

SEGURIDAD DE LA INFORMACIÓN

TEMA 5 – PARTE A

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

Introducción



- La expansión de la WWW en los 90 y su utilidad para realizar transacciones a nivel web, plantean nuevos riesgos de seguridad
 - en lado del cliente Web,
 - en el lado del servidor Web,
 - **la propia información entre el cliente y el servidor** ←

Amenazas		Consecuencias
Integridad	Modificación de los datos de usuario, troyano en el navegador, modificación del mensaje en tránsito, etc.	Pérdida de la información, afecta al dispositivo, desencadenamiento de otros ataques
Confidencialidad	Escuchas en el canal de comunicaciones, robo de información, robo de configuración de red, etc.	Pérdida de información y privacidad
Denegación de servicio	Interrumpir procesos de usuario, interrupciones en el lado del cliente o del servidor, etc.	Interrupción, inaccesibilidad
Autenticación	Suplantación de identidad, falsificación de datos	Creer en datos falsos o en entidades ilegítimas

Introducción



- 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-ielf-wts-01](#) (Approved)

Personnel

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

Mailing List

Address: www-security@nsmx.rutgers.edu
To Subscribe: www-security-request@nsmx.rutgers.edu
Archive: <http://www-ns.rutgers.edu/www-security>

Introducción



- 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)

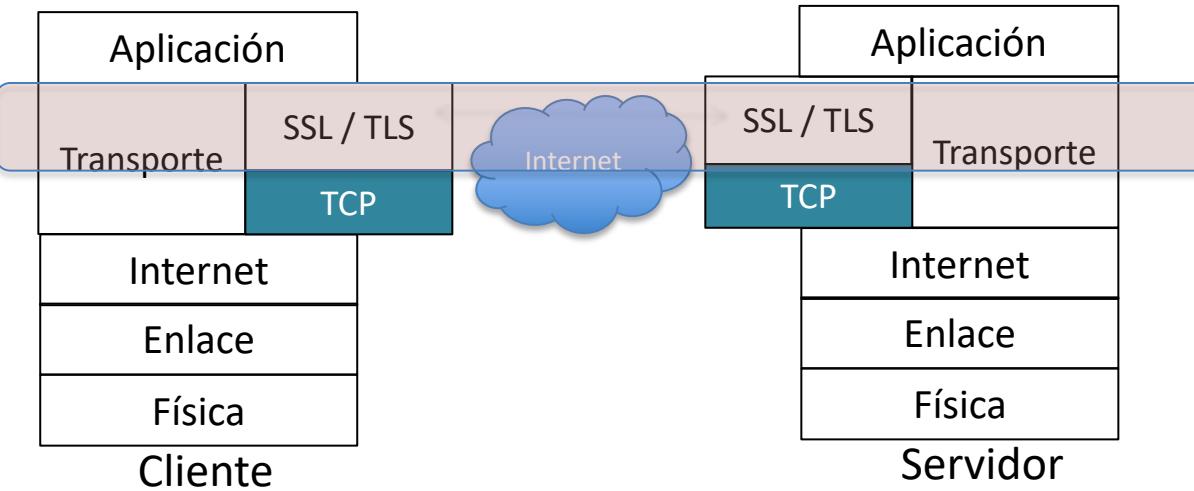
[Documents](#) | [Charter](#) | [History](#) | [List Archive »](#) | [Tools WG Page »](#)

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

Introducción



- 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 fiables** y transmitir datos a través de esas conexiones
- la última versión producida fue la v3.0



EVOLUCIÓN DE SSL/TLS

Introducción (un poco de historia ☺ ...)



Cuándo	Quién	Qué
A mediados de 1994	Netscape	SSL v1.0
A finales de 1994	Netscape	SSL v2.0
1995	Netscape	SSL v2.0 (actualizada)
1996	Netscape	SSL v3.0 (obsoleta)



Netscape

- **SSL es una iniciativa de Netscape para abordar los problemas de seguridad del Internet**
- Sustituye por completo el protocolo SHTTP, como un intento de proteger los canales de comunicación vía el Internet

Introducción (un poco de historia ☺ ...)



Cuándo	Quién	Qué
A mediados de 1994	Netscape	SSL v1.0
A finales de 1994	Netscape	SSL v2.0
1995	Netscape	SSL v2.0
1996	Netscape	SSL v3.0 (obsoleta)
1999	IETF	TLS v1.0 (SSL v3.1) – RFC: 2246

MISIÓN

This document specifies Version 1.0 of the Transport Layer Security (TLS) protocol. The TLS protocol provides communications privacy over the Internet. The protocol allows client/server applications to communicate in a way that is designed to prevent eavesdropping, tampering, or message forgery.

- **TLS (Transport Layer Security) 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:

Network Working Group
Request for Comments: 2246 ← RFC
Category: Standards Track

T. Dierks
Ceritcom
C. Allen
Ceritcom
January 1999

The TLS Protocol
Version 1.0 ← año

Status of this Memo ← versión

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Copyright Notice
Copyright (C) The Internet Society (1999). All Rights Reserved.

Abstract
This document specifies Version 1.0 of the Transport Layer Security (TLS) protocol. The TLS protocol provides communications privacy over the Internet. The protocol allows client/server applications to communicate in a way that is designed to prevent eavesdropping, tampering, or message forgery. ← descripción

Table of Contents

1. Introduction	3
2. Goals	4
3. Goals of this document	5
4. Presentation language	5
4.1. Basic block size	6
4.2. Miscellaneous	6
4.3. Vectors	6
4.4. Numbers	7
4.5. Enumerateds	7
4.6. Constructed types	8
4.6.1. Variants	9
4.7. Cryptographic attributes	10
4.8. Constants	11
5. HMAC and the pseudorandom function	11
6. The TLS Record Protocol	13
6.1. Connection states	14

Introducción (un poco de historia ☺ ...)



Cuándo	Quién	Qué
A mediados de 1994	Netscape	SSL v1.0
A finales de 1994	Netscape	SSL v2.0
1995	Netscape	SSL v2.0
1996	Netscape	SSL v3.0 (obsoleta)
1999	IETF	TLS v1.0 (SSL v3.1) – RFC: 2246

- **TLS (Transport Layer Security) 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”

- Sin embargo, para entender estas diferencias es necesario primero ver cómo funciona el protocolo SSL (el cual es equivalente a TLS)

Introducción (un poco de historia ☺ ...)



Cuándo	Quién	Qué
A mediados de 1994	Netscape	SSL v1.0
A finales de 1994	Netscape	SSL v2.0
1995	Netscape	SSL v2.0
1996	Netscape	SSL v3.0 (obsoleta)
1999	IETF	TLS v1.0 (SSL v3.1) – RFC: 2246
2006	IETF	TLS v1.1 (SSL v3.2) – RFC: 4346
2008	IETF	TLS v1.2 (SSL v3.3) – RFC: 5246
2014	IETF	TLS v1.3 (draft 1) – (SSL v3.4) – RFC: 8446
2018	IETF	TLSv 1.3

[RFC Home] [TEXT|PDF|HTML] [Tracker] [IPR] [Errata] [Info page]

Obsoleted by: 5246, 6066
Updated by: 5746
Network Working Group
Request for Comments: 4366
Obsoletes: 3546
Updates: 4346
Category: Standards Track

PROPOSED STANDARD
Errata Exist
S. Blake-Wilson
M. Nystrom
RSA Security
D. Hopwood
Independent Consultant
J. Mukkavilli
Transactionware
T. Wright
Vodafone
April 2006

Transport Layer Security (TLS) Extensions

Status of This Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Copyright Notice

Copyright (C) The Internet Society (2006).

Abstract

This document describes extensions that may be used to add functionality to Transport Layer Security (TLS). It provides both generic extension mechanisms for the TLS handshake client and server hellos, and specific extensions using these generic mechanisms.

The extensions may be used by TLS clients and servers. The extensions are backwards compatible; communication is possible between TLS clients that support the extensions and TLS servers that do not support the extensions, and vice versa.

[RFC Home] [TEXT|PDF|HTML] [Tracker] [IPR] [Errata] [Info page]

Obsoleted by: 8446
Updated by: 5746, 5878, 6176, 7465, 7507, 7568, 7627, 7685, 7985, 7919, 8447, 9155
Network Working Group
Request for Comments: 5246
Obsoletes: 3268, 4346, 4366
Updates: 4492
Category: Standards Track

PROPOSED STANDARD
Errata Exist
T. Dierks
Independent
E. Rescorla
RTFM, Inc.
August 2008

The Transport Layer Security (TLS) Protocol Version 1.2

Status of This Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Abstract

This document specifies Version 1.2 of the Transport Layer Security (TLS) protocol. The TLS protocol provides communications security over the Internet. The protocol allows client/server applications to communicate in a way that is designed to prevent eavesdropping, tampering, or message forgery.

Fuente: <https://www.rfc-editor.org/rfc/rfc4366>

Fuente: <https://www.rfc-editor.org/rfc/rfc5246>

[RFC Home] [TEXT|PDF|HTML] [Tracker] [IPR] [Errata] [Info page]

PROPOSED STANDARD
Errata Exist
E. Rescorla
Mozilla
August 2018

Internet Engineering Task Force (IETF)
Request for Comments: 8446
Obsoletes: 5077, 5246, 6961
Updates: 5705, 6066
Category: Standards Track
ISSN: 2070-1721

The Transport Layer Security (TLS) Protocol Version 1.3

Abstract

This document specifies version 1.3 of the Transport Layer Security (TLS) protocol. TLS allows client/server applications to communicate over the Internet in a way that is designed to prevent eavesdropping, tampering, and message forgery.

This document updates RFCs 5075 and 6066, and obsoletes RFCs 5077, 5246, and 6961. This document also specifies new requirements for TLS 1.2 implementations.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in [Section 2 of RFC 7841](#).

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at <https://www.rfc-editor.org/info/rfc8446>.

SSL - Secure Sockets Layer



- El protocolo SSL es un **protocolo cliente/servidor** que proporciona un conjunto de servicios de seguridad para garantizar prevención contra:
 - “ataques de escucha (*eavesdropping*)”,
 - “ataques de manipulación (*tampering*)” y
 - “ataques de falsificación (*forgery*)”

This document specifies Version 1.0 of the Transport Layer Security (TLS) protocol. The TLS protocol provides communications privacy over the Internet. The protocol allows client/server applications to communicate in a way that is designed to prevent eavesdropping, tampering, or message forgery.

Fuente: <https://www.ietf.org/rfc/rfc2246.txt>

- Esto significa que para abordar:
 - Ataques de escuchas es necesario: **confidencialidad**
 - Ataques de manipulación: **integridad**
 - Ataques de falsificación (de ID): **autenticación**

SSL - Secure Sockets Layer



- Efectivamente el **protocolo SSL** tenía como misión garantizar:
 - **Autenticación**
 - tanto de las entidades origen y destino de los datos
 - usando, por ejemplo, certificados (aunque opcional) y MAC
 - **Confidencialidad** de la conexión
 - **Integridad** de la conexión
- Para ello, el **protocolo SSL** tenía también como misión garantizar una conexión segura entre un cliente y un servidor,
 - por lo que consiste básicamente en un pase de mensajes previos antes de iniciar la conexión



SSL - Secure Sockets Layer



- Más concretamente, SSL (v3.0) empleaba:
 - **criptografía de clave pública**
 - autenticación de las entidades
 - establecimiento de clave
 - **criptografía simétrica**
 - autenticación de los datos (mensajes)
 - cifrado de los mensajes
- Para el intercambio de clave SSLv3.0 aplicaba:
 - RSA
 - Diffie-Hellmann
 - Fortezza KEA (sólo hasta SSL v3.0)
- A pesar de la utilización de criptografía de clave pública,
SSL no proporciona el servicio de no-repudio
 - ni no-repudio de origen ni no-repudio de entrega

**Criptografía
Híbrida ☺**

SSL - Secure Sockets Layer

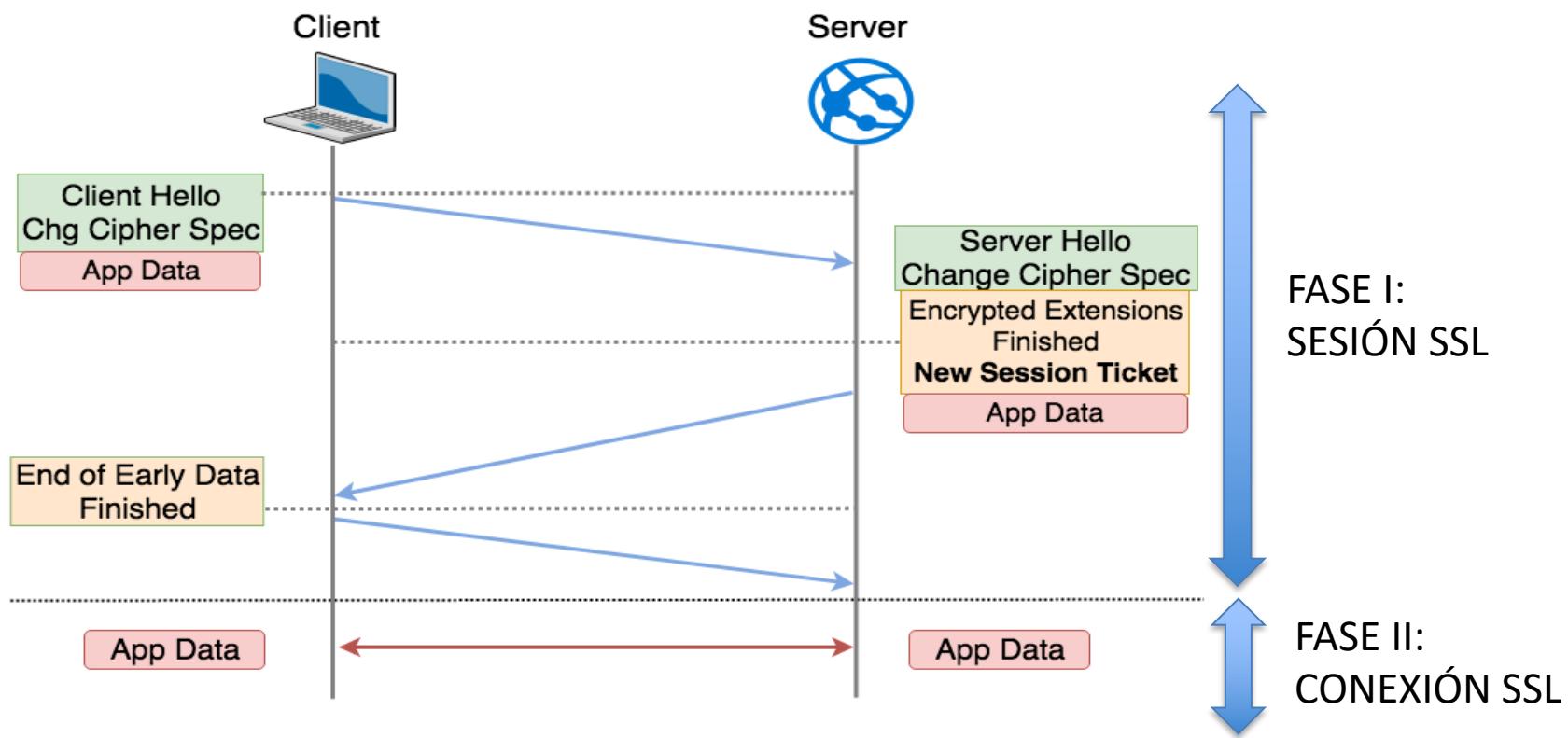


- Una ventaja del protocolo SSL es que es independiente del protocolo de la capa de aplicación
 - es decir, cualquier protocolo de aplicación basado en TCP se puede beneficiar de SSL (éste le dota de los servicios de seguridad mencionados)

Protocolo	Descripción	Puerto
https	HTTP sobre SSL/TLS	443
ldaps	LDAP sobre SSL/TLS	636
ftps-data	FTP Data sobre SSL/TLS	989
ftps	FTP Control sobre SSL/TLS	990
telnets	Telnet sobre SSL/TLS	992
Impas	IMAP4 sobre SSL/TLS	993
Pop3s	POP3 sobre SSL/TLS	995
...



- 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



SSL - Secure Sockets Layer



- El ámbito de la funcionalidad de SSL es, por tanto, doble:
 1. **Establecer una conexión segura** entre los puntos que se comunican, y garantizar conexión:
 - Autenticada
 - Confidencial entre un cliente y un servidor (“*conexión segura punto a punto*”)
 2. **Utilizar esa conexión para transmitir** de forma segura los datos del nivel de aplicación entre el emisor y el receptor



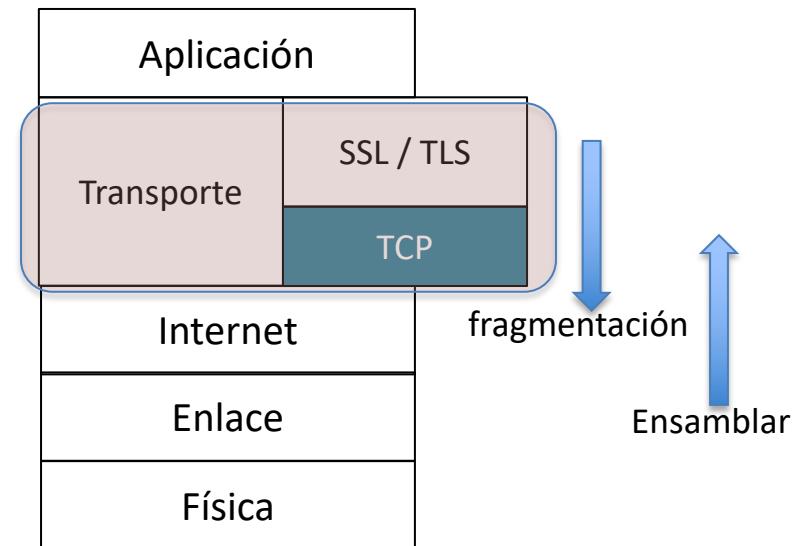
Negociación de
parámetros de seguridad



SSL - Secure Sockets Layer



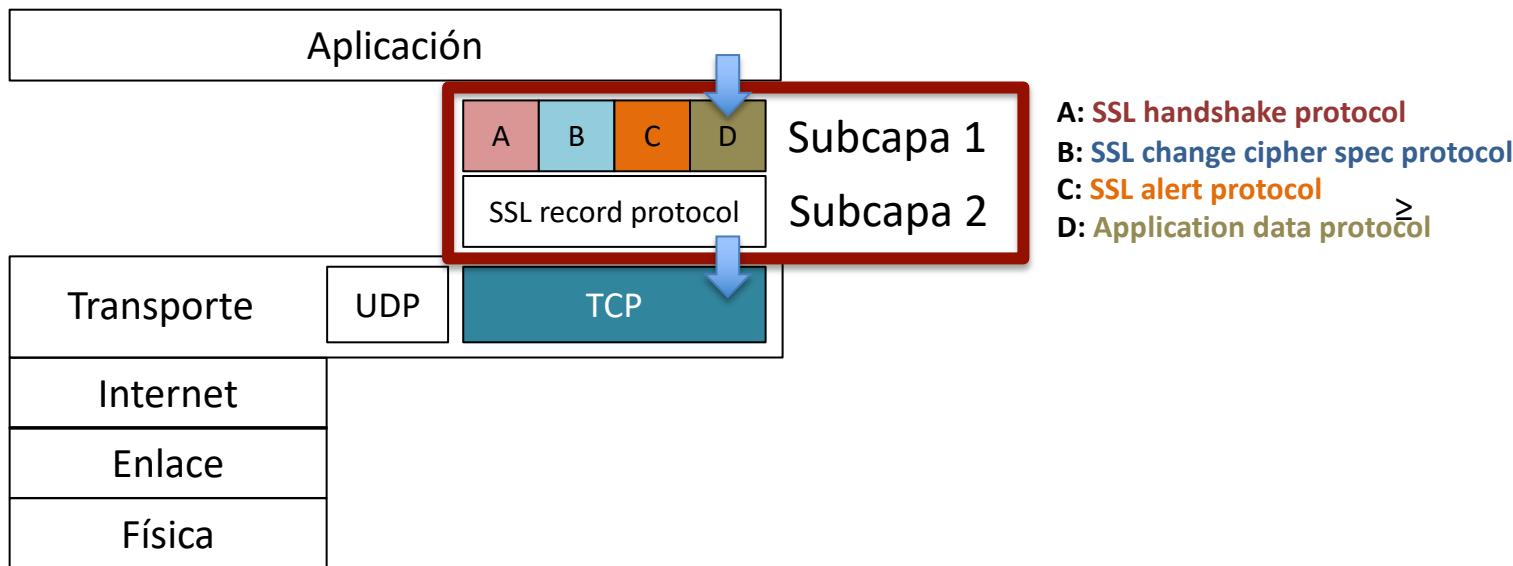
- Para una conexión segura es necesario que SSL considere las misiones propias de la capa de transporte, y especialmente aquellas asignadas a TCP
 - Fragmentar los paquetes de forma que siga una transmisión ordenada entre el origen y el destino
 - Ensamblar los paquetes de forma que se pueda construir los mensaje de manera ordenada
- 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**



SSL - Secure Sockets Layer



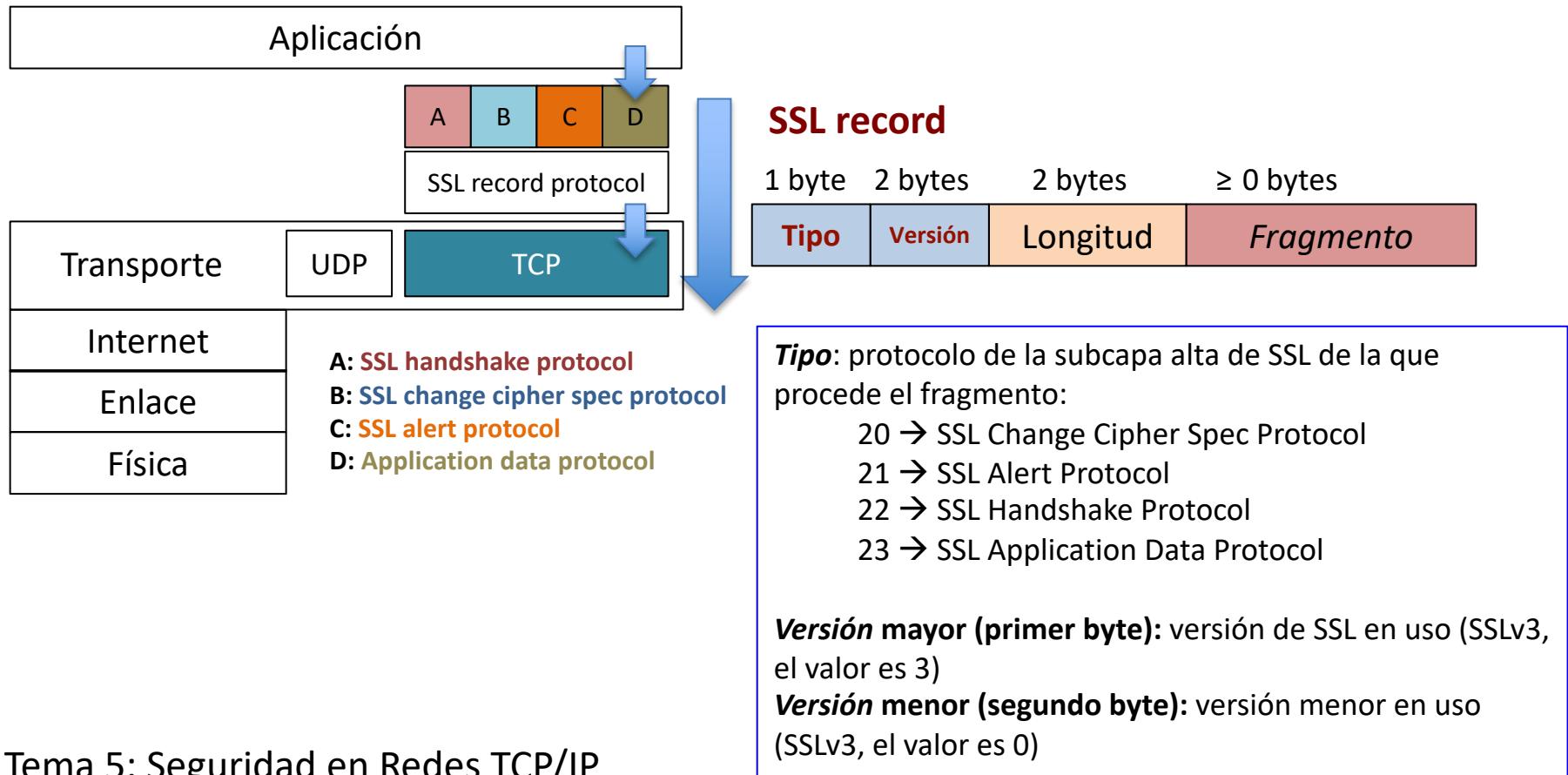
- Para llevar a cabo esa doble funcionalidad, SSL consta de dos subcapas y varios subprotocolos, como se observa en la siguiente figura:



