.2.100 esp 0x201 -m tunnel -E 3des-cbc \
0x7aeaca3f87d060a12f4a4487d5a5c3355920fae69a96c831 \
-A hmac-md5 0xc0291ff014dccdd03874d9e8e4cdf3e6;

add 192.168.2.100 192.168.1.100 esp 0x301 -m tunnel -E 3des-cbc \
0xf6ddb555acfd9d77b03ea3843f2653255afe8eb5573965df \
-A hmac-md5 0x96358c90783bbfa3d7b196ceabe0536b;

# Políticas de seguridad
spdadd 172.16.1.0/24 172.16.2.0/24 any -P out ipsec
esp/tunnel/192.168.1.100-192.168.2.100/require;

spdadd 172.16.2.0/24 172.16.1.0/24 any -P in ipsec
esp/tunnel/192.168.2.100-192.168.1.100/require;
```

## IKE

- Como se puede observar en la figura anterior, cuando no existe la SA, hay que negociarla
- De eso se encarga el protocolo IKE - Internet Key Exchange
  - IKE se encarga de la autenticación de las partes de la comunicación y el establecimiento de la clave secreta
- IKE utiliza certificados X.509 para la autenticación, y el algoritmo de Diffie-Hellman para establecer la clave secreta
- IKE se basa a su vez en los protocolos Oakley e ISAKMP (Internet Security Association and Key Management Protocol)

## Estructura interna de la suite IPSEC:

AH = Authentication Header

API = Application Programming Interface

DOI = Domain of Interpretation

ESP = Encapsulated Security Payload

ISAKMP = Internet Security Association  
and Key Management Protocol

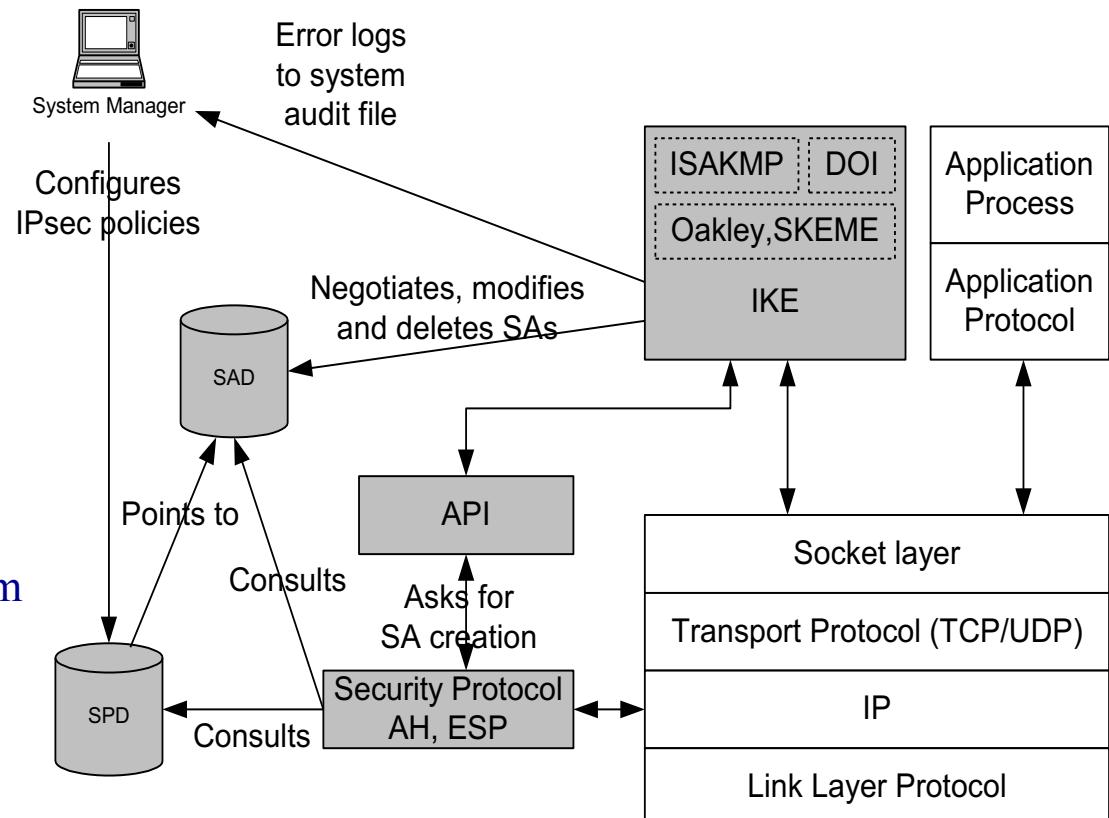
Oakley = Key Exchange Protocol

SA = Security Association

SAD = Security Association Database

SKEME = Secure Key Exchange Mechanism

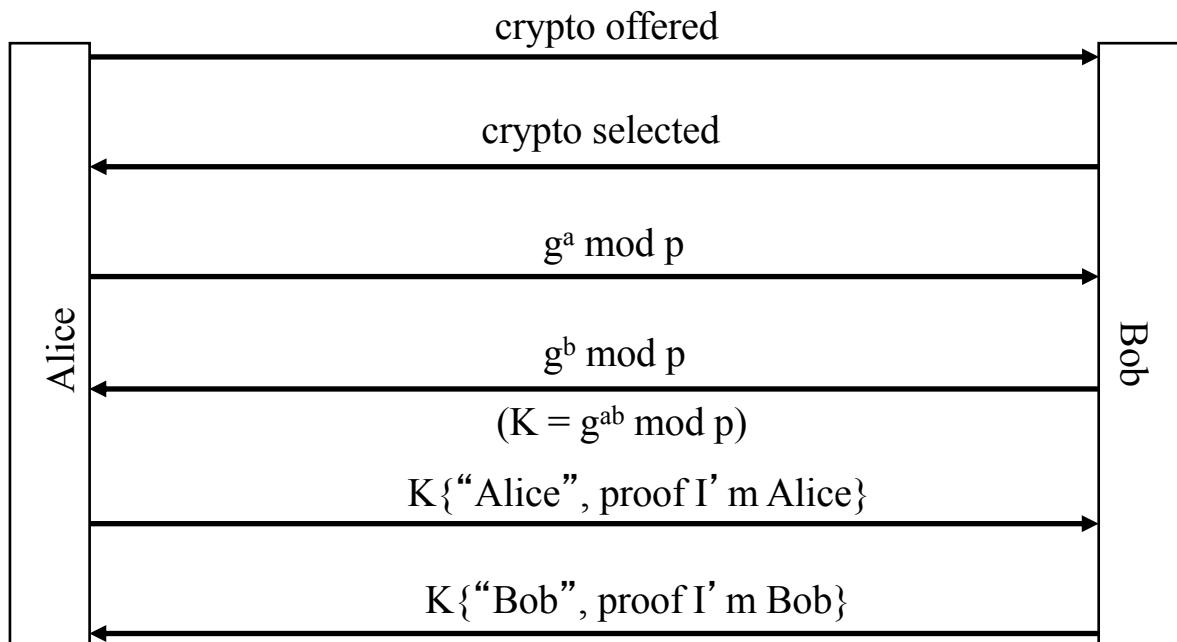
SPD = Security Policy Database



- Con ISAKMP, IKE funciona de la siguiente forma:
  - Fase 1: establece una SA ISAKMP previa a la SA de IPSec
    - La autenticación de las partes de la comunicación suele basarse en claves compartidas, claves RSA y certificados x.509
    - Esta fase soporta dos modos de autenticación: **agresivo y principal**
      - El modo agresivo proceso simple y sólo usa la mitad de los mensajes para finalizar el proceso rápidamente. Como no soporta la protección completa de las identidades de cada parte de la comunicación y transmite la identidad del cliente en claro, puede tender a ataques man-in-the-middle
  - Fase 2: el nuevo SA ISAKMP es empleado para **negociar y establecer los SAs de IPSec**
    - En esta fase el protocolo IKE intercambia propuestas de SAs, negocia asociaciones de seguridad basándose en el ISAKMP SA inicial, y establece la clave de sesión
    - Las claves de las SAs se derivan de las claves de la primera fase, los nonces y los SPI o usando un nuevo DH

## Fase 1 a Fase 2: Modo Principal

- El modo principal negocia una ISAKMP SA que se usa para crear las SAs de IPSec
- Tres pasos
  - Negociación de las SA
  - Diffie-Hellman e intercambio de nonce
  - Autenticación



## Fase 1 a Fase 2: Modo Agresivo