SSL - Secure Sockets Layer



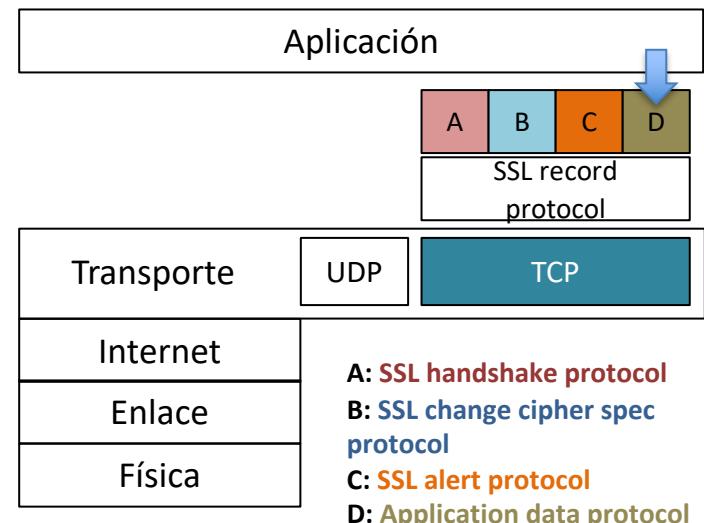
- Para llevar a cabo esa doble funcionalidad, SSL consta de dos subcapas y varios subprotocolos, como se observa en la siguiente figura:



SSL - Secure Sockets Layer



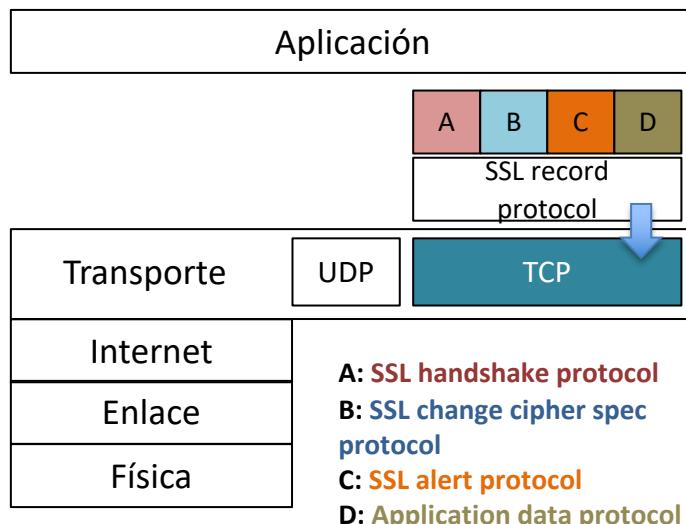
- La subcapa alta contiene:
 - **(22) SSL Handshake Protocol:** permite que los puntos de comunicación:
 - se autentiquen mutuamente, y que además,
 - se negocien un **cipher suite** y
 - (opcionalmente) un método de compresión
 - **(20) SSL Change Cipher Spec Protocol:** permite a los puntos de comunicación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



SSL - Secure Sockets Layer



- La subcapa baja contiene:
 - **SSL Record Protocol:** fragmenta los datos de la capa de aplicación y los procesa de forma individual



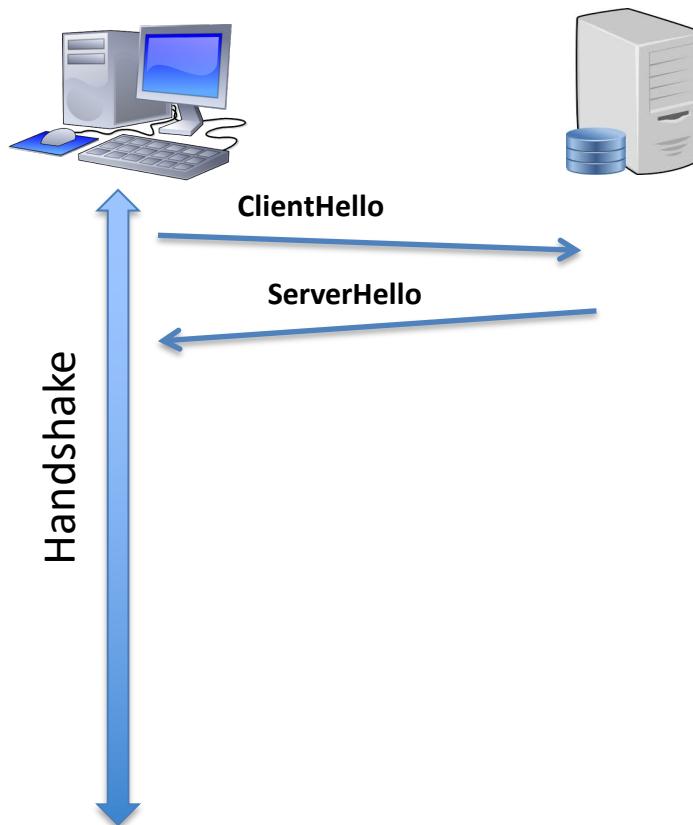
SSL record

1 byte	2 bytes	2 bytes	≥ 0 bytes
Tipo	Versión	Longitud	<i>Fragmento</i>

SUBCAPA 1

(SESIÓN ENTRE LOS PUNTOS)

SSL - Secure Sockets Layer

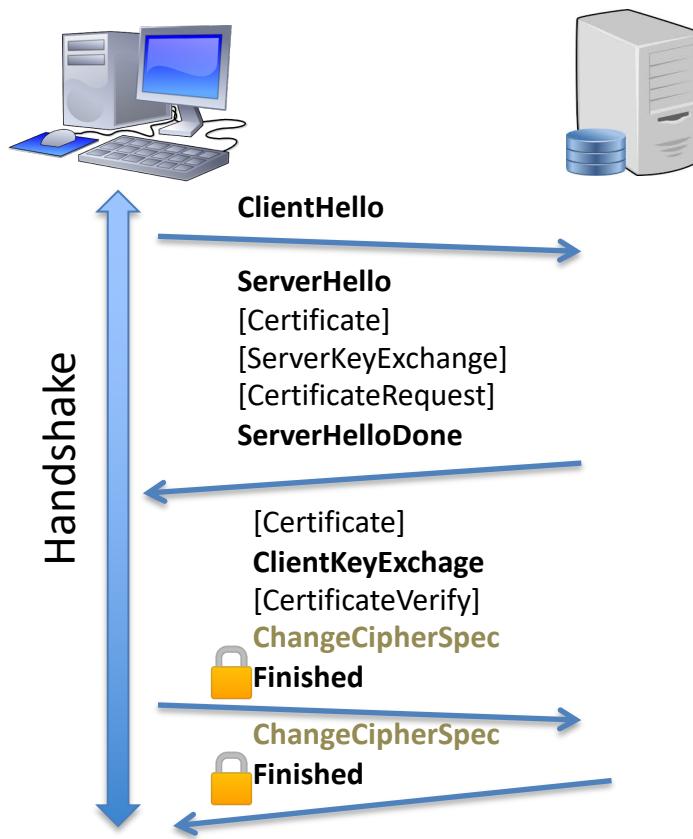


[] – significa opcional

Fase 1: se establece los parámetros de seguridad, incluyendo la versión del protocolo, la sesión de ID, el **suite de cifrado**, un número aleatorio y el método de comprensión (opcional, e incluso, en SSLv3.0 no se especifica, pero sí en TLSv1.2, pero ya en la TLSv1.3 vuelve a ser opcional)

Name of the Cipher Suite	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_KEA_WITH_NULL_SHA</i>	FORTEZZA_KEA	NULL	SHA
<i>SSL_FORTEZZA_KEA_WITH_FORTEZZA_CBC_SHA</i>	FORTEZZA_KEA	FORTEZZA_CBC	SHA
<i>SSL_FORTEZZA_KEA_WITH_RC4_128_SHA</i>	FORTEZZA_KEA	RC4_128	SHA

SSL - Secure Sockets Layer



Fase 1: se establece los parámetros de seguridad, incluyendo la versión del protocolo, la sesión de ID, el **suite de cifrado**, un número aleatorio y el método de comprensión (opcional, e incluso, en SSLv3.0 no se especifica, pero sí en TLSv1.2, pero ya en la TLSv1.3 vuelve a ser opcional)

Fase 2: el servidor **puede enviar un certificado (opcional)**, intercambia los parámetros necesarios para gestionar la clave secreta y puede solicitar un certificado al cliente

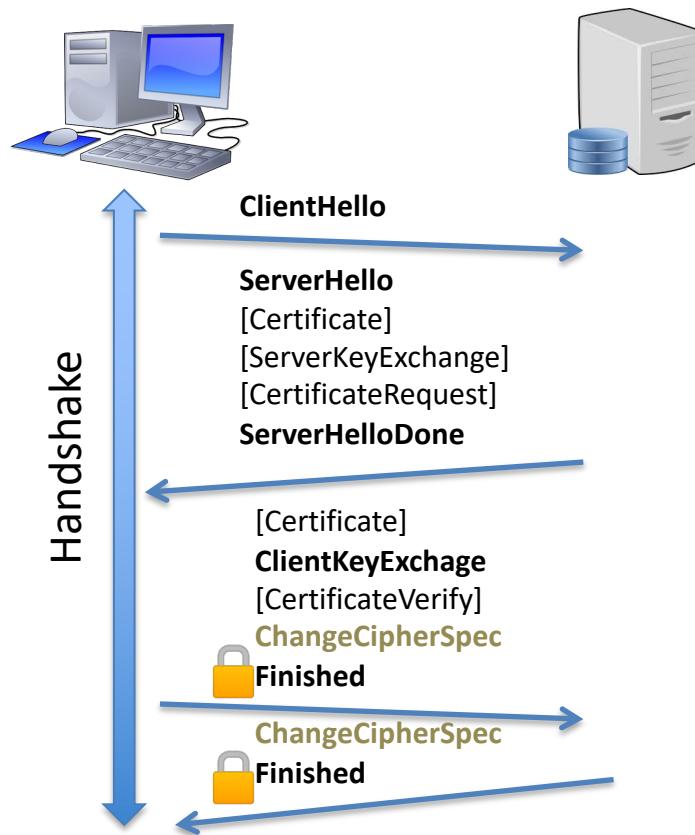
Fase 3: el cliente envía el certificado si es solicitado y los parámetros de seguridad necesarios para computar la clave de sesión

Si el cliente envía el certificado, entonces necesita enviar un certificado firmado para la verificación de la entidad origen

Fase 4: Se establece el suite de cifrado y se termina el proceso de handshake

INTERCAMBIO DE CLAVES

SSL - Secure Sockets Layer



¿Cómo se crea la clave de sesión?

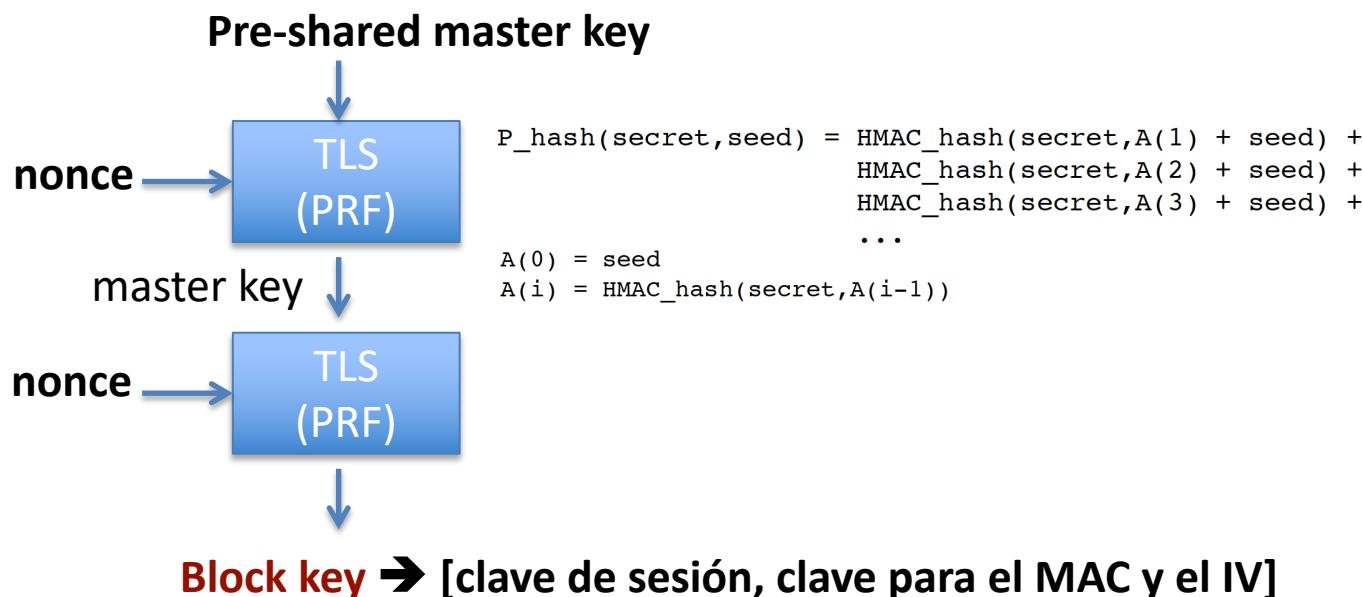
Sin embargo, para proteger la conexión entre el cliente y el servidor necesitamos calcular:

1. la **clave de sesión**
2. una **clave para el MAC**
3. un **IV** para computar el modo de operación

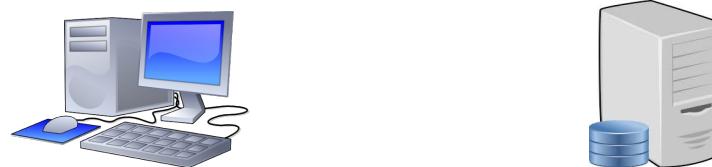
SSL - Secure Sockets Layer



- Los parámetros anteriores se calculan en función de varios parámetros de seguridad:
 - una semilla**
 - un valor aleatorio** (nonce)
 - una función pseudorandom** (PRF –pseudorandom function)



SSL - Secure Sockets Layer - intercambio de claves



Client_random

Paso 1:

- generar números aleatorios (la **salt** para generar después los parámetros de seguridad comentados anteriormente),
- establecer el resto de parámetros de seguridad (ej. tipo de algoritmos de intercambio de clave),
- enviar toda esta información a la otra parte

ClientHello = ({TLS_DHE_RSA_WITH_AES_256_CBC_SHA...}, client_random, ...)

Server_random

ServerHello = (TLS_DHE_RSA_WITH_AES_256_CBC_SHA, server_random)

[Certificate]
[ServerKeyExchange]
[CertificateRequest]
ServerHelloDone

[Certificate]
ClientKeyExchange
[CertificateVerify]
ChangeCipherSpec



Finished

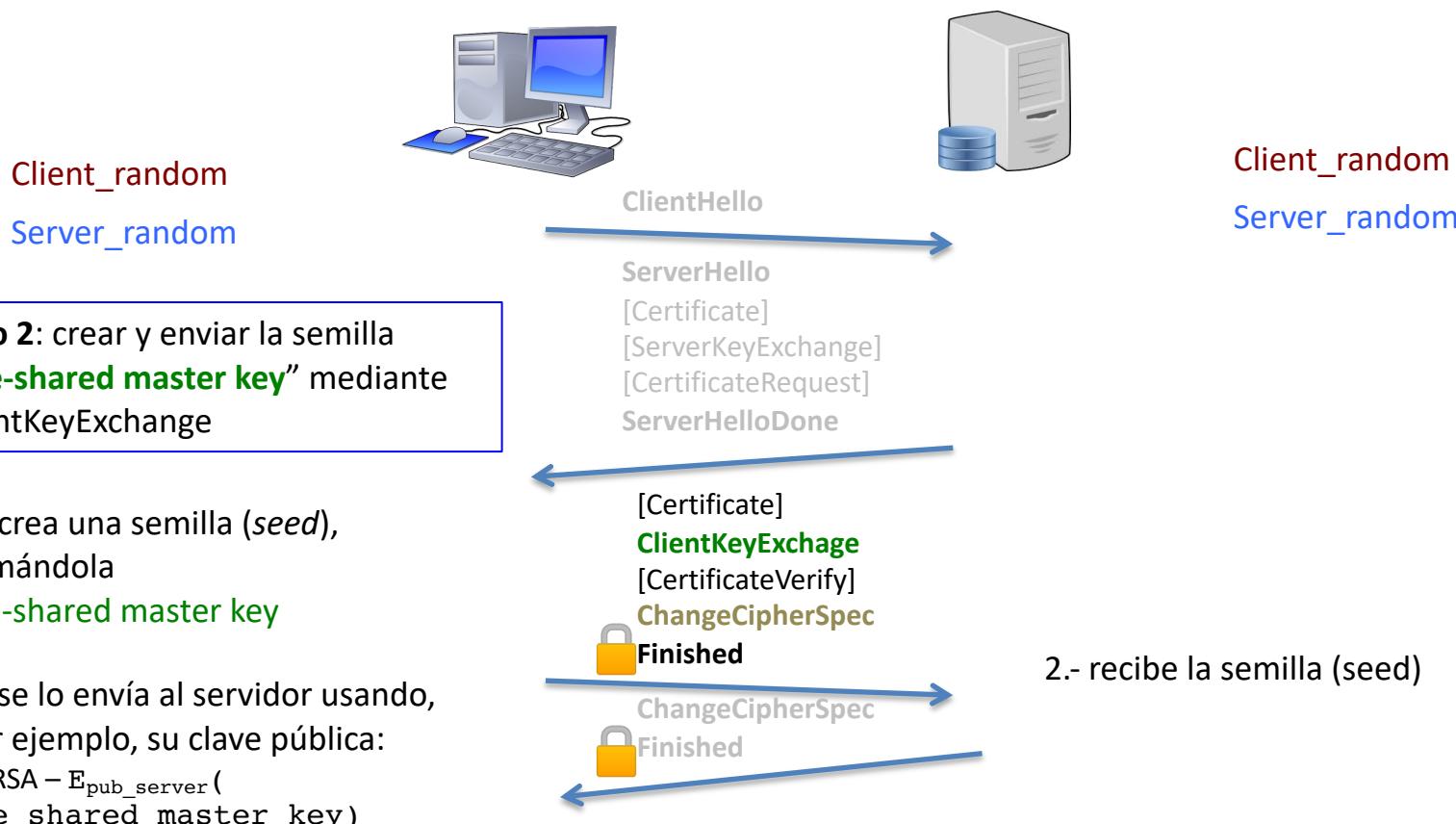


ChangeCipherSpec

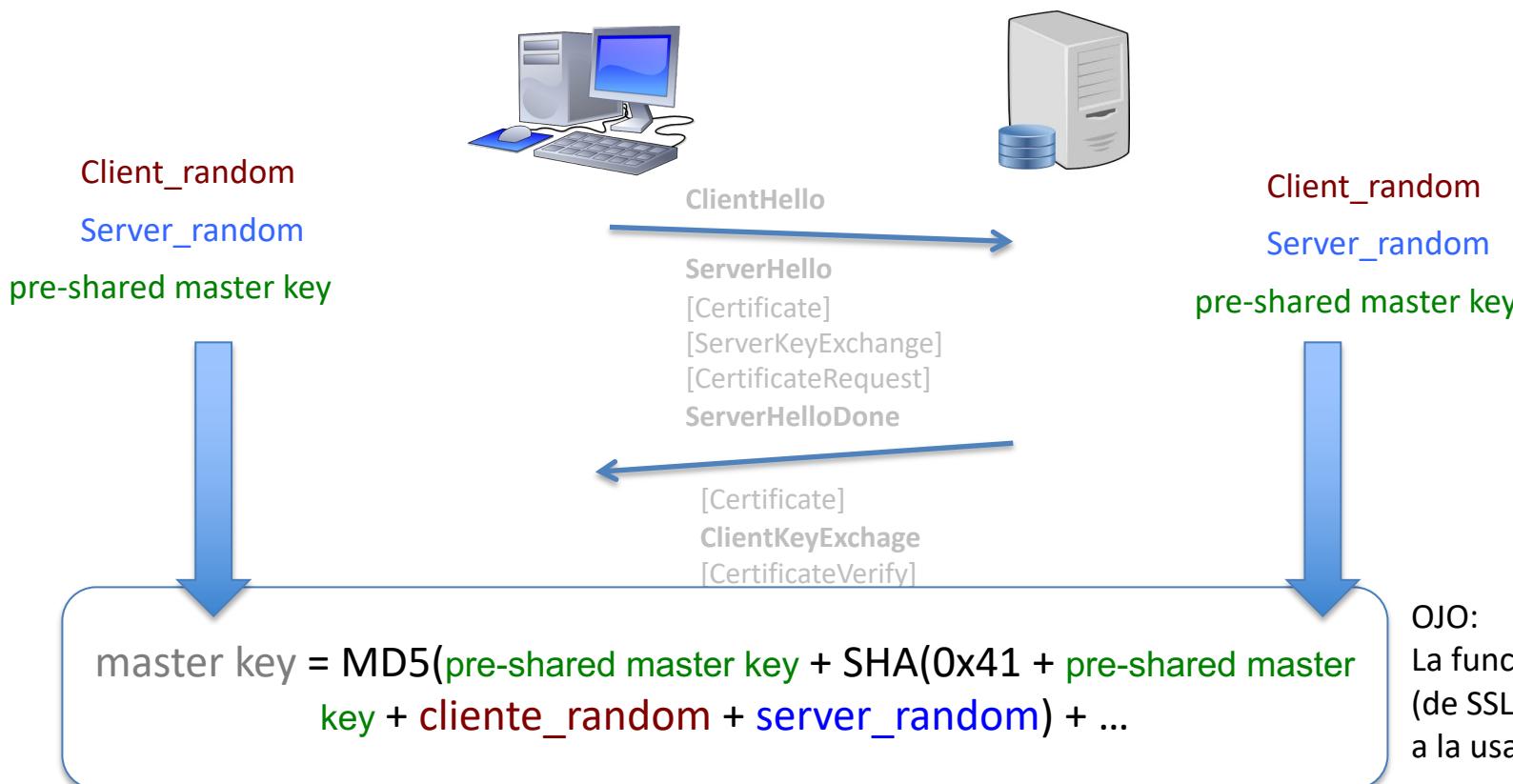


Finished

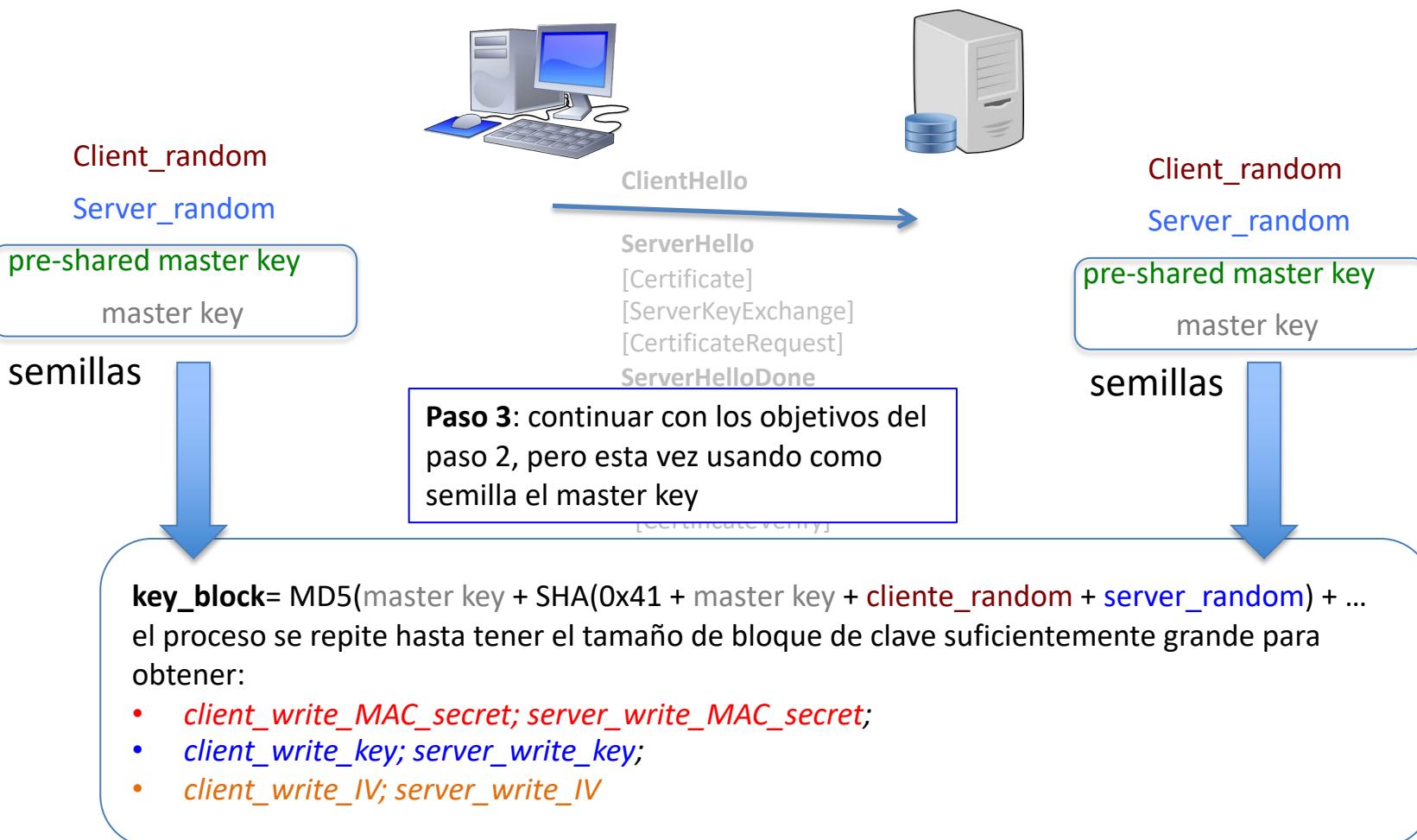
SSL - Secure Sockets Layer - intercambio de claves



SSL - Secure Sockets Layer - intercambio de claves



SSL - Secure Sockets Layer - intercambio de claves



SSL - Secure Sockets Layer - intercambio de claves

- Tanto SSL como TLS usan también DHE (Diffie Helman efímero):
 - Sin embargo, antes de entrar en el DHE, recordemos cómo funciona DH

- Alice:

- Valores públicos: q, α
- $X_a < q$, clave privada
- $Y_a = \alpha^{X_a} \text{ mod } q$, clave pública

- $Y_b = \alpha^{X_b} \text{ mod } q$
- $K_{AB} = (Y_b)^{X_a} \text{ mod } q$
- $K_{AB} = (\alpha^{X_b})^{X_a} \text{ mod } q$

- Bob:

- Valores públicos: q, α
- $X_b < q$, clave privada
- $Y_b = \alpha^{X_b} \text{ mod } q$, clave pública

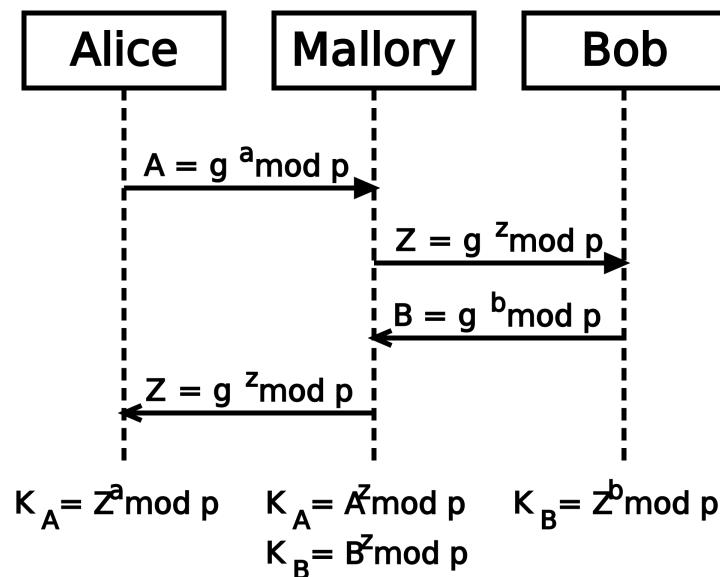
- $Y_a = \alpha^{X_a} \text{ mod } q$
- $K_{AB} = (Y_a)^{X_b} \text{ mod } q$
- $K_{AB} = (\alpha^{X_a})^{X_b} \text{ mod } q$

Recordatorio...

$$K_{AB} = \alpha^{X_b X_a} \text{ mod } q = \alpha^{X_a X_b} \text{ mod } q$$

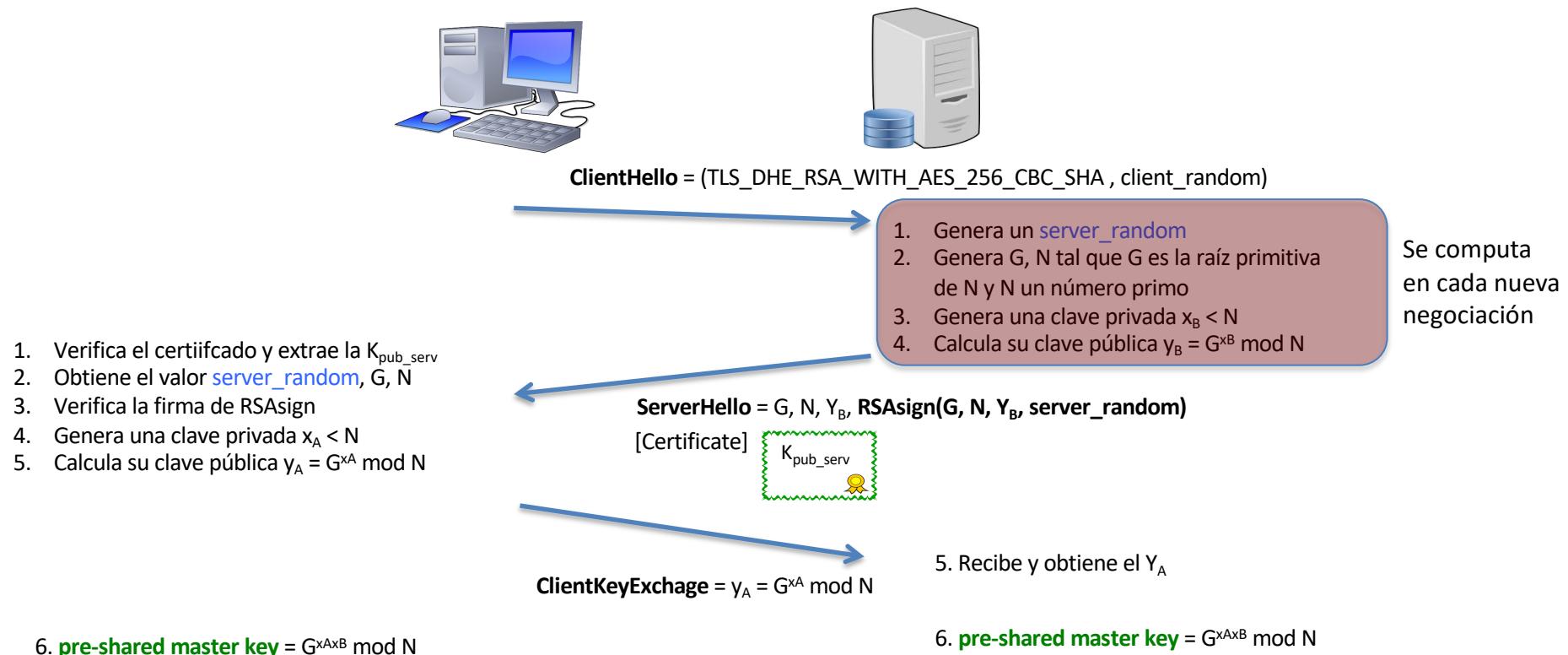
SSL - Secure Sockets Layer - intercambio de claves

- Tanto SSL como TLS usan también DHE (**Diffie Helman efímero**):
 - Tanto el cliente como el servidor generan sus valores secretos en **cada negociación** en lugar de usar valores estáticos para el intercambio
 - P. ej. usar sólo la clave privada en RSA y de manera permanente
 - Esta negociación constante se consigue el **Perfect Forward Secrecy** (PFS) en la creación del secreto compartido
 - El PFS evita el riesgo de MiTM
 - Si la clave privada de RSA (del servidor) se filtra o hay un MiTM en las negociaciones de DH entre dos entidades, entonces el MiTM puede derivar las claves de sección que se establezcan entre ambas entidades legítimas



SSL - Secure Sockets Layer - intercambio de claves

- DHE (Diffie Helman Efímero) con RSA



PROTOCOLOS DE SESIÓN

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:

SSL record

1 byte 2 bytes 2 bytes ≥ 0 bytes



Tipo: 22 → SSL Handshake Protocol

CONTENIDO DEL HANDSHAKE

1 byte	3 bytes	≥ 0 bytes
Tipo	Longitud	Contenido

- Por lo tanto, cada mensaje tiene 3 campos:
 - *Tipo* (1 byte): indica uno de 10 posibles mensajes (ver siguiente tabla)
 - *Longitud* (3 bytes): longitud del mensaje en bytes
 - *Contenido* (≥ 0 bytes): parámetros asociados con el mensaje (ver también siguiente tabla)

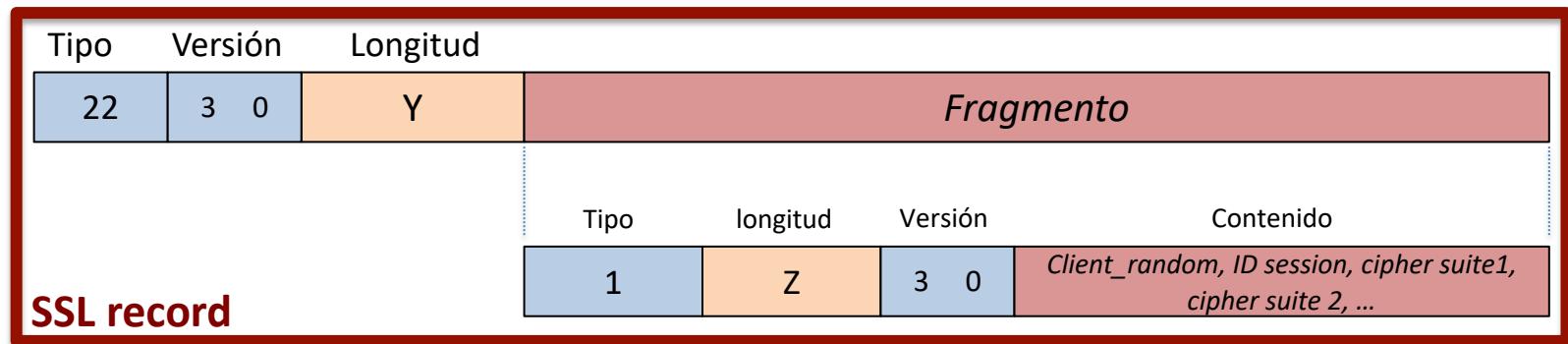
SSL Handshake Protocol

Mensajes	Parámetros asociados con los contenidos (contenido)	Tipo
Hello_request	null	Tipo = 0 Lo solicita el servidor para renegociar una sesión
Client_hello	Versión, random_cliente, sesión ID, cipher suite, método de compresión	Tipo = 1
Server_hello	Versión, random_servidor, sesión ID, cipher suite, método de compresión	Tipo = 2
Certificate	Cadena de certificados X.509	Tipo = 11
Server_key_exchange	Parámetros de seguridad necesarios para computar la clave de sesión (ej. usando DH como seed inicial) y firma del hash(cliente_random + servidor_random + parámetros de seguridad)	Tipo = 12
Certificate_request	Tipo de certificados y autoridades	Tipo = 13
Server_hello_done	Null	Tipo = 14
Client_key_exchange	El pre-shared master key cifrado con RSA, o añade los parámetros de DH/Fortezza para computar el master key en cada una de las partes	Tipo = 16
Certificate_verify	Se aplica cuando se solicita el certificado. Consiste en firmar el hash(master key + hash(todos los mensajes intercambiados hasta el momento))	Tipo = 15
Finished 	Es el cifrado del hash(master_key + hash(hash(todos los mensajes intercambiados hasta el momento) + ID_sender))	Tipo = 20

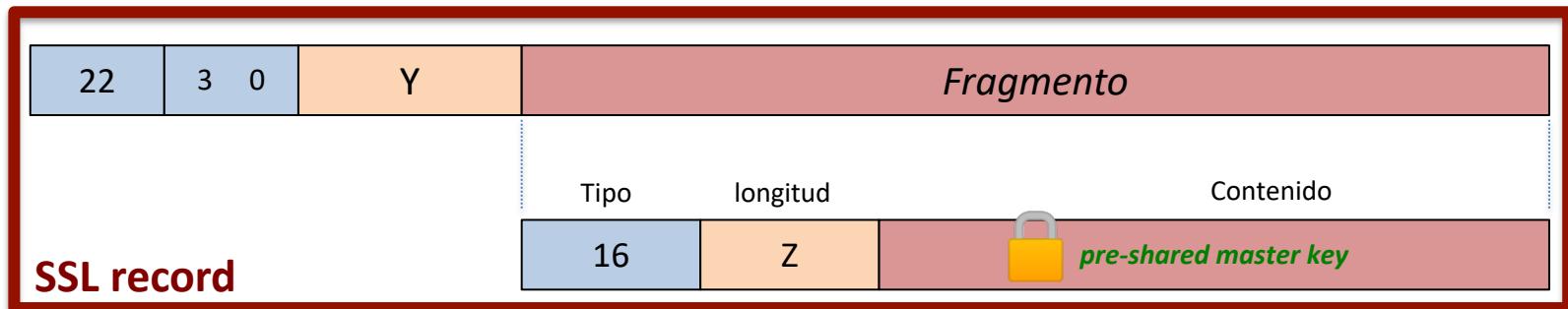
SSL Handshake Protocol



- El formato de algunos mensajes serían:
 - ClientHello



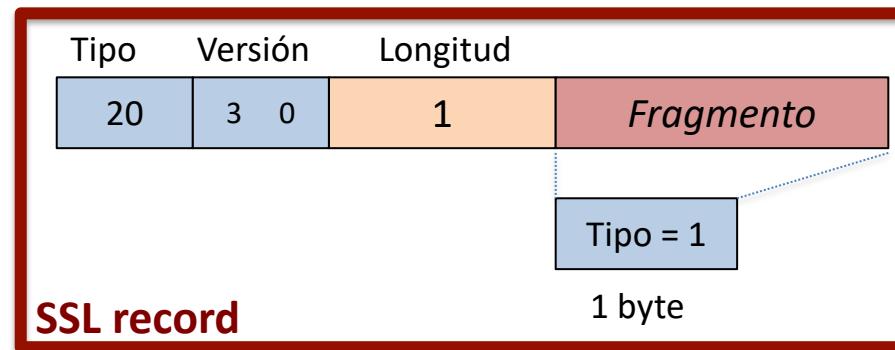
- ClientKeyExchange (RSA)



SSL Change Cipher Spec Protocol



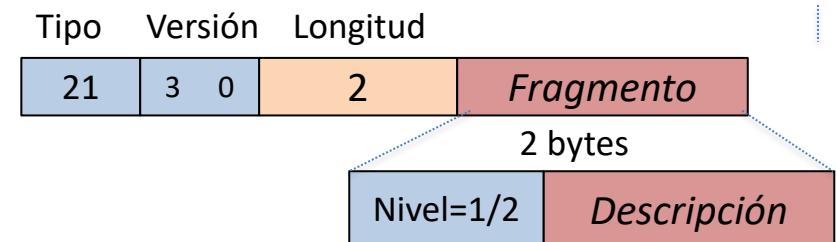
- Es un protocolo muy simple que consta de un solo mensaje de un sólo byte con valor 1 que permite activar el *cipher suite*



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 (opcional) y se cifran de acuerdo con lo establecido en la sesión

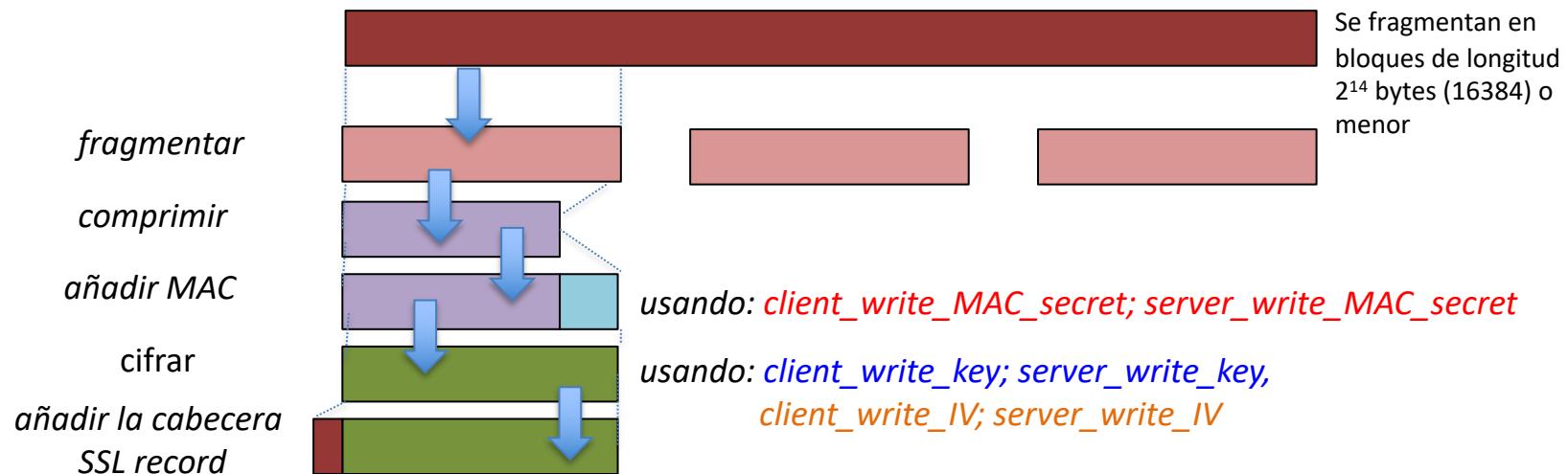


- El primer byte toma el valor 1 (warning) o 2 (fatal) para informar de la severidad del mensaje
 - Si el nivel es fatal, SSL termina la conexión de forma inmediata
 - Otras conexiones de la misma sesión pueden continuar pero no se producen nuevas conexiones dentro de la misma sesión
- El segundo byte contiene un código que indica la alerta específica
 - Ejemplos: *unexpected_message*, *bad_record_mac*, *decompression_failure*, *illegal_parameter*, ...

SSL Record Protocol



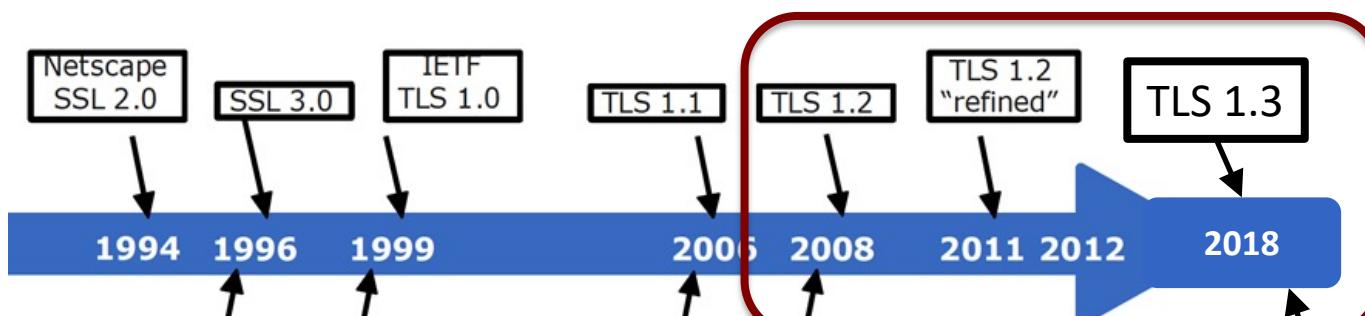
- Toma los datos de la subcapa alta:
 - *los fragmenta en bloques manejables,*
 - *los comprime de forma opcional,*
 - *añade el MAC,*
 - *cifra el paquete, y*
 - *añade una cabecera SSL record*
- El resultado final se trasmite en un segmento TCP



- En el destino, los datos recibidos son descifrados, verificados, descomprimidos y reensamblados antes de entregarlos a la capa de aplicación

TLSV1.2 Y TLSV1.3

Transport Layer Security



Nos centraremos en las dos últimas versiones de TLS

MAC Message Authentication Code
IETF Internet Engineering Task Force
CBC Cipher Block Chaining
IV Initialization Vector

MD5 Message Digest Algorithm
SHA Secure Hash Algorithm
AES Advanced Encryption Standard
CCM Counter with CBC-MAC

Major changes

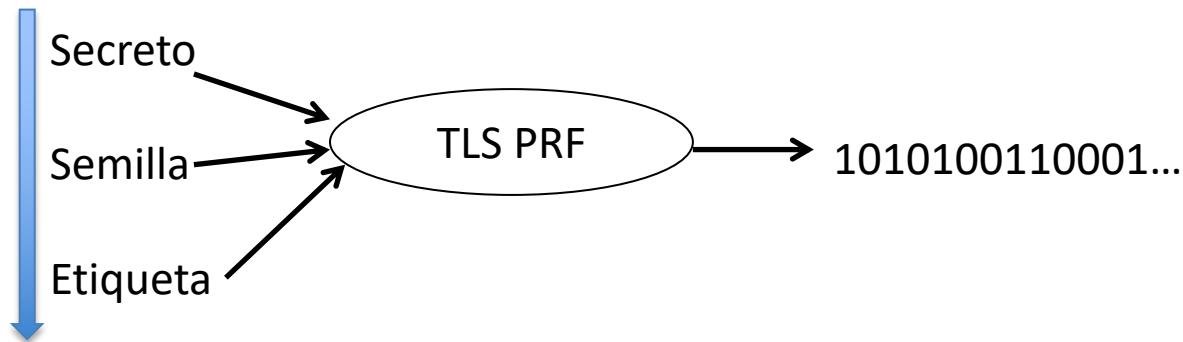
Operation modes are reduced, only AEAD (GCM and AES-CBC-MAC / AES-CCM) are supported, only PFS is applied to generate “pre-shared master key” does not use compression, shorter handshake, 0-RTT

“Authenticated Encryption with Addition Data” (AEAD)

TLSv1.2



- TLS 1.2 introduce cambios muy significativos:
 - Transacciones del handshake:
 - el master-key se computa con SHA-256
 - en vez de usar un generador PRF con MD5 y SHA-1 como lo usa TLSv1.0 y TLSv1.1
 - Concretamente, TLSv1.2 y TLSv1.3 calculan las claves de la siguiente forma:
master key (48 bytes) = PRF-SHA-256(pre-shared master key, “master_key”, client_random + server_random) sabiendo que



key block = PRF(master key, “key_expansion”, client_random + server_random) y de esta bloque de clave se deriva:
 $\text{client_write_MAC_secret}; \text{server_write_MAC_secret}; \text{client_write_key}; \text{server_write_key}; \text{client_write_IV}; \text{server_write_IV}$

- permite incluir una lista de “extensiones” en los mensajes ClientHello/ServerHello
 \rightarrow + detalles sobre los certificados, parámetros de seguridad, autorizaciones, tipos de certificados, etc.

TLSv1.2

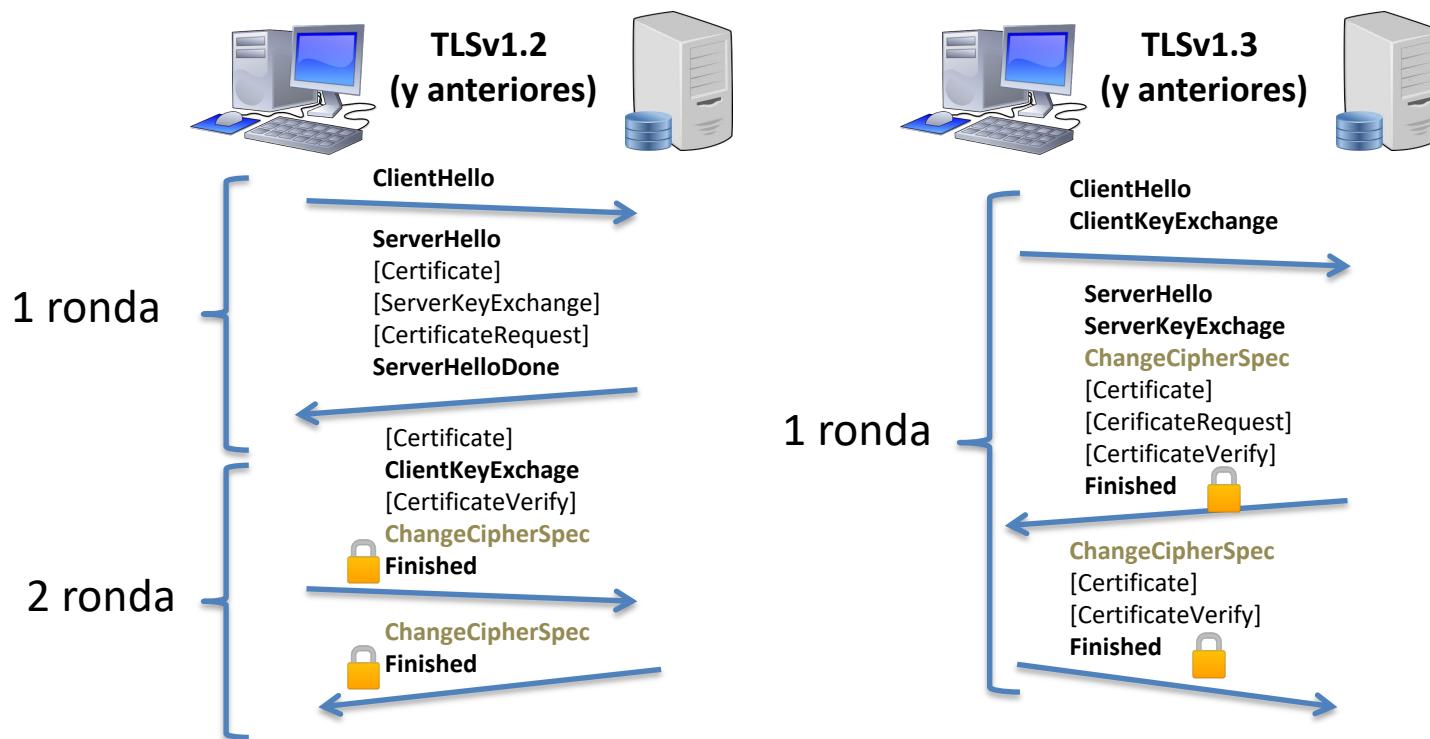


- Cipher suite:
 - Quita DES e IDEA, y añade AES
 - Añade criptografía de clave pública basada en curvas elípticas con ECDHE para la negociación de claves
 - Introduce el concepto de “Authenticated Encryption with Addition Data” (AEAD)
 - AES-CBC-MAC / AES-CCM
 - AES-GCM (Galois-Counter Mode)
 - » Funciona de forma similar que el modo CRT pero usa Carter-Wegman **MAC** en un campo de Galois
 - » Es rápido y eficiente

TLSv1.3

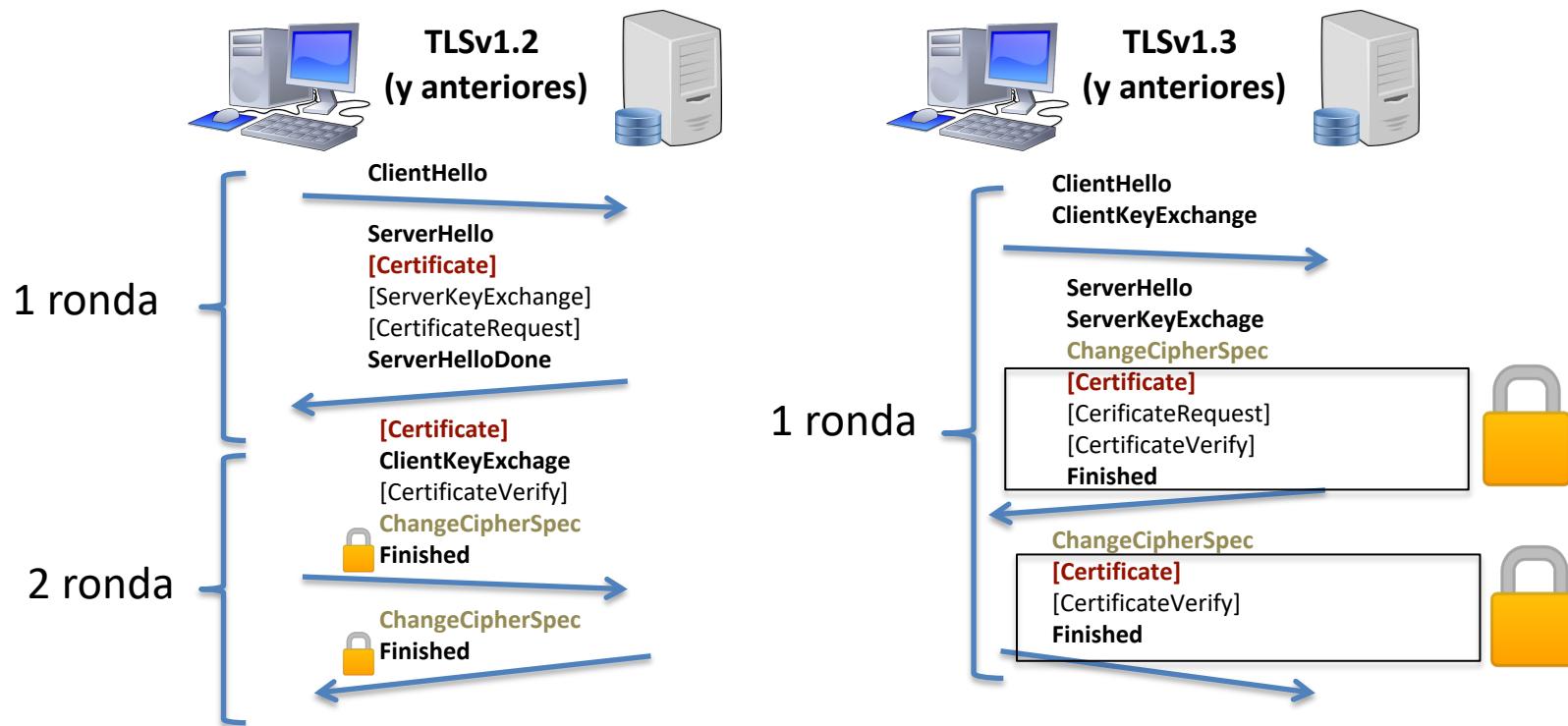


- TLS 1.3 introduce cambios muy significativos en términos de rendimiento y seguridad:
 - Un único Round-Trip Time (RTT)



TLSv1.3

- Además, TLSv1.3 es más seguro:



TLSv1.3



- TLS 1.3 introduce cambios muy significativos en términos de rendimiento y seguridad:
 - Un único Round-Trip Time (RTT)
 - 0-RTT para reconexión por usar las credenciales de seguridad de la sesión anterior – PSK (pre-shared master key)
 - Prevalece sesiones del tipo Perfect Forward Secrecy – DHE-RSA / ECDHE
 - Reduce el número de modos de operación soportados, limitándolo a CBC y AEAD: GCM y CCM

TLS_RSA_WITH_CAMELLIA_128_CBC_SHA256
TLS_DH_DSS_WITH_CAMELLIA_128_CBC_SHA256
TLS_DH_RSA_WITH_CAMELLIA_128_CBC_SHA256
TLS_DHE_DSS_WITH_CAMELLIA_128_CBC_SHA256
TLS_DHE_RSA_WITH_CAMELLIA_128_CBC_SHA256
TLS_DH_anon_WITH_CAMELLIA_128_CBC_SHA256
TLS_RSA_WITH_CAMELLIA_256_CBC_SHA256
TLS_DH_DSS_WITH_CAMELLIA_256_CBC_SHA256
TLS_DH_RSA_WITH_CAMELLIA_256_CBC_SHA256
TLS_DHE_DSS_WITH_CAMELLIA_256_CBC_SHA256
TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA256
TLS_DH_anon_WITH_CAMELLIA_256_CBC_SHA256
TLS_EMPTY_RENEGOTIATION_INFO_SCSV
TLS_FALLBACK_SCSV
TLS_ECDH_ECDSA_WITH_NULL_SHA
TLS_ECDH_ECDSA_WITH_RC4_128_SHA
TLS_ECDH_ECDSA_WITH_3DES_EDE_CBC_SHA
TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA

TLS_ECDH_ECDSA_WITH_AES_256_CBC_SHA
TLS_ECDHE_ECDSA_WITH_NULL_SHA
TLS_ECDHE_ECDSA_WITH_RC4_128_SHA
TLS_ECDHE_ECDSA_WITH_3DES_EDE_CBC_SHA
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA
TLS_ECDHE_RSA_WITH_NULL_SHA
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA
TLS_ECDHE_RSA_WITH_RC4_128_SHA
TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA

TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA256
TLS_ECDH_ECDSA_WITH_AES_256_CBC_SHA384
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384
TLS_ECDH_RSA_WITH_AES_128_CBC_SHA256
TLS_ECDH_RSA_WITH_AES_256_CBC_SHA384
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
TLS_ECDH_ECDSA_WITH_AES_128_GCM_SHA256
TLS_ECDH_ECDSA_WITH_AES_256_GCM_SHA384
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
...
...

DTLS

DTLS (Datagram Transport Layer Security)



- 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 escenarios que se requiera comunicación en tiempo real o restringidos (IoT)

 Datatracker Groups Documents Meetings Other User Document search

DTLS In Constrained Environments (dice) Concluded WG

[About](#) [Documents](#) [Meetings](#) [History](#) [Photos](#) [Email expansions](#) [List archive](#) [Tools »](#)

Document	Date	Status	IPR	AD / Shepherd
RFC (1 hit)				
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

Atom feed: [All changes](#) [Significant](#) [Subscribe to changes](#) [!\[\]\(9e495a407410ff9660147aa2d767824e_img.jpg\) Export as CSV](#)

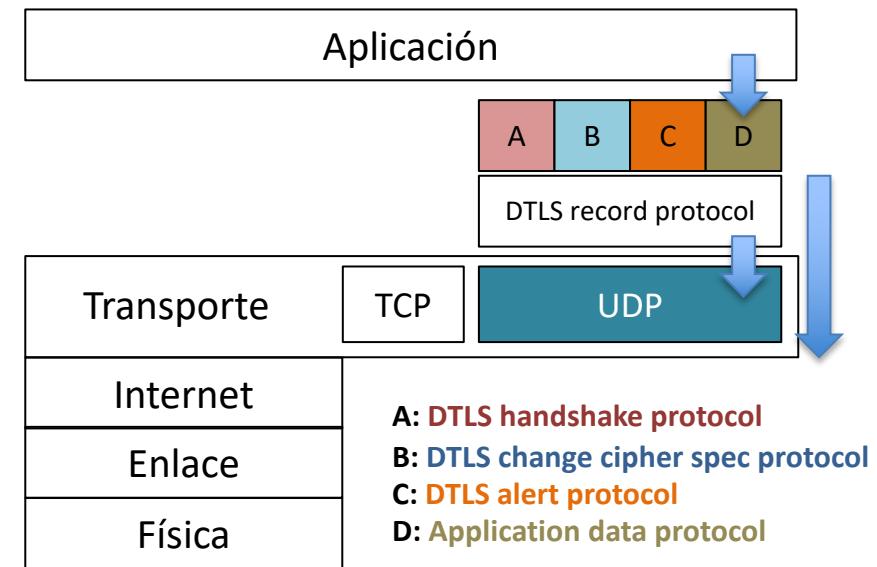
[ISOC](#) [IETF Trust](#) [RFC Editor](#) [IRTF](#) [IESG](#) [IETF](#) [IAB](#) [IASA & IAOC](#) [IETF Tools](#) [IANA](#)

About | IETF Datatracker | Version 6.89.2 | 2018-12-19 | Report a bug: Tracker | Email | Python 2.7.15 | Django 1.11.17

<https://datatracker.ietf.org/wg/dice/documents/>

DTLS (Datagram Transport Layer Security)

- Como se puede ver la estructura es similar al del TLS, pero con la diferencia que la comunicación va sobre UDP
 - esto significa que los datagramas son transmitidos de forma no fiable (puede haber pérdidas o ensamblado no ordenado)
- Para evitar el punto anterior:
 - incluir mecanismos para controlar el flujo de la información
 - un explícito número de secuencia en cada **DTLS record**



Transport Layer Security



Transport Layer Security (tls)

[About](#) [Documents](#) [Meetings](#) [History](#) [Photos](#) [Email expansions](#) [List archive »](#) [Tools »](#)

WG Name Transport Layer Security

Acronym tls

Area Security Area (sec)

State Active

Charter [charter-ietf-tls-06](#) Approved

Status Update [Show update](#) (last changed 2018-11-07)

Dependencies [Document dependency graph \(SVG\)](#)

Additional - [Github](#)

Resources - [Home Page](#)

- [Wiki](#)

Personnel Chairs Christopher Wood

Joseph Salowey

Sean Turner

Area Director Benjamin Kaduk

Mailing list Address tls@ietf.org

To subscribe <https://www.ietf.org/mailman/listinfo/tls>

Archive <https://mailarchive.ietf.org/arch/browse/tls/>

Jabber chat Room address <xmpp:tls@jabber.ietf.org?join>

Logs <https://jabber.ietf.org/logs/tls/>

Charter for Working Group

The TLS (Transport Layer Security) working group was established in 1996 to standardize a 'transport layer' security protocol. The basis for the work was SSL (Secure Socket Layer) v3.0 [RFC6101]. The TLS working group has completed a series of specifications that describe the TLS protocol v1.0 [RFC2246], v1.1 [RFC4546], v1.2 [RFC5346], and v1.3 [RFC8446], and DTLS (Datagram TLS) v1.0 [RFC4347], v1.2 [RFC6347], and v1.3 [draft-ietf-tls-dtls13], as well as extensions to the protocols and ciphersuites.

The working group aims to achieve three goals. First, improve the applicability and suitability of the TLS family of protocols for use in emerging protocols and use cases. This includes extensions or changes that help protocols better use TLS as an authenticated key exchange protocol, or extensions that help protocols better leverage TLS security properties, such as Exported Authenticators. Extensions that focus specifically on protocol extensibility are also in scope. This goal also includes protocol changes that reduce TLS resource consumption without affecting security. Extensions that help reduce TLS handshake size meet this criterion.

The second working group goal is to improve security, privacy, and deployability. This includes, for example, Delegated Credentials and Encrypted SNI. Security and privacy goals will place emphasis on the following:

- Encrypt the ClientHello SNI (Server Name Indication) and other application-sensitive extensions, such as ALPN (Application-Layer Protocol Negotiation).
- Identify and mitigate other (long-term) user tracking or fingerprinting vectors enabled by TLS deployments and implementations.

The third goal is to maintain current and previous version of the (D)TLS protocol as well as to specify general best practices for use of (D)TLS, extensions to (D)TLS, and cipher suites. This includes recommendations as to when a particular version should be deprecated. Changes or additions to older versions of (D)TLS whether via extensions or ciphersuites are discouraged and require significant justification to be taken on as work items.

The working group will also place a priority in minimizing gratuitous changes to (D)TLS.

Transport Layer Security



ietf.org Datatracker Groups Documents Meetings Other User

Transport Layer Security (tls)

About Documents Meetings History Photos Email expansions List archive » Tools »

Document		Date	Status
Active Internet-Drafts (14 hits)			
draft-ietf-tls-ctls-01	2020-11-02	I-D Exists	
Compact TLS 1.3	19 pages	WG Document	
draft-ietf-tls-dtls-connection-id-08	2020-11-02	AD Evaluation::AD Followup for 82 days	
Connection Identifiers for DTLS 1.2	18 pages	Submitted to IESG for Publication: Proposed Standard	
draft-ietf-tls-dtls13-39	2020-11-02	AD Evaluation::Revised I-D Needed for 23 days	
The Datagram Transport Layer Security (DTLS) Protocol Version 1.3	58 pages	Submitted to IESG for Publication: Proposed Standard	
draft-ietf-tls-esni-08	2020-10-16	I-D Exists	
TLS Encrypted Client Hello	37 pages	WG Document	
draft-ietf-tls-exported-authenticator-13	2020-06-26	Waiting for Writeup::Revised I-D Needed for 46 days	
Exported Authenticators in TLS	14 pages	Submitted to IESG for Publication: Proposed Standard	
draft-ietf-tls-external-psk-guidance-01	2020-11-02	I-D Exists	
Guidance for External PSK Usage in TLS	14 pages	WG Document	
draft-ietf-tls-external-psk-importer-06	2020-12-03	Waiting for Writeup::AD Followup for 46 days	
Importing External PSKs for TLS	11 pages New	Submitted to IESG for Publication: Proposed Standard	
draft-ietf-tls-hybrid-design-01	2020-10-15	I-D Exists	
Hybrid key exchange in TLS 1.3	34 pages	WG Document	
draft-ietf-tls-md5-sha1-deprecate-04	2020-10-09	Waiting for Writeup::AD Followup for 21 days	
Deprecating MD5 and SHA-1 signature hashes in TLS 1.2	6 pages	Submitted to IESG for Publication: Proposed Standard	
draft-ietf-tls-oldversions-deprecate-09	2020-11-09	I-D Exists	
Deprecating TLSv1.0 and TLSv1.1	23 pages	Waiting for Writeup for 6 days	
draft-ietf-tls-rfc8446bis-00	2020-10-05	Submitted to IESG for Publication: Best Current Practice	
The Transport Layer Security (TLS) Protocol Version 1.3	152 pages	Reviews: genart, oopsdir, secdir	
draft-ietf-tls-subcerts-09	2020-06-26	Waiting for WG Chair Go-Ahead: Proposed Standard	
Delegated Credentials for TLS	18 pages	Sep 2020	
draft-ietf-tls-ticketrequests-07	2020-12-03	IESG Evaluation for 1 day	
TLS Ticket Requests	8 pages New	IESG telechat: 2020-12-17	
draft-ietf-tls-tlsflags-03	2020-07-03	Submitted to IESG for Publication: Proposed Standard	
A Flags Extension for TLS 1.3	8 pages	Reviews: genart, oopsdir, secdir	
		Nov 2020	
		Nov 2020	

Transport Layer Security (tls)

About Documents Meetings History Photos Email expansions List archive » Tools »

WG

Name	Transport Layer Security
Acronym	tls
Area	Security Area (sec)
State	Active
Charter	charter-ietf-tls-06 Approved
Status Update	Show update (last changed 2018-11-07)
Dependencies	Document dependency graph (SVG)
Additional Resources	- Github - Home Page - Wiki

Personnel

Chairs	Christopher Wood Joseph Salowey Sean Turner
Area Director	Benjamin Kaduk

Mailing list

Address	tls@ietf.org
To subscribe	https://www.ietf.org/mailman/listinfo/tls
Archive	https://mailarchive.ietf.org/arch/browse/tls

Jabber chat

Room address	xmpp:tls@jabber.ietf.org?join
Logs	https://jabber.ietf.org/logs/tls/

Charter for Working Group

The TLS (Transport Layer Security) working group was established in 1996 to standardize the transport layer security protocol. The protocol has evolved from version 1.0 [RFC2246], v1.1 [RFC4346], v1.2 [RFC5546], and v1.3 [RFC8446]. The working group aims to achieve three goals. First, improve the applicability of extensions that help protocols better leverage TLS security properties, such as security. Extensions that help reduce TLS handshake size meet this criterion. The second working group goal is to improve security, privacy, and deployability. - Encrypt the ClientHello SNI (Server Name Indication) and other application-layer protocols. - Identify and mitigate other (long-term) user tracking or fingerprinting vectors. The third goal is to maintain current and previous versions of the (D)TLS protocol additions to older versions of (D)TLS whether via extensions or ciphersuites and other mechanisms. The working group will also place a priority in minimizing gratuitous changes.

SEGURIDAD EN LA CAPA DE INTERNET



Seguridad en la Capa de Internet

- En 1994, la *Internet Architecture Board (IAB)* publicó un informe titulado “*Security in the Internet Architecture*” (RFC1636)
 - En este informe se identificaba la necesidad de proporcionar **seguridad a la infraestructura de red**
 - En este informe se aconseja la incorporación de mecanismos de
 - cifrado y
 - autenticaciónpara la siguiente versión de IP (IPv6)

The screenshot shows the header and abstract of RFC1636. The header includes links for RFC Home, TEXT, PDF, HTML, Tracker, IPR, and Info page. It also indicates the document is informational. The abstract discusses the purpose of the workshop, its date (February 8-10, 1994), and the status of the memo (unlimited distribution).

[RFC Home] [TEXT] [PDF] [HTML] [Tracker] [IPR] [Info page]

INFORMATIONAL

Network Working Group
Request for Comments: 1636
Category: Informational

R. Braden
ISI
D. Clark
MIT Laboratory for Computer Science
S. Crocker
Trusted Information Systems, Inc.
C. Huitema
INRIA, IAB Chair
June 1994

Report of IAB Workshop on
Security in the Internet Architecture
February 8-10, 1994

Status of this Memo

This memo provides information for the Internet community. This memo does not specify an Internet standard of any kind. Distribution of this memo is unlimited.

Abstract

This document is a report on an Internet architecture workshop, initiated by the IAB and held at USC Information Sciences Institute on February 8-10, 1994. This workshop generally focused on security issues in the Internet architecture.

This document should be regarded as a set of working notes containing ideas about security that were developed by Internet experts in a broad spectrum of areas, including routing, mobility, realtime service, and provider requirements, as well as security. It contains some significant diversity of opinions on some important issues. This memo is offered as one input in the process of developing viable security mechanisms and procedures for the Internet.

<https://www.rfc-editor.org/rfc/rfc1636>

Seguridad en la Capa de Internet

- 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**

[RFC Home] [TEXT|PDF|HTML] [Tracker] [IPR] [Errata] [Info page]

PROPOSED STANDARD
Errata Exist
S. Kent
K. Seo
BBN Technologies
December 2005

Updated by: [6040](#), [7619](#)
Network Working Group
Request for Comments: 4301
Obsoletes: [2401](#)
Category: Standards Track

Security Architecture for the Internet Protocol

Status of This Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Copyright Notice

Copyright (C) The Internet Society (2005).

Abstract

This document describes an updated version of the "Security Architecture for IP", which is designed to provide security services for traffic at the IP layer. This document obsoletes [RFC 2401](#) (November 1998).

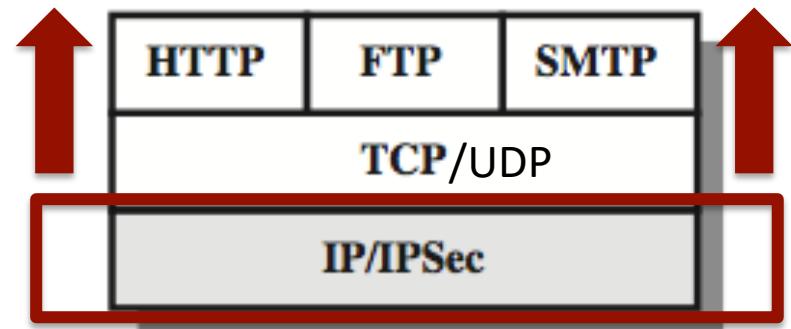
Dedication

This document is dedicated to the memory of Charlie Lynn, a long-time senior colleague at BBN, who made very significant contributions to the IPsec documents.

<https://www.rfc-editor.org/rfc/rfc4301>

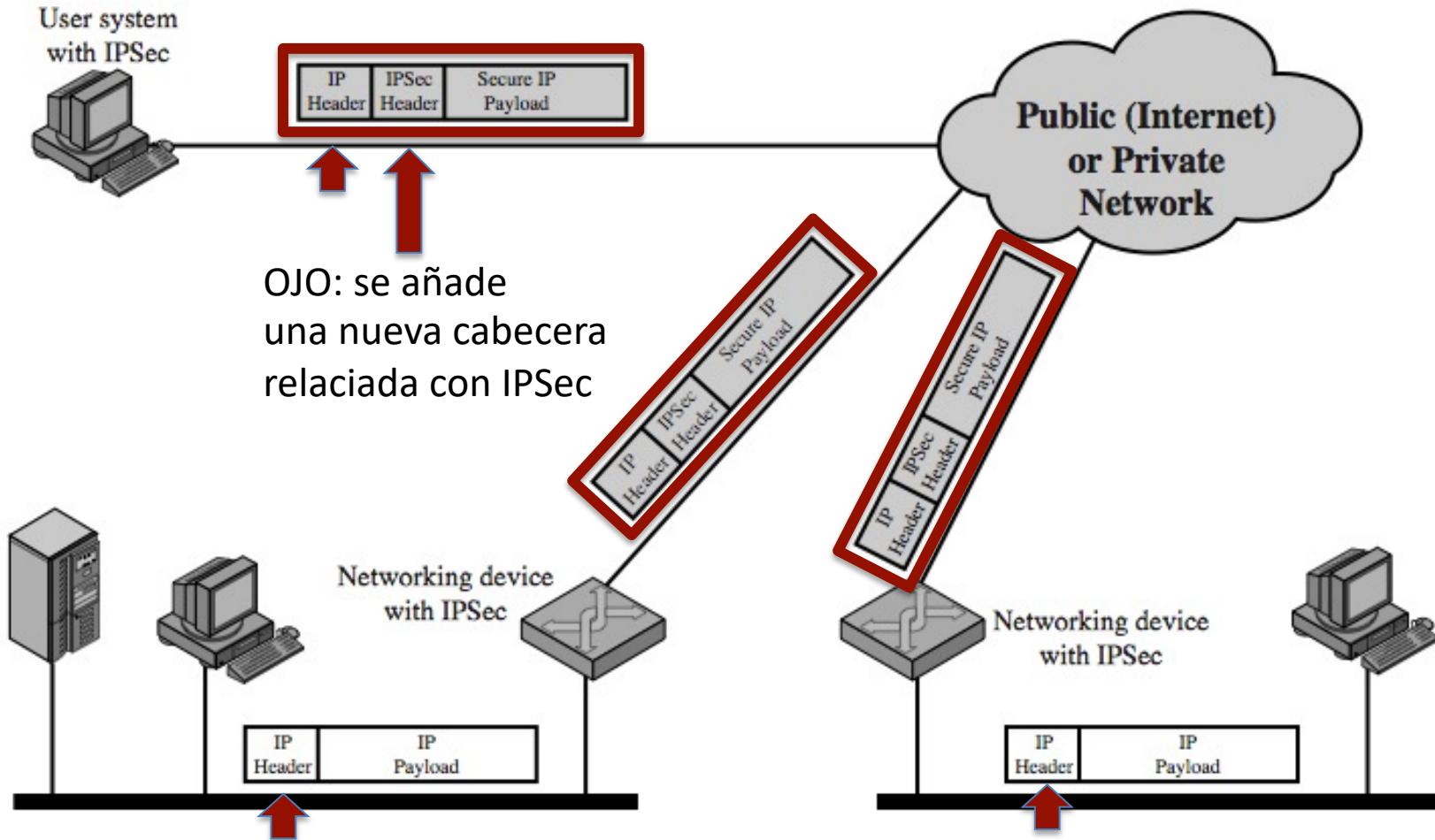
Seguridad en la Capa de Internet

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



- Su aplicación se puede extender a múltiples tipos de escenarios:
 - Conectividad segura entre sucursales a través de Internet
 - Acceso remoto seguro vía el Internet
 - Establecimiento de conectividad extranet e intranet con socios
 - Aplicaciones de comercio electrónico
 - etc.

Seguridad en la Capa de Internet



Seguridad en la Capa de Internet

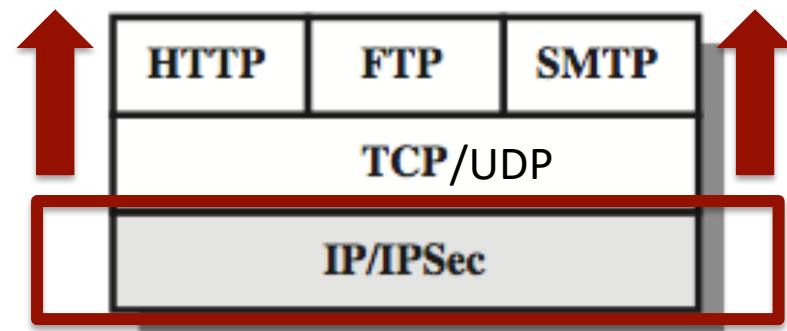
- Como en TLS, la seguridad en IPSec se centra en proporcionar:
 - Autenticación del origen de los mensajes
 - Integridad
 - Confidencialidad
 - Intercambio de claves entre los puntos que se comunican

Esto se consigue con:

- MAC
- cifrado y
- algoritmos para el intercambio de clave (ej. DH)
- IPSec **NO** proporciona:
 - servicios de no-repudio, como SSL/TLS, y
 - protección frente a ataques DoS, aunque sí proporciona una forma de protección ante ataques de repetición

Seguridad en la Capa de Internet

- Como se ha comentado anteriormente, 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 informar a los usuarios sobre el uso de mecanismos de seguridad
 - no hace falta que gestionen claves

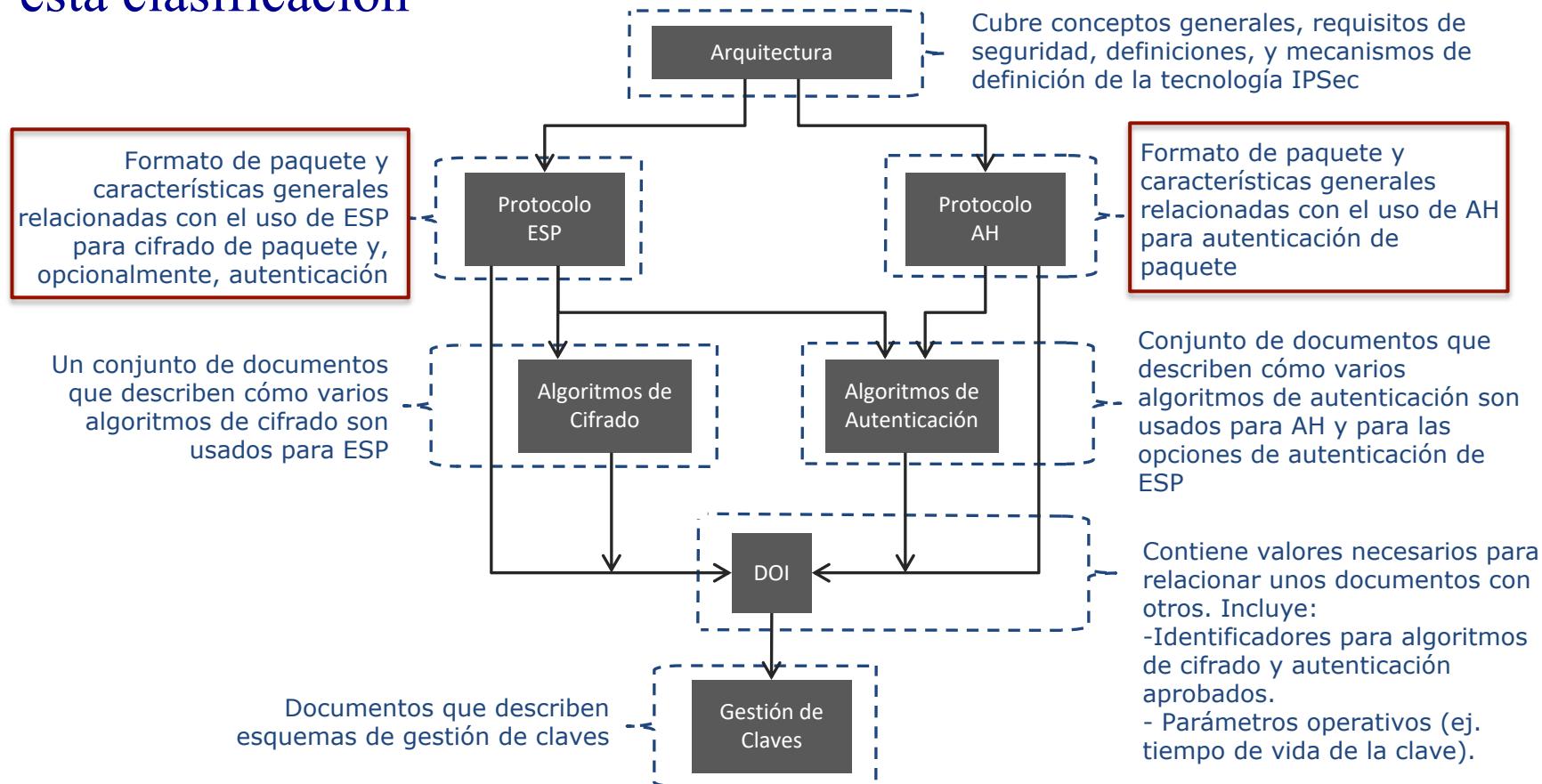


Seguridad en la Capa de Internet

- Para la comunicación segura entre dos puntos, IPSec utiliza los siguientes protocolos:
 - **ESP (Encapsulating Security Payload)**
 - Garantiza **confidencialidad, integridad y autenticación (opcional)**
 - Principalmente proporciona protección frente a lecturas ilícitas de tráfico en red
 - También incluye un número de secuencia que proporciona una forma de **protección ante ataques de repetición**
 - **AH (Authentication Header)**
 - Garantiza **integridad y autenticación** del origen de datos
 - También incluye un número de secuencia que proporciona una forma de **protección ante ataques de repetición** (lo mismo que ESP)
 - **IKE (Internet Key Exchange)**
 - Protocolo específico para **generar y distribuir claves** para ESP y AH
 - También autentica la identidad del sistema remoto
- Cuando usando estos tres protocolos:
 - Antes de que dos puntos se comuniquen de forma segura, tienen que acordar qué **parámetros de seguridad** se van a aplicar

Seguridad en la Capa de Internet

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



DOI: Domain of Interpretation

Seguridad en la Capa de Internet

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

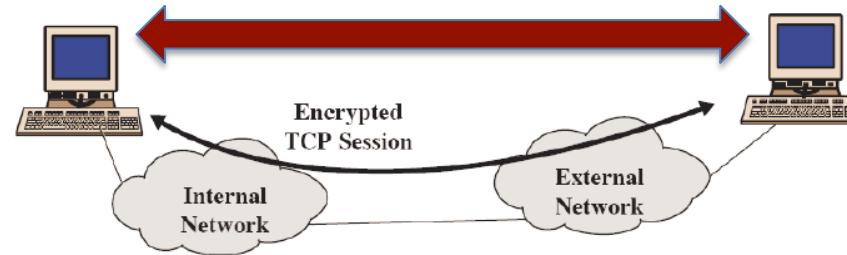
Some IP protocol codes	
Protocol code	Protocol Description
1	ICMP — Internet Control Message Protocol
2	IGMP — Internet Group Management Protocol
4	IP within IP (a kind of encapsulation)
6	TCP — Transmission Control Protocol
17	UDP — User Datagram Protocol
41	IPv6 — next-generation TCP/IP
47	GRE — Generic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



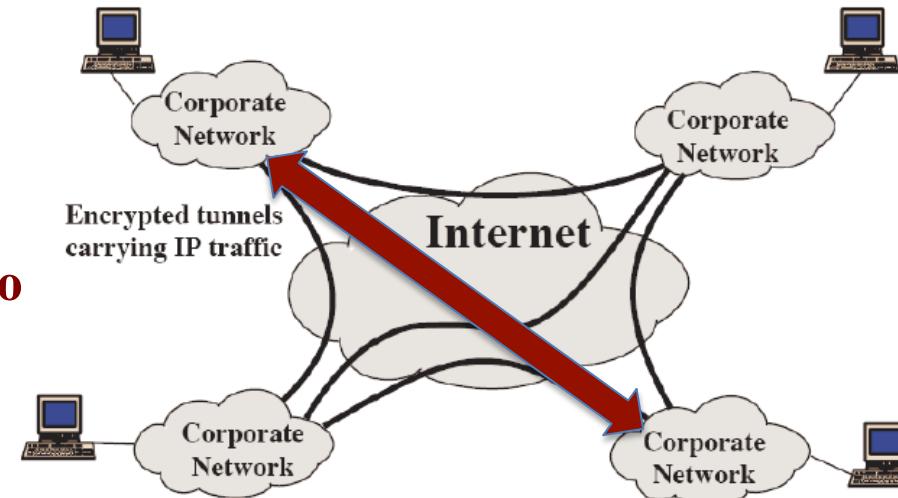
- Lista completa en:
<http://www.iana.org/assignments/protocol-numbers/protocol-numbers.xhtml>

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, ...

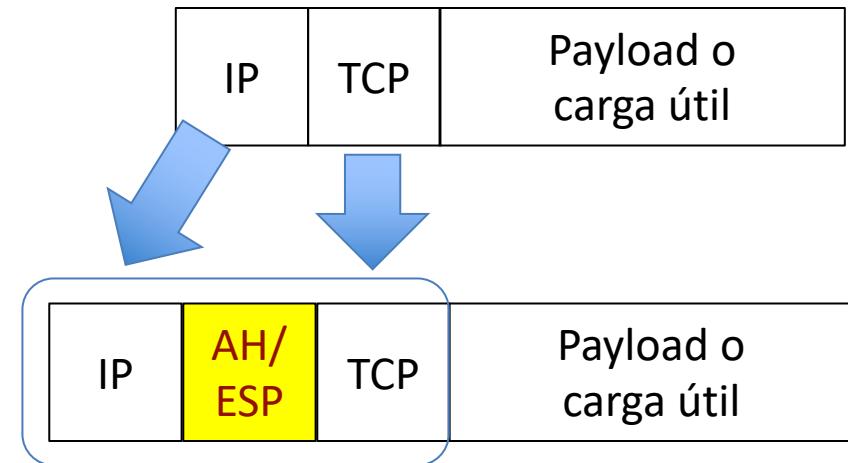


- Modo túnel:
 - Se usa cuando los puntos a comunicar son **gateways** de seguridad o **routers**
 - Proporciona **protección a todo el paquete IP**



Modos de IPSec

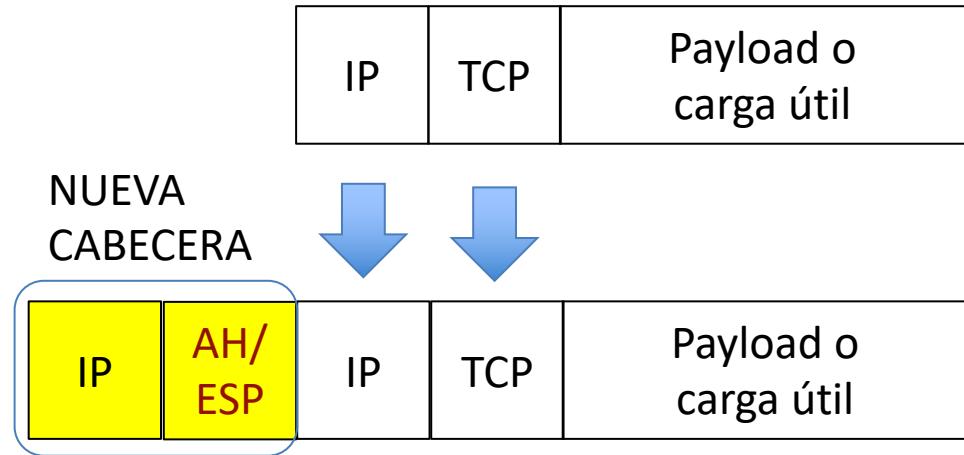
- **El modo transporte**
IPsec sólo protege la carga del datagrama IP
 - insertando la cabecera IPsec (AH, ESP) entre la cabecera IP y la cabecera del protocolo de capas superiores



- *IP origen – host*
- *IP destino - host*
- Cifrado
- Autenticación
- Integridad

Modos de IPSec

- El modo túnel encapsula TODO el datagrama IP dentro de un nuevo datagrama IP
 - donde añade la IP de los routers/gateways y la información del protocolo IPsec (AH, ESP)
- Por tanto, la protección es en todo el paquete IP:
 - añadiendo las cabecera AH o ESP al paquete IP original, y
 - 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**

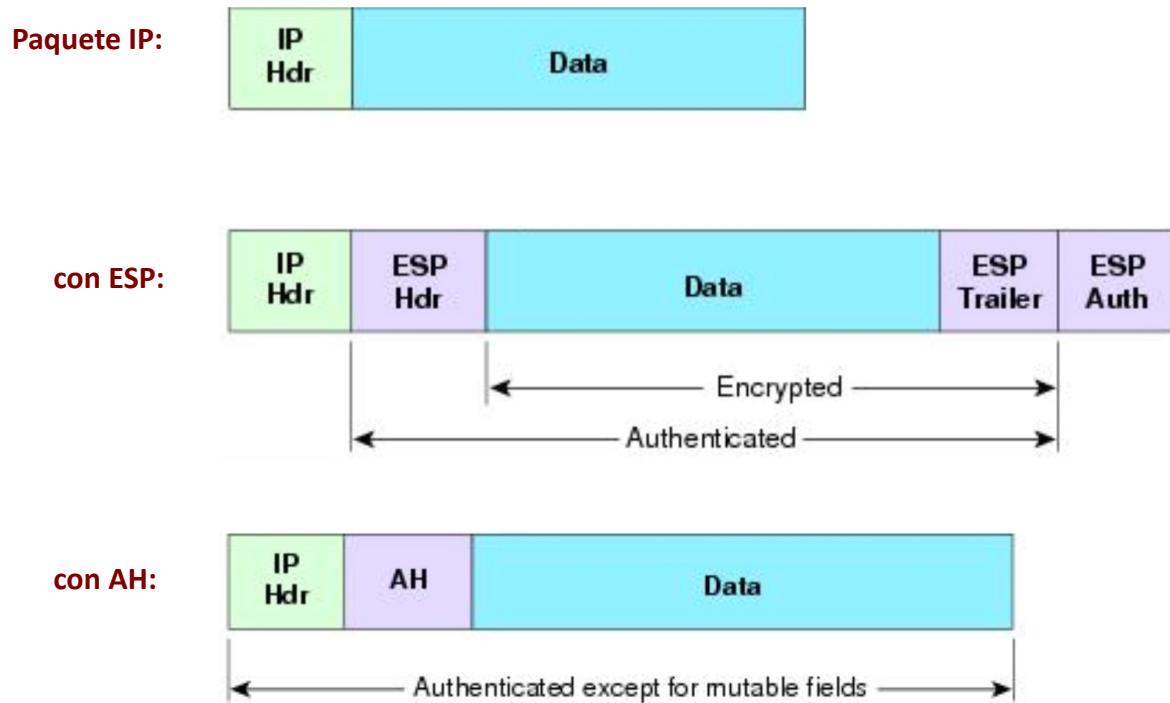


- *IP origen – router*
- *IP destino - router*
- Cifrado
- Autenticación
- Integridad

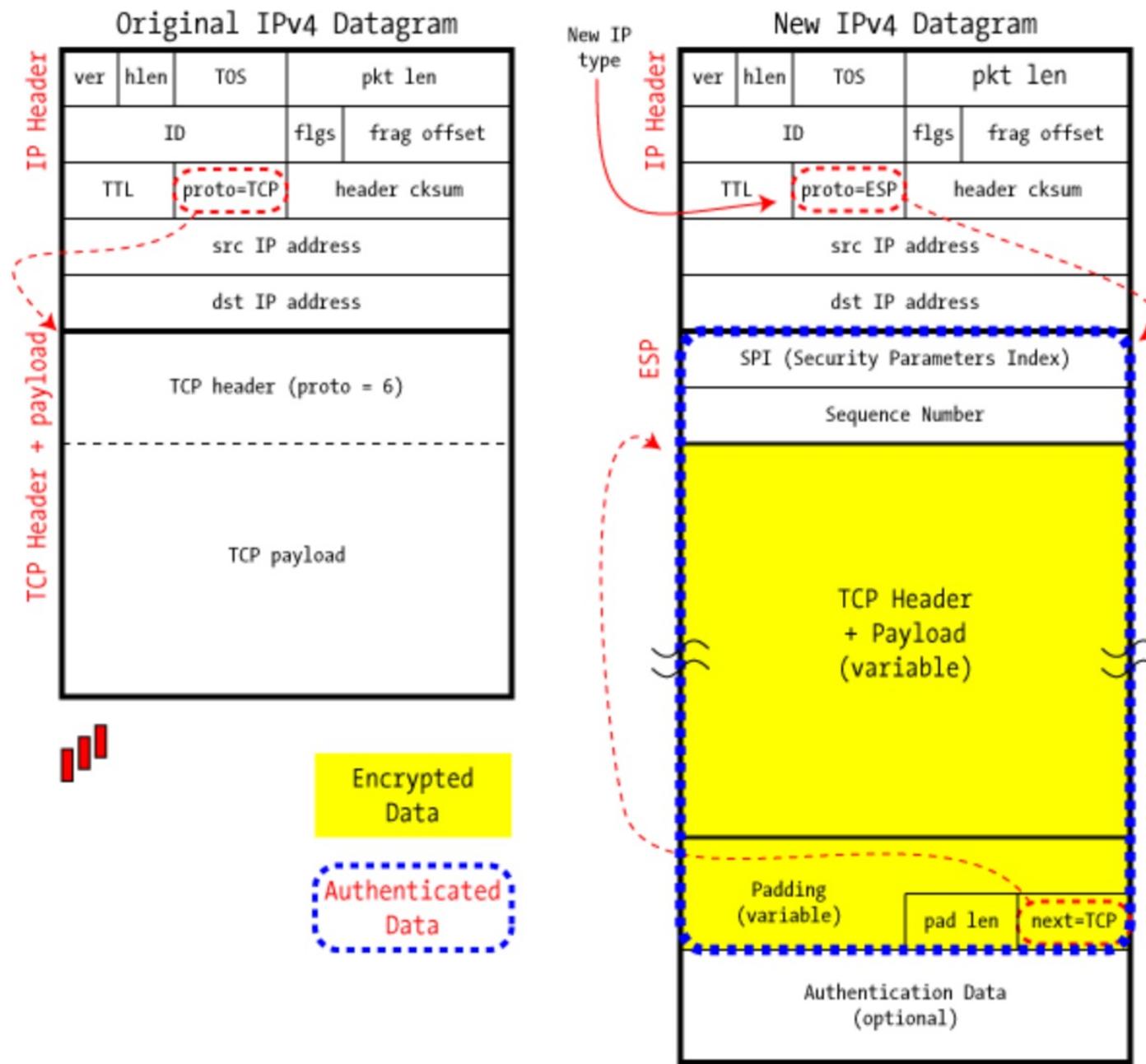
Modo transporte

- En este modo de uso:

- **si se utiliza ESP:** se cifra el *payload* y opcionalmente lo autentica, pero no la cabecera IP
- **si se utiliza AH:** se autentica el *payload* y algunas porciones de la cabecera IP

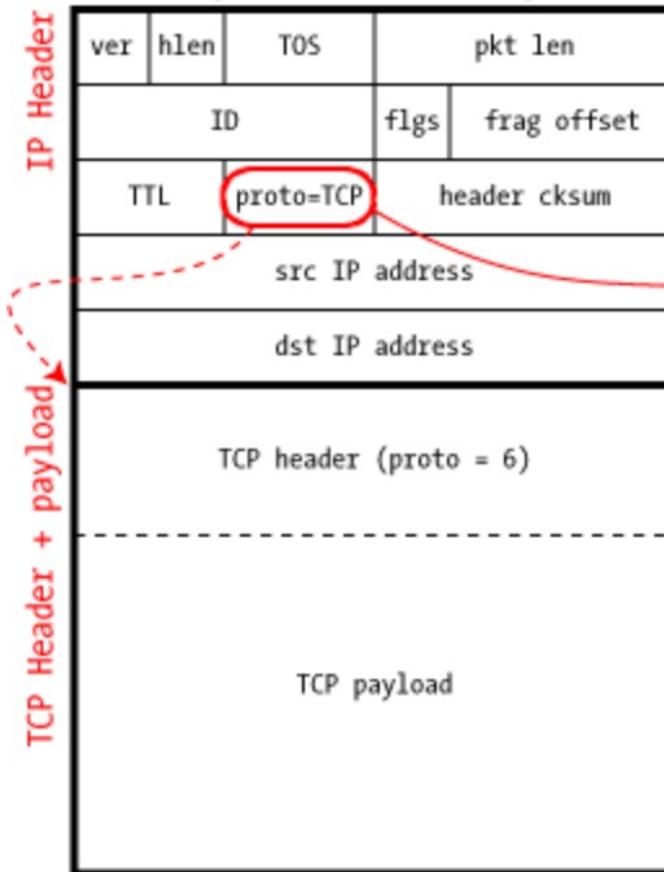


IPSec in ESP Transport Mode

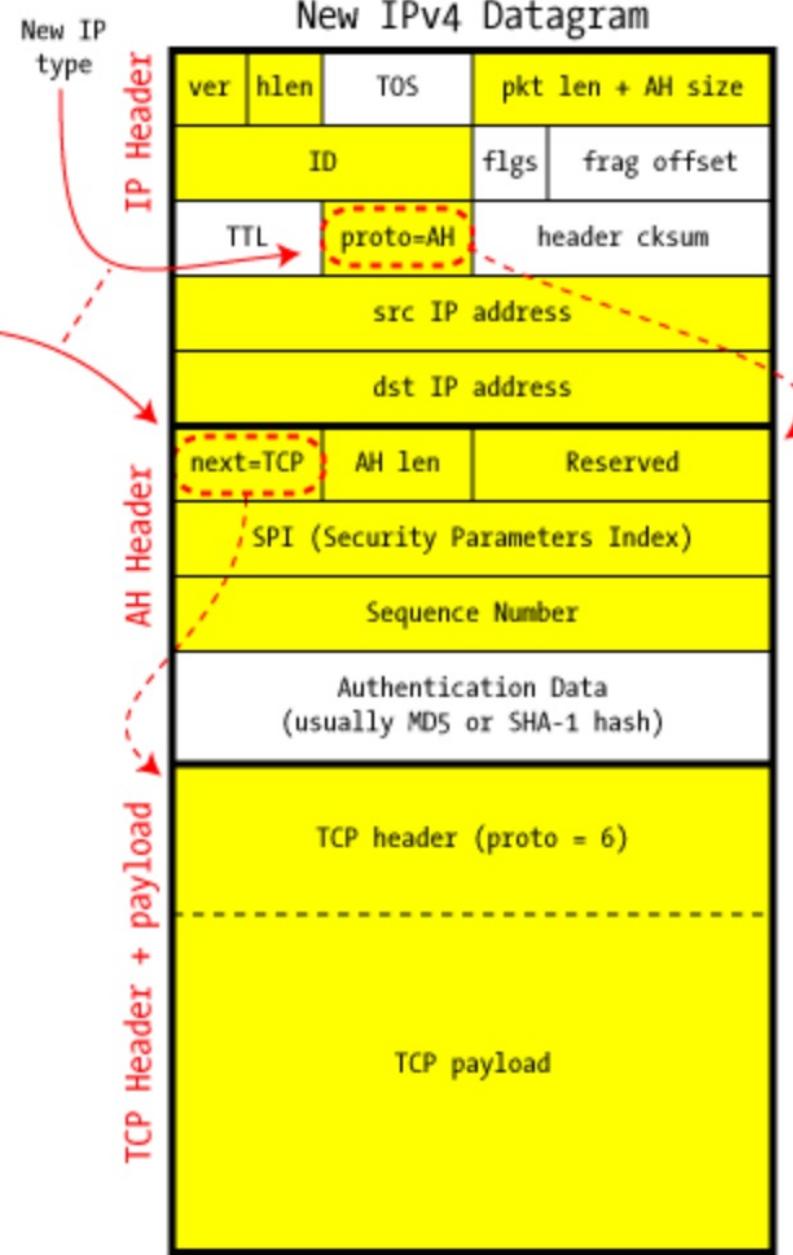


IPSec in AH Transport Mode

Original IPv4 Datagram



New IPv4 Datagram

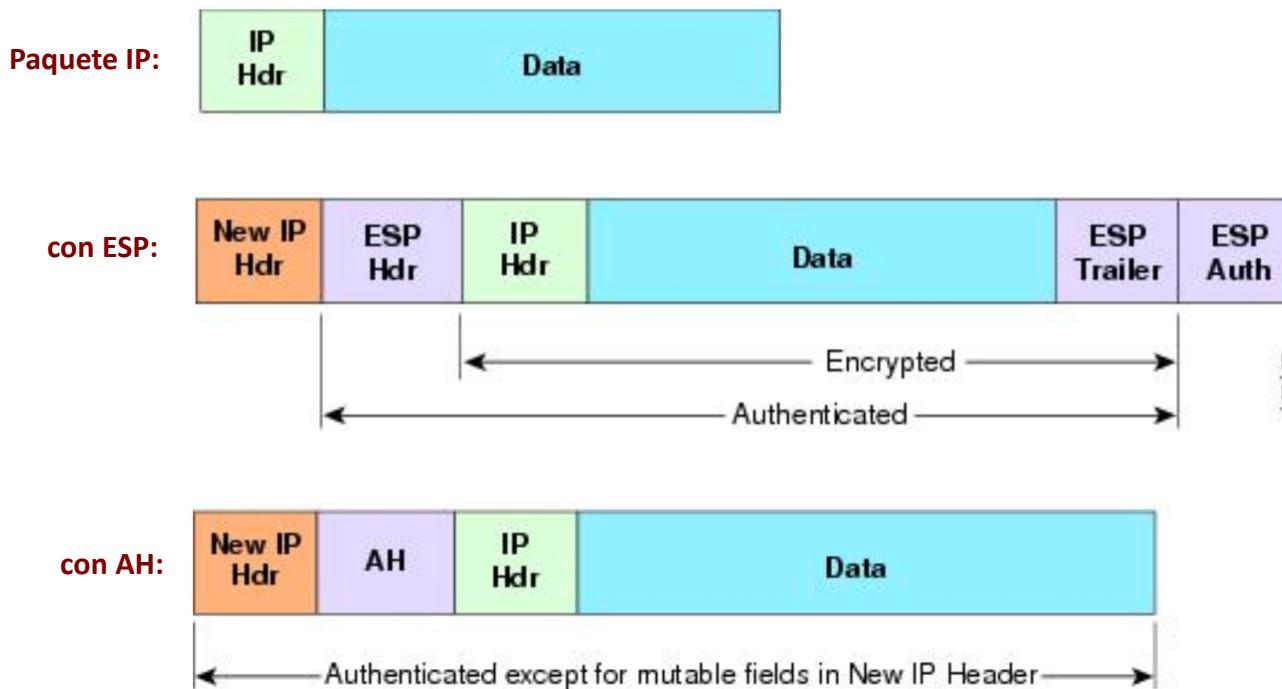


Protected by
AH Auth Data

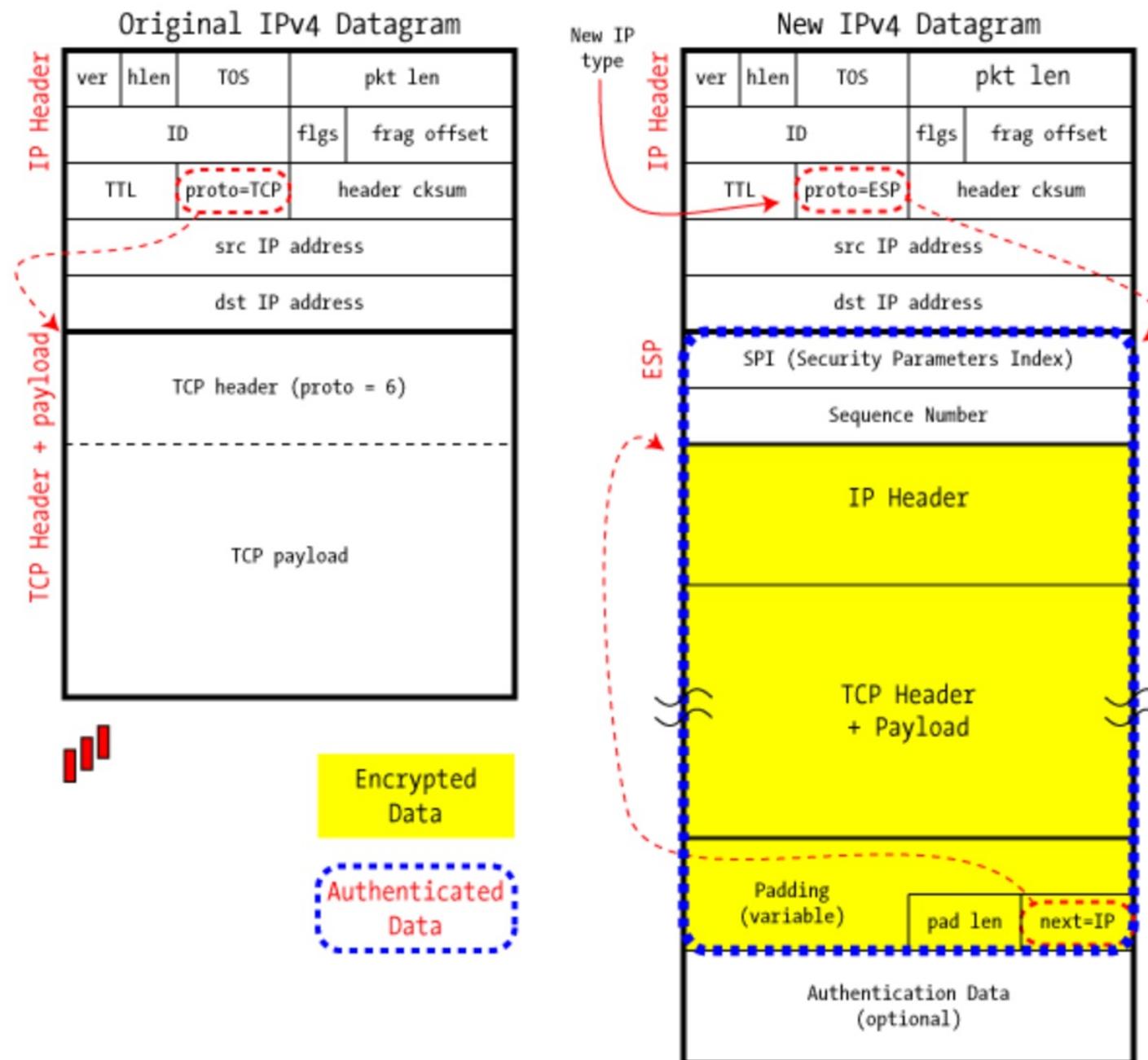
Modo túnel

– En este modo de uso:

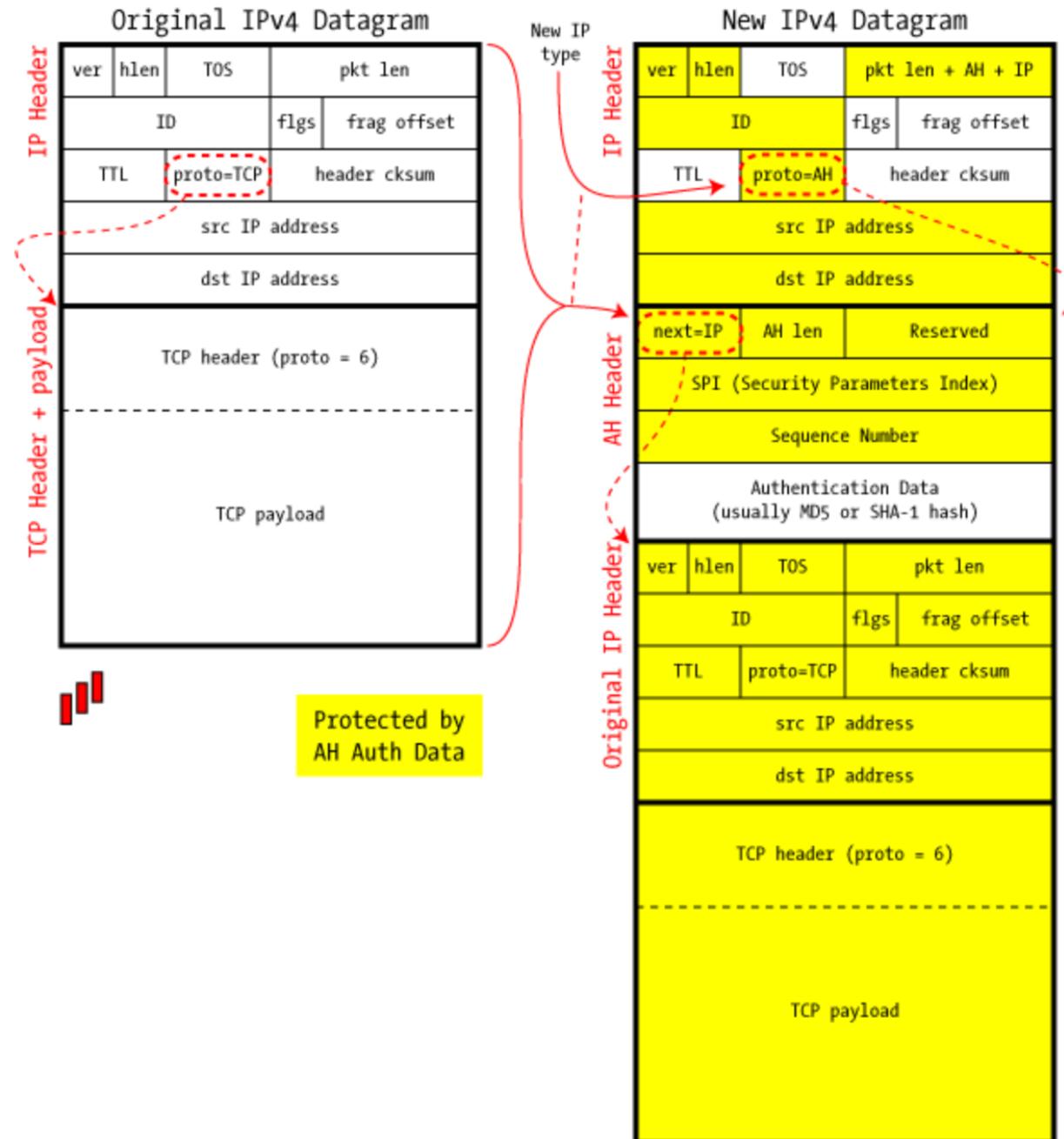
- **si se usa ESP:** se cifra y opcionalmente autentica todo el paquete IP original (paquete interno), incluyendo la cabecera de ese paquete original
- **si se usa AH:** se autentica todo el paquete original y algunas partes de la nueva cabecera externa



IPSec in ESP Tunnel Mode

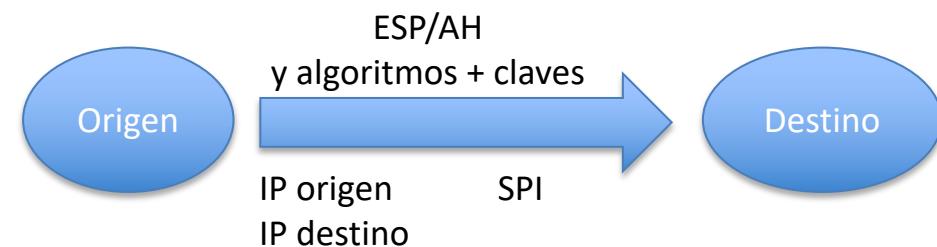


IPSec in AH Tunnel Mode



Asociaciones de seguridad en IPSec

- Para activar IPSec es necesario configurar:
 - El origen y el destino de los paquetes IPSec
 - El modo de autenticación de los mensajes
 - P. ej. el HMAC basado en MD5 o SHA
 - El algoritmo de cifrado
 - P. ej: DES, 3DES, AES y Blowfish
 - El índice de parámetro de seguridad (**SPI - Security Parameter Index**)
 - Es un número de 32 bits único para cada asociación de seguridad definida para ESP / AH
 - Un número de secuencia única de paquetes para controlar los ataques *replay*
 - *Sequence Number* – de las transparencias anteriores
 - Sólo se aceptan paquetes que tienen un número actual de secuencia o posterior, las anteriores se descartan
- A toda esta información se le conoce con el nombre de **Asociación de seguridad (SA – Security Associations)**
 - OJO, sólo se protege un sentido
 - El emisor y el receptor deben aplicar la misma SA, pero teniendo en cuenta el destino y el origen



No.	Time	Source	Destination	Protocol	Length	Info
134	215.661944	190.0.0.1	190.0.0.14	ESP	134	ESP (SPI=0x00000070)
135	216.661895	190.0.0.1	190.0.0.14	ESP	134	ESP (SPI=0x00000070)
136	217.663971	190.0.0.1	190.0.0.14	ESP	134	ESP (SPI=0x00000070)
137	218.681817	190.0.0.1	190.0.0.14	ESP	134	ESP (SPI=0x00000070)
138	219.681772	190.0.0.1	190.0.0.14	ESP	134	ESP (SPI=0x00000070)
139	220.681692	190.0.0.1	190.0.0.14	ESP	134	ESP (SPI=0x00000070)
140	230.684156	190.0.0.1	190.0.0.15	ESP	122	ESP (SPI=0x00000071)
141	231.683968	190.0.0.1	190.0.0.15	ESP	122	ESP (SPI=0x00000071)
142	232.683915	190.0.0.1	190.0.0.15	ESP	122	ESP (SPI=0x00000071)
143	233.683761	190.0.0.1	190.0.0.15	ESP	122	ESP (SPI=0x00000071)

► Frame 143: 122 bytes on wire (976 bits), 122 bytes captured (976 bits)
 ► Ethernet II, Src: Dell_4a:d7:0a (00:11:43:4a:d7:0a), Dst: 00:0c:29:00:00:15 (00:00:00:00:00:15)
 ▼ Internet Protocol Version 4, Src: 190.0.0.1, Dst: 190.0.0.15
 0100 = Version: 4
 0101 = Header Length: 20 bytes
 ► Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECN)
 Total Length: 108
 Identification: 0x0003 (3)
 ► Flags: 0x02 (Don't Fragment)
 Fragment offset: 0
 Time to live: 64
 Protocol: Encap Security Payload (50)
 ► Header checksum: 0xbe4c [validation disabled]
 Source: 190.0.0.1
 Destination: 190.0.0.15
 [Source GeoIP: Unknown]
 [Destination GeoIP: Unknown]
 ▼ Encapsulating Security Payload
 ESP SPI: 0x00000071 (113)
 ESP Sequence: 6

0000	00 00 00 00 00 15 00 11 43 4a d7 0a 08 00 45 00E. l..@2
0010	00 6c 00 03 40 00 40 32 be 4c be 00 00 01 be 00	.L.....
0020	00 0f 00 00 00 71 00 00 00 06 08 00 0a 18 7e 64	...q.....~d
0030	00 04 3b a9 f9 43 44 8f 0b 00 08 09 0a 0b 0c 0d	...;..CD.....
0040	0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d
0050	1e 1f 20 21 22 23 24 25 26 27 28 29 2a 2b 2c 2d	.. !#\$% &'()*+,-./012345 67.....
0060	2e 2f 30 31 32 33 34 35 36 37 01 02 02 01 ab f8	...0)g/. l.
0070	af 87 f4 4f 29 67 2f c4 6c c8	

Asociaciones de seguridad en IPSec

- Las SAs se almacenan en una **base de datos de asociaciones de seguridad (SAD)**
- Cada entrada en la SAD existen varios campos:
 - *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)

Políticas de seguridad en IPSec

- 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 son las mismas
 - Los protocolos y puertos a proteger
 - Algunas implementaciones no permiten la definición de protocolos específicos a proteger
 - Si no se especifica nada, se protege todo el tráfico
 - El modo de protección: túnel o transporte

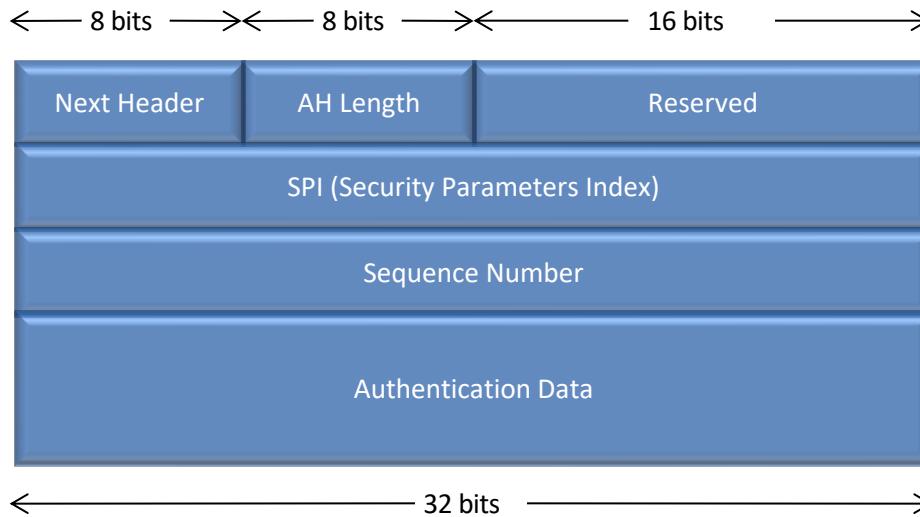
Políticas de seguridad en IPSec

- Un ejemplo de SPD es el siguiente:

Protocol	Local IP	Port	Remote IP	Port	Action	Comment
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

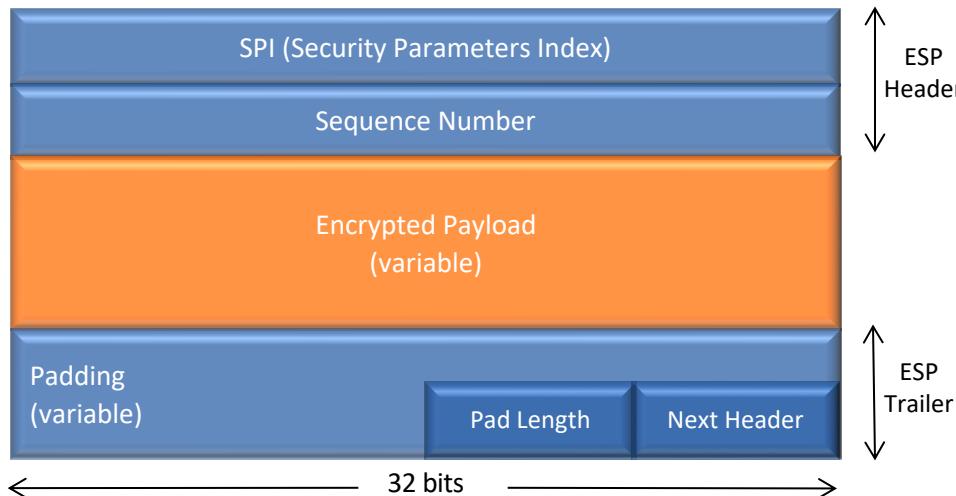
SPI (Security Parameters Index) (32 bits): Identifica una “asociación de seguridad”

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

Cabeceras AH y ESP

- Encapsulating Security Payload – ESP (sólo cifrado)



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

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

Encrypted Payload (32 bits): Segmento de nivel de transporte (modo transporte) o paquete IP (modo túnel) protegido por medio de cifrado

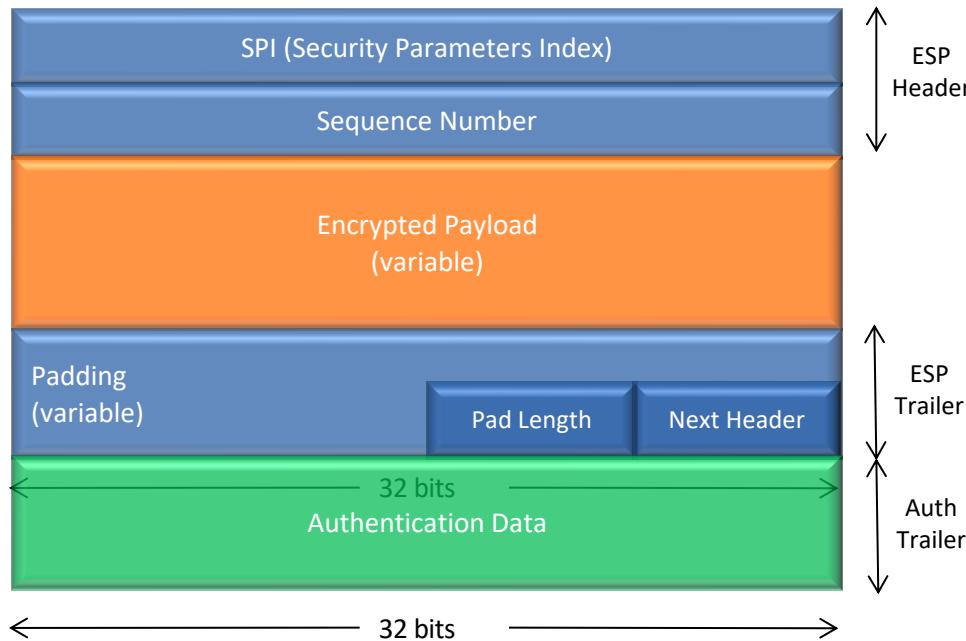
Padding (0-255 bytes): Espacio adicional incluido porque los algoritmos de encriptación basados en bloque pueden requerir espacios diferentes

Pad Length (8 bits): longitud del “Padding”

Next Header (8 bits): guarda el tipo de la siguiente cabecera (IP, TCP, UDP, etc.)

Cabeceras AH y ESP

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



Cabeceras AH y ESP

- Las distintas opciones al seleccionar tipos de cabeceras:

	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 for 192.168.1.100

# 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 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 compresión

ESP



SAs: ESP y la clave para el cifrado

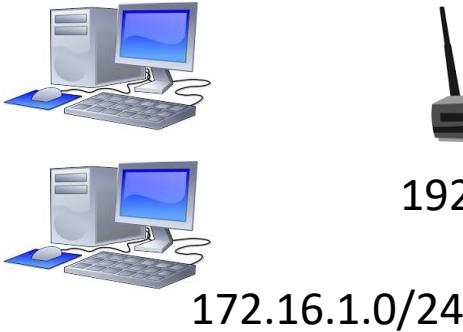
Protocolo y puerto



SPDs: para ambos lados

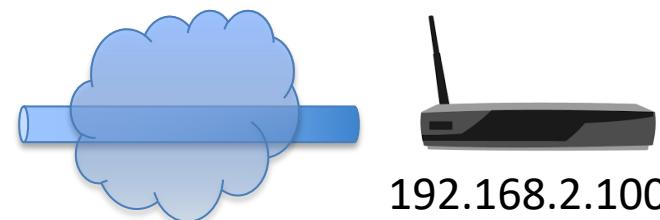
Dirección de la acción de la política

Ejemplo: modo túnel

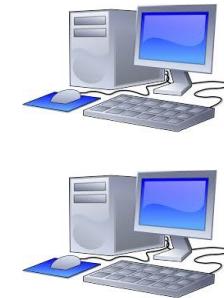


192.168.1.100

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;
```

Protocolo Internet Key Exchange (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** (Key Exchange Protocol)
 - **ISAKMP** (Internet Security Association and Key Management Protocol)

Protocolo Internet Key Exchange (IKE)

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

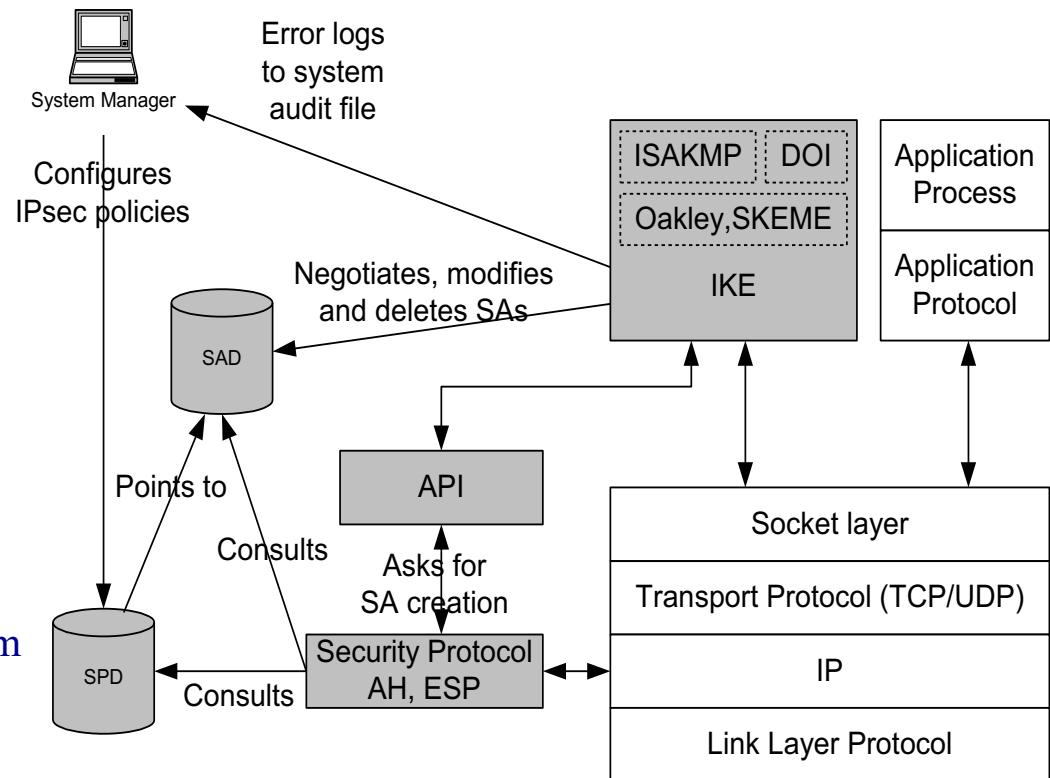
Oakley = Key Exchange Protocol

SA = Security Association

SAD = Security Association Database

SKEME = Secure Key Exchange Mechanism

SPD = Security Policy Database



- Con ISAKMP, IKE funciona en dos fases:
 - Fase 1: autentica cada parte
 - Fase 2: negocia y establece las SAs de IPSEC

Protocolo Internet Key Exchange (IKE)

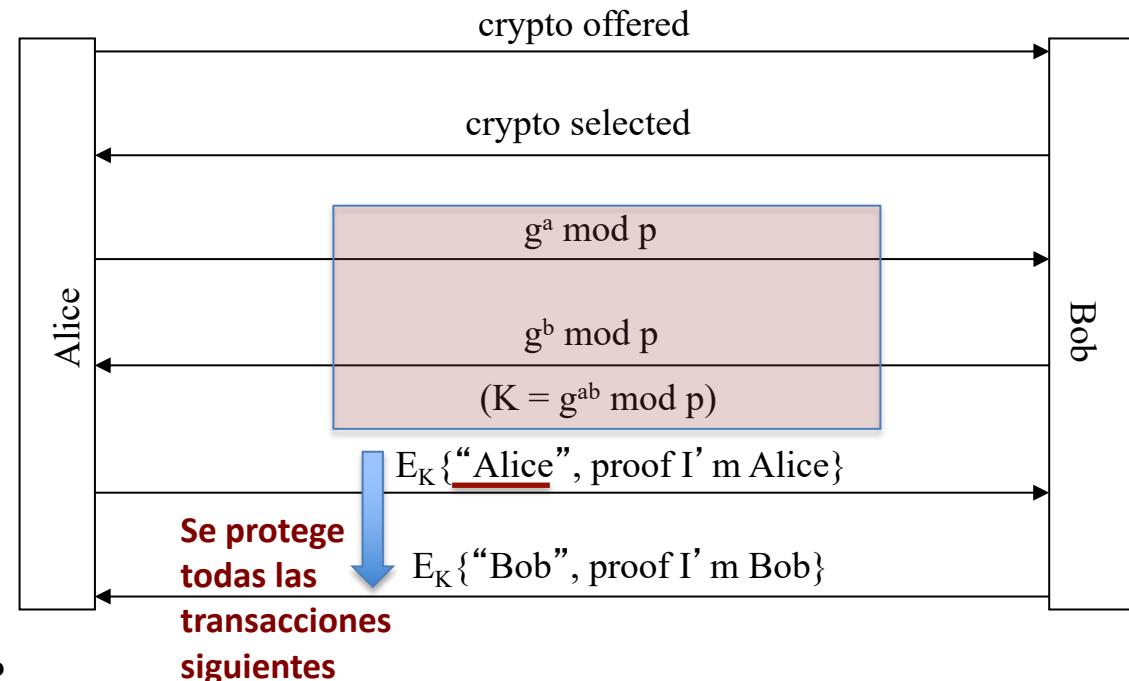
- Fase 1: **autentica cada parte**
 - La autenticación de las partes de la comunicación
 - suele basarse en claves compartidas
 - claves RSA
 - certificados x.509
 - Esta fase soporta dos modos de autenticación: **agresivo y principal**
 - El modo agresivo:
 - proceso simple y sólo usa la mitad de los mensajes para finalizar el proceso rápidamente
 - Sin embargo, no soporta la protección completa de las identidades de cada parte de la comunicación y transmite la identidad del cliente en claro
 - El modo principal:
 - realiza el proceso de autenticación de forma lenta y segura

Protocolo Internet Key Exchange (IKE)

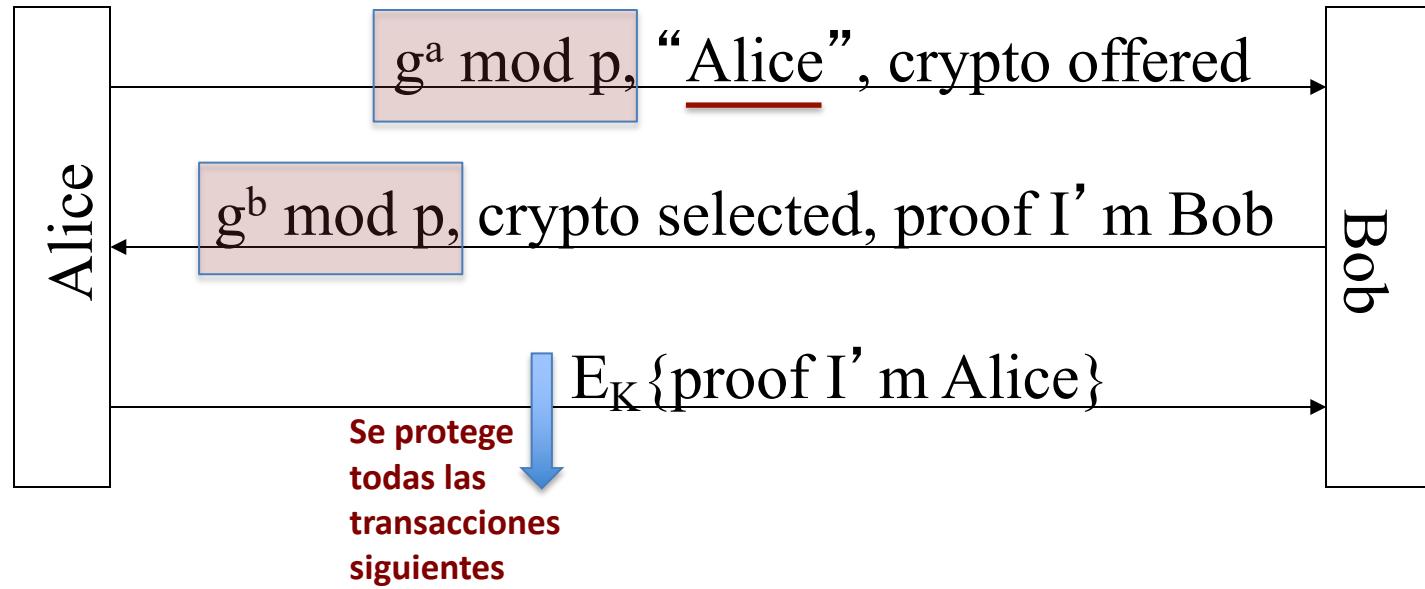
- Fase 2: el nuevo SA ISAKMP es empleado para **negociar** y establecer los SAs de IPSec
 - En esta fase el protocolo IKE intercambia propuestas de SA
 - negocia asociaciones de seguridad basándose en el ISAKMP SA inicial, y
 - establece la clave de sesión
 - Las claves de las SA se derivan de
 - las claves de la primera fase, los nonces y los SPI, o usando un nuevo DH

Protocolo IKE - 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
 - Autenticación
 - Negociación de las SA
 - Diffie-Hellman e intercambio de nonce



Protocolo IKE - Fase 1 a Fase 2: Modo Agresivo

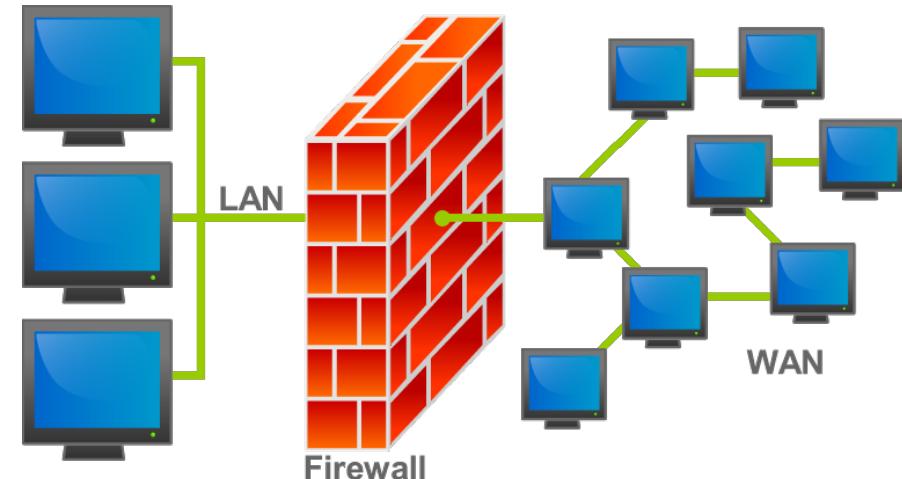


FIREWALLS EN REDES

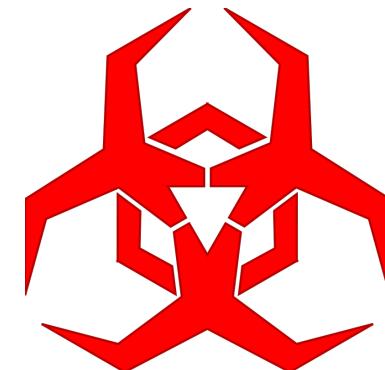
¿Qué es un cortafuego?

Firewalls

- Un cortafuego (firewall) es un sistema (software o hardware) que establece un conjunto de políticas de control

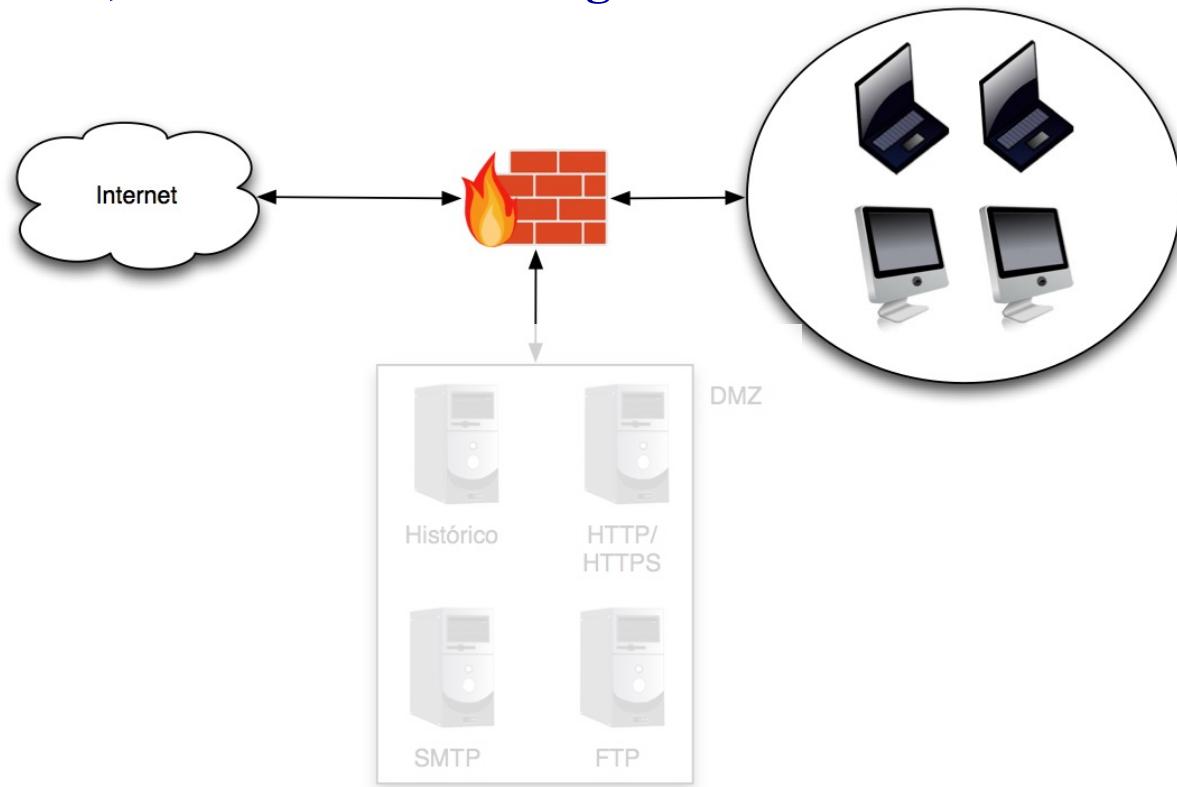


- Considerada la primera herramienta de seguridad para las redes y surge a finales de 1980 (aprox. a principios de los noventa)
 - cuando el Internet comienza a aplicarse a nivel de usuario y cuando surgen los primeros ataques a nivel de red como el *gusano Morris* (por su autor Robert Morris)
 - El malware Morris apareció en 1988 e infectó aprox. 6000 de los 60.000 servidores conectados a la red
 - aunque no era extremadamente peligroso afectó muchas máquinas del mundo



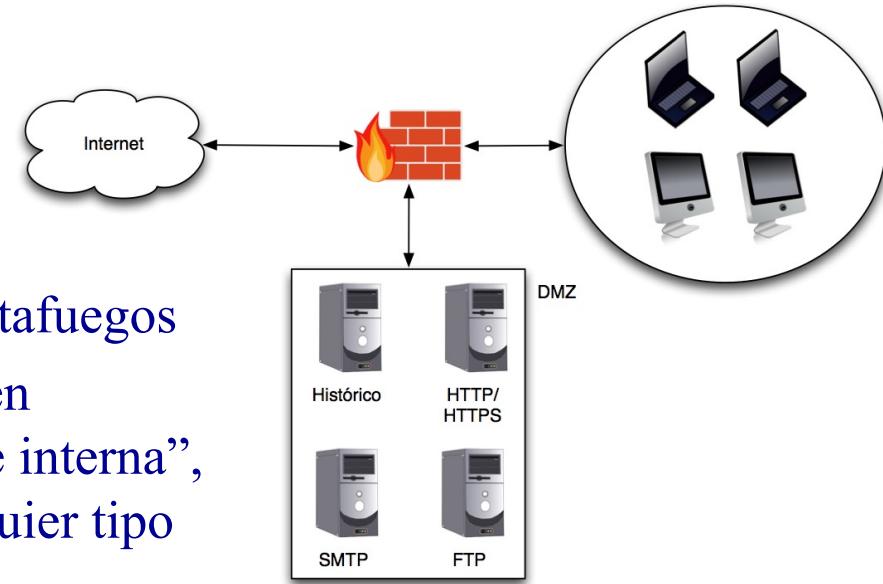
Firewalls

- Un firewall define:
 - El espacio protegido, denominado **perímetro de seguridad**, suele ser propiedad de la misma organización, y
 - la protección se realiza generalmente contra una red externa (ej. el Internet) no confiable, llamada **zona de riesgo**



Firewalls

- **Zonas desmilitarizadas (De-Militarized Zones, DMZ):**
 - Añaden un nivel específico de seguridad en las arquitecturas de cortafuegos
 - Situando una subred DMZ (basado en servidores) entre las redes “externa e interna”, de forma que aísla y/o protege cualquier tipo de acceso a los hosts del sistema
- Se establecen configuraciones específicas de firewall tal que:
 - La conexión de las redes externas al DMZ son permisivas
 - La conexión del DMZ a la intranet son prohibitivas



Tipos de firewalls

- Hay cuatro generaciones de firewalls principales:
 - **Primera generación:**
 - filtrado de paquetes (*packet filter*)
 - **Segunda generación:**
 - cortafuegos de estado (*stateful inspection*)
 - **Tercer generación:**
 - cortafuegos de aplicación (*application filtering*)
 - **Cuarta generación:**
 - cortafuegos de nueva generación (*Next-Generation Firewall, NGFW*)

Tipos de firewalls

- **Filtrado de paquetes (*packet filter*):**

- Definido en 1988 por Digital Equipment Corporation (DEC) para inspeccionar cada paquete entrante/saliente, usando reglas sencillas de filtrado y a parámetros específicos como: IP del emisor, IP del destino, tipo de protocolo, puerto, DNS

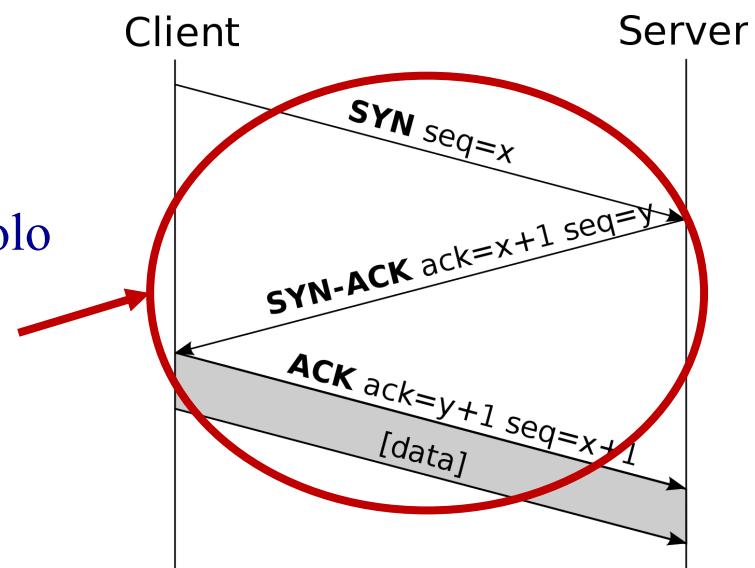
SOLO MIRA EN LA CABECERA DEL
PAQUETE (IPs, puertos, MAC,
protocolo, etc.)

Tipos de firewalls

- **Cortafuegos de estado (*stateful inspection*):**
 - Definido en 1989-1990 por AT&T Bell para hacer lo mismo que los cortafuegos de primera generación, pero considerando el estado de cada paquete los **flags**: SYN, ACK, ...

SOLO MIRA EN LA CABECERA DEL
PAQUETE (IPs, puertos, MAC,
protocolo, etc.) + **FLAGS**

- Los estados ayudan a:
 - Controlar el estado de cada protocolo (ej. conexiones TCP)

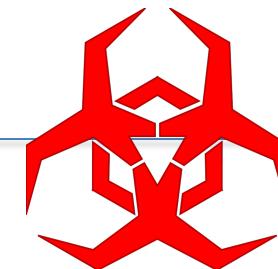


Tipos de firewalls

- **Cortafuegos de aplicación (*application filtering*):**
 - El objetivo era mejorar las capacidades de los cortafuegos de segunda generación y permitir:
 - **DPI “Deep Packet Inspection”** para detectar irregularidades o infección malware en los contenidos de los paquetes
 - Combinar el firewall con otros sistemas de protección integrados como:
 - Antimalware y sistemas de detección y prevención de intrusiones
 - VPN, TLS

SOLO MIRA EN LA CABECERA DEL
PAQUETE (IPs, puertos, MAC,
protocolo, etc.) + **FLAGS**

DPI



Tipos de firewalls

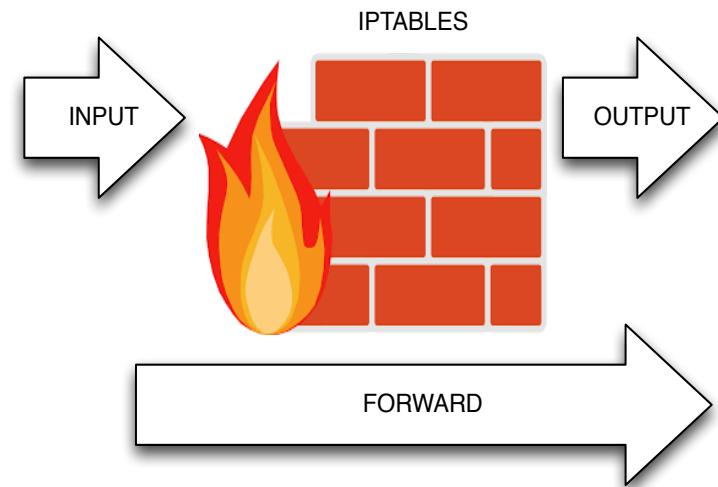
- **Cortafuegos de nueva generación (*Next-Generation Firewall, NGFW*):**
 - El objetivo era mejorar para incorporar nuevos servicios:
 - Gestión y monitorización de usuarios y aplicaciones (email, app de descarga, compartición de recursos y colaborativos,...)
 - Eliminación de apps no deseadas para reducir la superficie de ataque (especialmente los APTs)
 - Validación de aplicaciones, el aislamiento de los datos, el control sobre las aplicaciones no deseadas, etc.
 - Protección a los sistemas de red en la nube
 - Control desde plataformas móviles
 - Etc.

IPTABLES: sintaxis y políticas

- IPtables ilustra la siguiente sintaxis de comandos:

```
[root@localhost alumno]# #iptables -A <chain> -j <target>
```

- <chain>: define una cadena de reglas que pueden ser del tipo INPUT, OUTPUT y FORWARD



- -A <chain>: agrega la regla a la cadena
 - -D <chain>: elimina la regla
 - -I <chain>: inserta una nueva regla en una posición determinada
- -j <target>: especifica el fin de la regla; es decir, qué hacer si un paquete coincide con la regla, como, por ejemplo: ACCEPT y DROP

IPTABLES: sintaxis y políticas

- **Eliminar cualquier tráfico telnet entrante:**
iptables -I INPUT -p tcp --dport 23 -j DROP
- **Eliminar cualquier tráfico web saliente:**
iptables -I OUTPUT -p tcp --dport 80 -j DROP
- **Eliminar cualquier tráfico saliente a la IP 192.168.0.1**
iptables -I OUTPUT -p tcp --dest 192.168.0.1 -j DROP
- **Permitir cualquier tráfico web entrante:**
iptables -I INPUT -p tcp --dport 80 -j ACCEPT
- **Permitir tráfico entrante al puerto 25 (del servidor SMTP):**
iptables -A INPUT -s 0.0.0.0/0 -p tcp --dport 25 -j ACCEPT
- **Permitir tráfico pop3 entrante:**
iptables -A INPUT -s 0.0.0.0/0 -p tcp --dport 110 -j ACCEPT

IPTABLES: sintaxis y políticas

- **Permitir tráfico HTTPS (443) entrante desde la IP 11.3.2.1**
iptables -I INPUT -s 11.3.2.1 -p tcp --dport 443 -j ACCEPT
- **Denegar el tráfico saliente a la red 192.2.4.0-192.2.4.255:**
iptables -I OUTPUT -d 192.2.4.6.0/24 -j DROP
- **Bloquear cualquier tráfico saliente de un particular dominio o host:**
 - 1) host -t a elpais.es ➔ elpais.es tiene IP:78.120.152.203
 - 2) iptables -A OUTPUT -d 78.120.152.203 -j DROP

IPTABLES: flush y políticas por defecto

- Política establecida por defecto:
 - Para cada cadena <chain> se define una política inicial que puede ser ACCEPT o DROP, y para ello se usa la opción –P

```
### FLUSH de reglas
```

```
iptables -F  
iptables -X  
iptables -Z  
iptables -t nat -F
```

```
### Políticas por defecto
```

```
iptables -P INPUT ACCEPT  
iptables -P OUTPUT ACCEPT  
iptables -P FORWARD ACCEPT  
iptables -t nat -P PREROUTING ACCEPT  
iptables -t nat -P POSTROUTING ACCEPT
```

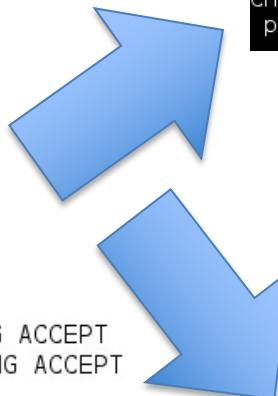
IPTABLES: flush y políticas por defecto

FLUSH de reglas

```
iptables -F  
iptables -X  
iptables -Z  
iptables -t nat -F
```

Políticas por defecto

```
iptables -P INPUT ACCEPT  
iptables -P OUTPUT ACCEPT  
iptables -P FORWARD ACCEPT  
iptables -t nat -P PREROUTING ACCEPT  
iptables -t nat -P POSTROUTING ACCEPT
```



```
[alumno@router ~]$ su  
Contraseña:  
[root@router alumno]# iptables -nvL  
Chain INPUT (policy ACCEPT 20845 packets, 27M bytes)  
pkts bytes target prot opt in out source destination  
  
Chain FORWARD (policy ACCEPT 84 packets, 5676 bytes)  
pkts bytes target prot opt in out source destination  
  
Chain OUTPUT (policy ACCEPT 14649 packets, 1370K bytes)  
pkts bytes target prot opt in out source destination
```

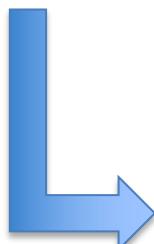
```
[root@router alumno]# iptables -t nat -nvL  
Chain PREROUTING (policy ACCEPT 3 packets, 718 bytes)  
pkts bytes target prot opt in out source destination  
  
Chain INPUT (policy ACCEPT 3 packets, 718 bytes)  
pkts bytes target prot opt in out source destination  
  
Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)  
pkts bytes target prot opt in out source destination  
  
Chain POSTROUTING (policy ACCEPT 0 packets, 0 bytes)  
pkts bytes target prot opt in out source destination
```

IPTABLES: INPUT, OUTPUT, FORWARD - RESTRICTIVO

- **Ejemplo 1: NO permitir tráfico entrante y saliente, y para todas las redes (incluyendo al Router)**

```
[root@router alumno]# iptables -P INPUT DROP
[root@router alumno]# iptables -P OUTPUT DROP
[root@router alumno]# ping 192.168.0.15
PING 192.168.0.15 (192.168.0.15) 56(84) bytes of data.
^C
--- 192.168.0.15 ping statistics ---
0 packets transmitted, 0 received

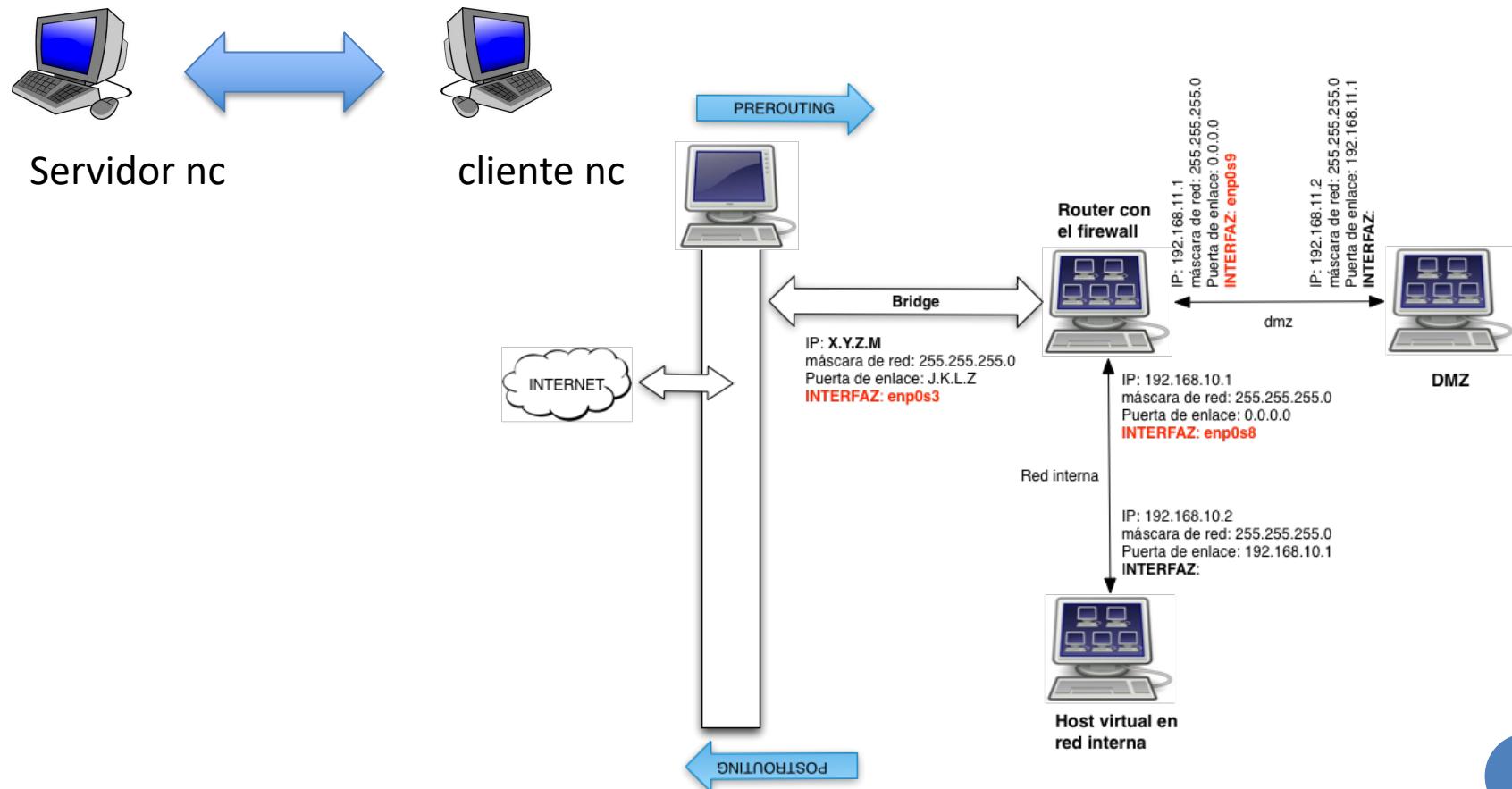
[root@router alumno]# ping 192.168.11.1
PING 192.168.11.1 (192.168.11.1) 56(84) bytes of data.
^C
--- 192.168.11.1 ping statistics ---
0 packets transmitted, 0 received
```



```
[root@router sysctl.d]# iptables -nL
Chain INPUT (policy DROP)
target     prot opt source          destination
Chain FORWARD (policy ACCEPT)
target     prot opt source          destination
Chain OUTPUT (policy DROP)
target     prot opt source          destination
```

IPTABLES: INPUT, OUTPUT, FORWARD

- Ejemplo 2: crear un cliente-servidor netcat en las máquinas instaladas en la red DMZ y la red interna para que se comuniquen, sólo y únicamente, por el puerto 55555



IPTABLES: INPUT, OUTPUT, FORWARD

- **Ejemplo 2:** crear un cliente-servidor netcat en las máquinas instaladas en la red DMZ y la red interna para que se comuniquen, sólo y únicamente, por el puerto 55555

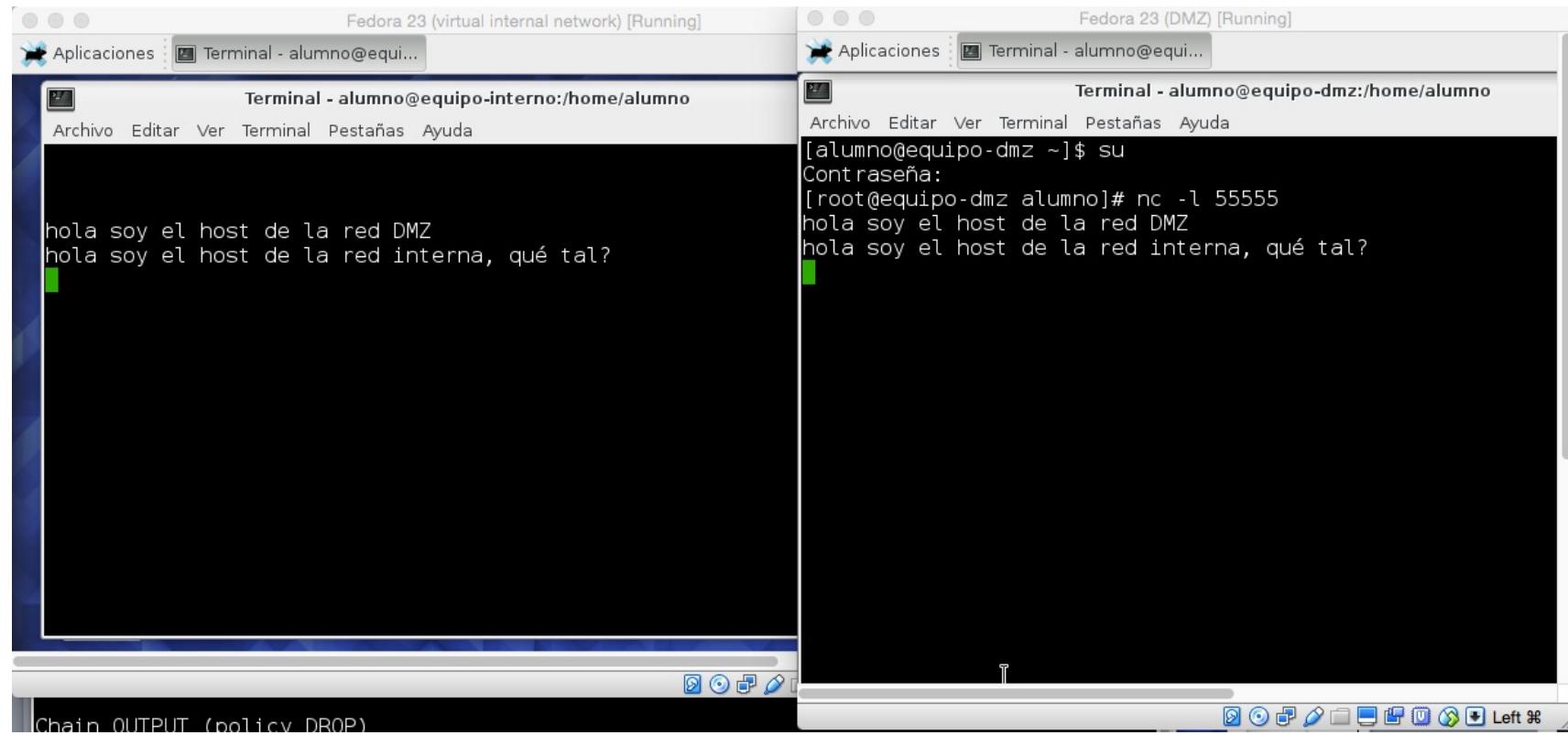
```
### ASEGURAR EL FORWARDING
sysctl -w net.ipv4.ip_forward=1
echo "net.ipv4.ip_forward = 1" > /etc/sysctl.d/ip_forwarding.conf

### FLUSH de reglas
iptables -F
iptables -X
iptables -Z
iptables -t nat -F

### Políticas por defecto
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
iptables -P FORWARD DROP
iptables -P FORWARD DROP
iptables -t nat -P PREROUTING ACCEPT
iptables -t nat -P POSTROUTING ACCEPT

## PROPORCIONAR LA COMUNICACIÓN ENTRE LAN-LAN PERO SOLO POR EL PUERTO 55555
iptables -A FORWARD -i enp0s9 -o enp0s8 -p tcp --dport 55555 -j ACCEPT
iptables -A FORWARD -i enp0s8 -o enp0s9 -p tcp --sport 55555 -j ACCEPT
```

IPTABLES: INPUT, OUTPUT, FORWARD



Cliente nc – Router

Servidor nc – Red DMZ

IPTABLES: PREROUTING Y POSTROUTING

- La comunicación hacia o desde redes públicas requiere:

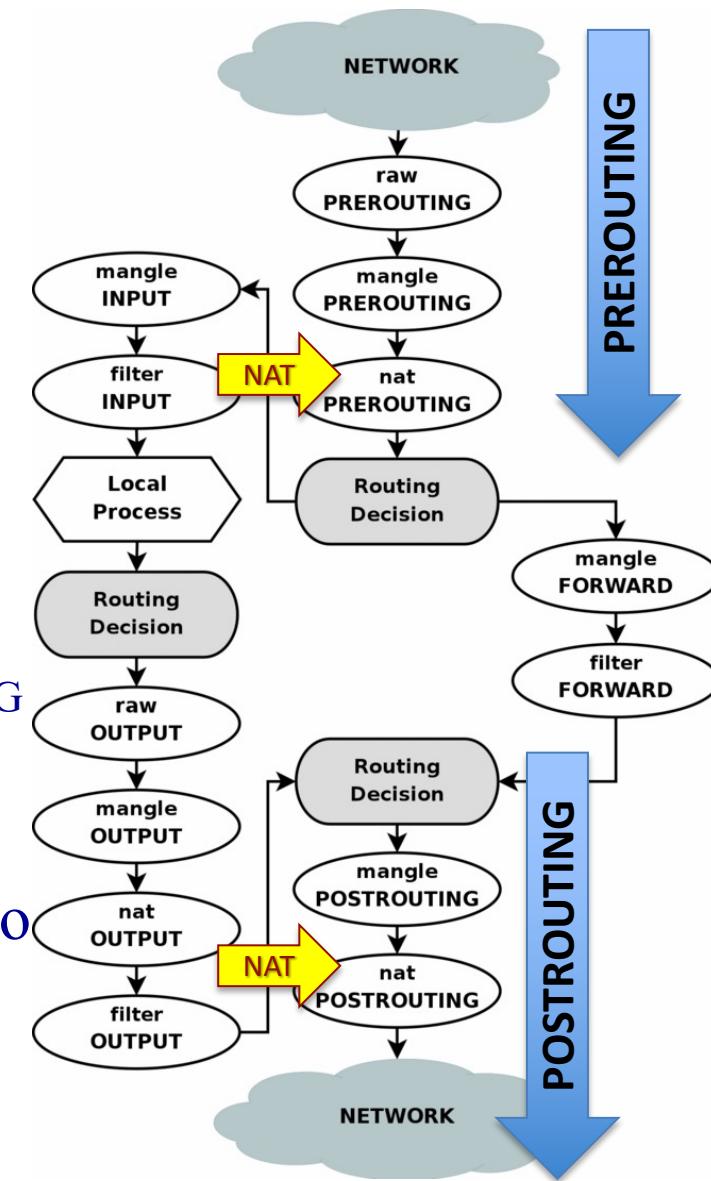
– PREROUTING:

- EXTERIOR → INTERIOR
 - NAT: exterior → interior
- PREROUTING → INPUT / FORWARD

– POSTROUTING

- INTERIOR → EXTERIOR
 - NAT: interior → exterior
(requiere ENMASCARAMIENTO)
- OUTPUT / FORWARD → POSTROUTING

- Como se aprecia en la figura, al trabajar con paquetes que vienen de o van hacia redes externas, hay que aplicar NAT (Network Address Translation)



IPTABLES: PREROUTING Y POSTROUTING

- La comunicación hacia o desde redes públicas requiere:
 - **POSTROUTING**: INTERIOR → EXTERIOR

- Proceso de **enmascaramiento** para poder trabajar con el NAT

```
[root@router alumno]# iptables -t nat -A POSTROUTING -o enp0s3 -j MASQUERADE
```



- El orden de las reglas importa:
 - Primero **FORWARD/OUTPUT** y
 - Despues el **POSTROUTING**

- **PREROUTING**: EXTERIOR → INTERIOR

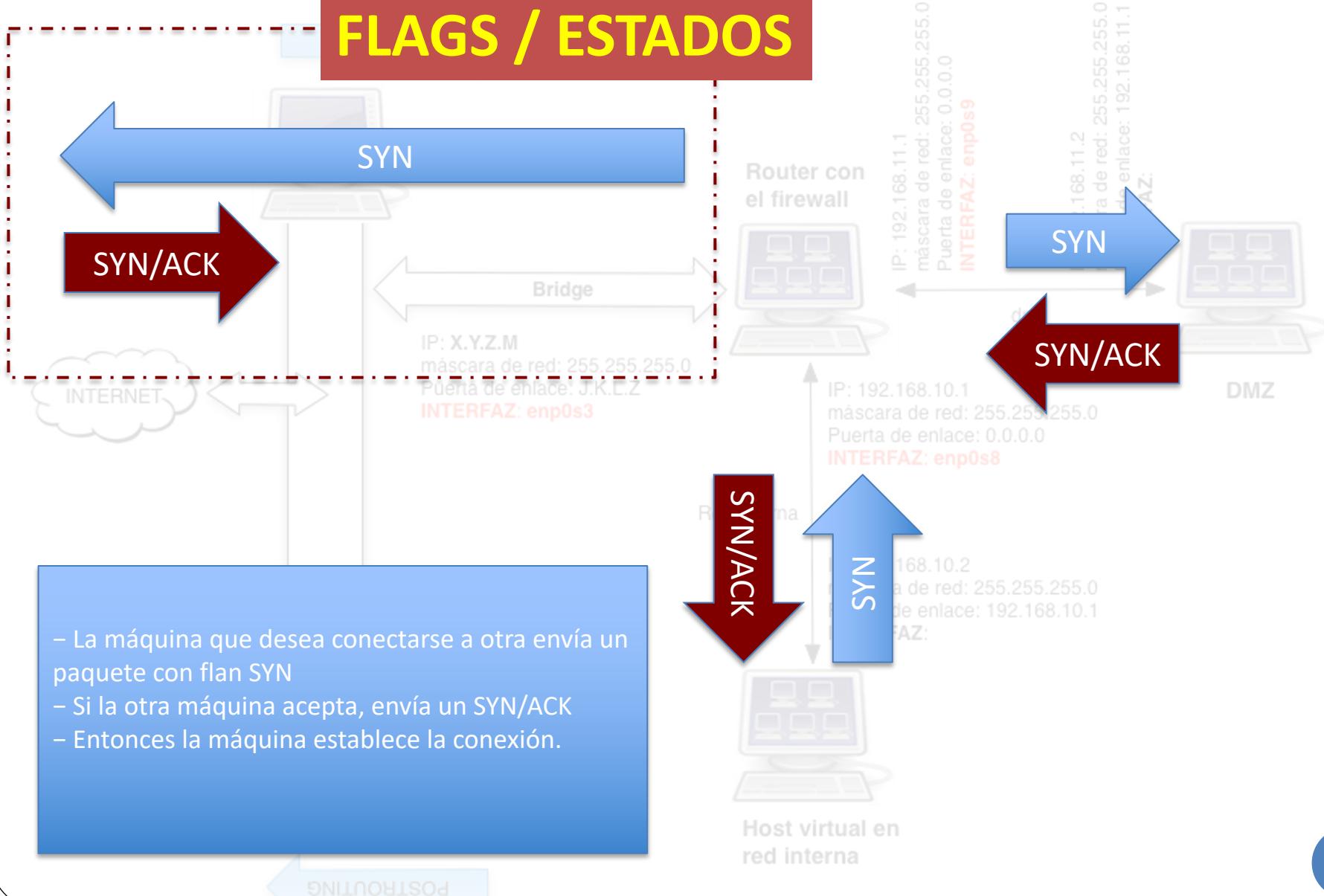


```
[root@router alumno]# iptables -t nat -A PREROUTING -i enp0s3 -d X.Y.Z.M -p tcp  
--dport 443-j DNAT --to 192.168.11.2:443
```

- El orden de las reglas importa:
 - Primero el **PREROUTING** y
 - Despues **FORWARD/INPUT**

Three-way-handshake en TCP/IP y problema a controlar

FLAGS / ESTADOS

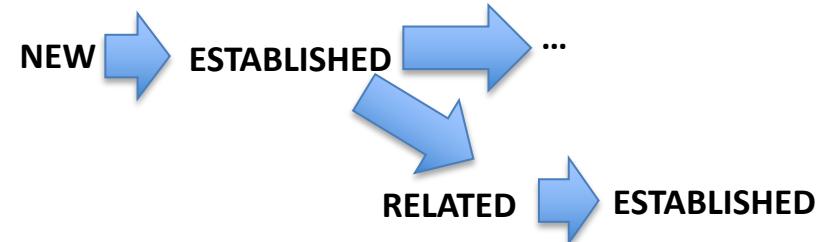


Three-way-handshake en TCP/IP y problema a controlar



- NEW: se ha establecido una nueva conexión y se requiere comunicación bidireccional (ej. SYN -- SYN/ACK)
- ESTABLISHED: el paquete está asociado con una conexión ya establecida pero se requiere que haya paquetes en ambas direcciones (ej. SYN -- SYN/ACK)
- RELATED: el paquete está comenzando una nueva conexión pero está asociada a una conexión ya existente, como puede ser una comunicación FTP

NEW → ESTABLISHED



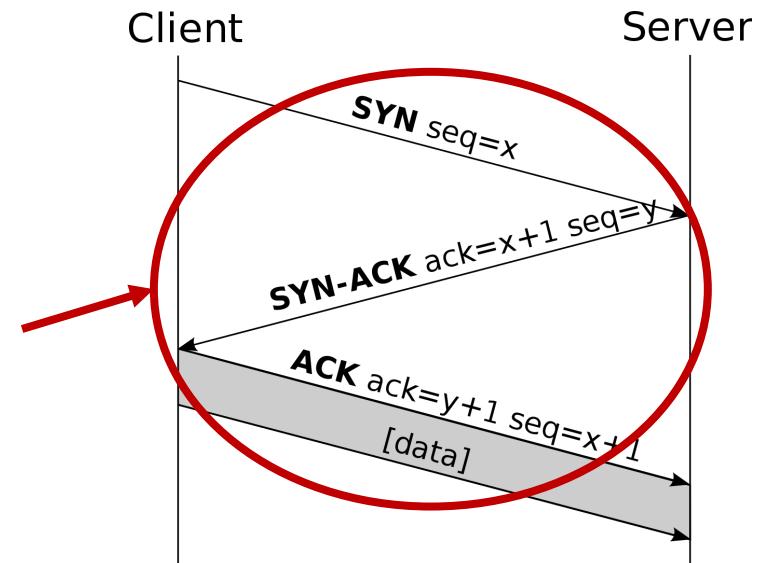
Three-way-handshake en TCP/IP y problema a controlar

- Un ejemplo: el 3-way-handshake de TCP
 - El firewall debe recordar estos estados y controlar en este caso el:

- SYN
- SYN/ACK
- ACK

Tal que SYN sería el NEW,
y el SYN/ACK y ACK serían
los RELATED

FLAGS / ESTADOS



Three-way-handshake en TCP/IP y problema a controlar

FLAGS / ESTADOS



Políticas por defecto

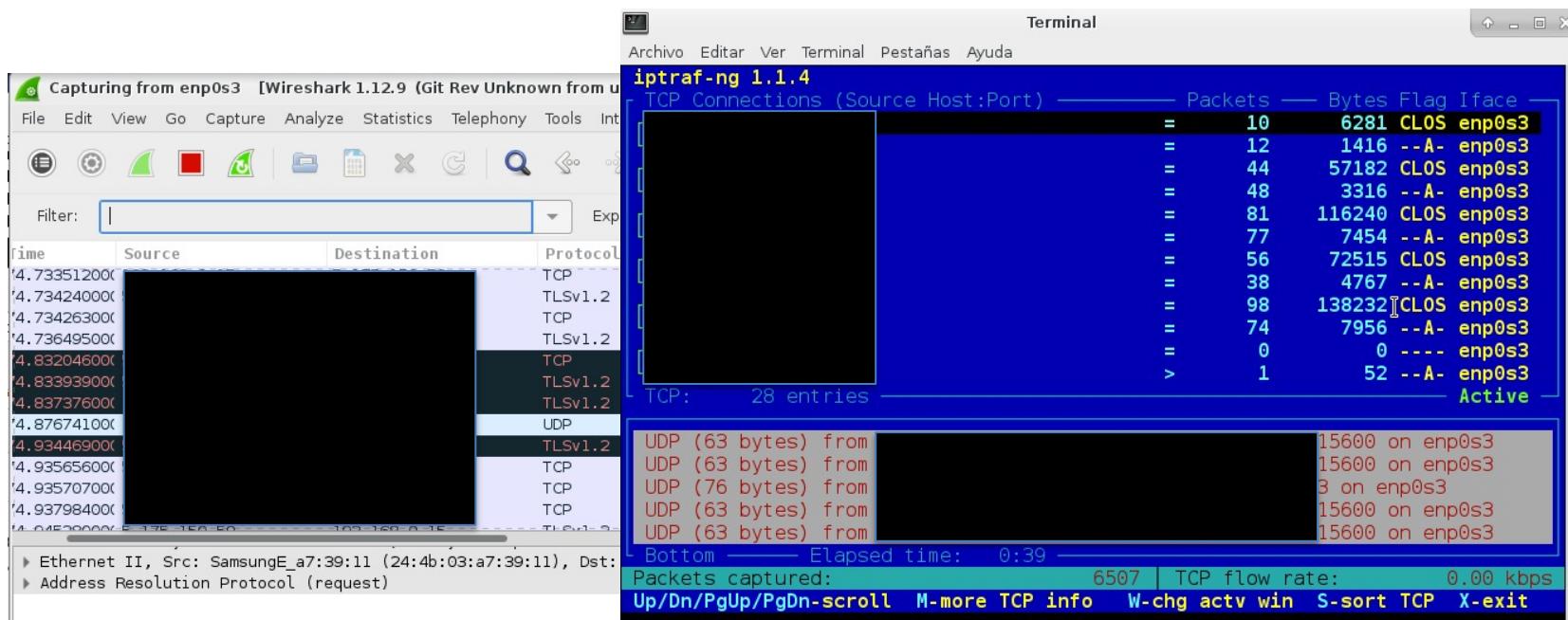
```
iptables -P INPUT ACCEPT  
iptables -P OUTPUT ACCEPT  
iptables -P FORWARD DROP  
iptables -t nat -P PREROUTING ACCEPT  
iptables -t nat -P POSTROUTING ACCEPT
```

Permitiendo comunicación con la red externa

```
iptables -A FORWARD -i enp0s9 -o enp0s3 -j ACCEPT  
iptables -A FORWARD -i enp0s3 -o enp0s9 -m state --state NEW,ESTABLISHED,RELATED -j ACCEPT  
iptables -A FORWARD -i enp0s8 -o enp0s3 -j ACCEPT  
iptables -A FORWARD -i enp0s3 -o enp0s8 -m state --state NEW,ESTABLISHED,RELATED -j ACCEPT  
iptables -t nat -A POSTROUTING -o enp0s3 -j MASQUERADE
```

```
iptables -nvL
```

Herramientas de monitorización



```
0000 ff ff ff ff ff ff [root@router pub]# tcpdump -D
0010 08 00 06 04 00 00 1.enp0s3 [Up, Running]
0020 00 00 00 00 00 00 2.enp0s8 [Up, Running]
0030 00 00 00 00 00 00 3.enp0s9 [Up, Running]
4.any (Pseudo-device that captures on all interfaces) [Up, Running]
5.lo [Up, Running, Loopback]
6.bluetooth-monitor (Bluetooth Linux Monitor)
7.usbmon1 (USB bus number 1)
[root@router pub]# tcpdump -i 3
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp0s9, link-type EN10MB (Ethernet), capture size 262144 bytes
18:39:45.534659 IP [REDACTED] 934 > mad06s09-in-f3.1e100.net.https: Flags
., seq 1367477101:1367477132, ack 1352877774, win 507, options [nop,nop,TS val
38372341 ecr 3259937716], length 31
18:39:45.534906 IP [REDACTED] 34 > mad06s09-in-f3.1e100.net.https: Flags
., seq 31, ack 1, win 507, options [nop,nop,TS val 38372341 ecr 3259937716],
length 0
18:39:45.534938 IP [REDACTED] > mad06s09-in-f3.1e100.net.https: Flags
```

SEGURIDAD EN LA CAPA DE ACCESO A RED: EL CASO DE LAS REDES INALÁMBRICAS

Consideraciones generales

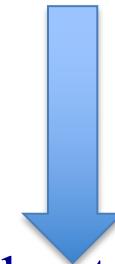
- Existe una amplia variedad de tecnologías y tipos de redes inalámbricas, entre las que destacan **Wi-Fi**, Bluetooth, WiMAX, Zigbee, etc.
 - Los requisitos de seguridad de estas redes son los mismos que en el caso de las redes cableadas
- Sin embargo, hay algunas amenazas de seguridad que aumentan cuando se consideran las redes inalámbricas
 - y además, hay otras amenazas que son propias/específicas de estos entornos
- La fuente de riesgo más significativa en las redes inalámbricas es el **medio de comunicación** subyacente
 - pero también hay riesgos de seguridad en los propios **protocolos inalámbricos** cuando no se diseñan apropiadamente

Consideraciones generales

- A grandes rasgos, el entorno inalámbrico tiene tres puntos de ataque:
 - cliente (o estación),
 - punto de acceso (AP) y
 - medio de transmisión
- Las posibles amenazas generales son:
 - Robo de identidad (o sea, de la dirección MAC del dispositivo)
 - Man-in-the-middle
 - Denegación de servicio
 - Inyección en la red
 -

Consideraciones generales

- Las **medidas de seguridad** se pueden clasificar de acuerdo a:
 - la transmisión inalámbrica
 - los puntos de acceso (AP)
 - los elementos de interconexión (ej., routers)
- Medidas de seguridad relacionadas con las **transmisiones inalámbricas**:
 - las principales amenazas son:
 - la copia de mensajes – *sniffing / eavesdropping*
 - la alteración – *tampering / manipulación*
 - la inserción de mensajes – *generación / falsificación*
 - la interrupción de los mismos – *denegación de servicio*
 - las contramedidas son:
 - técnicas de ocultación o engaño (ej. aleatorización en el routing)
 - cifrado (ej. AES-CBC)
 - autenticación (ej. MAC)
 - métodos contra DoS, ej. uso de nonce ☺



Consideraciones generales

- Las **medidas de seguridad** se pueden clasificar de acuerdo a:
 - la transmisión inalámbrica
 - **el puntos de acceso (AP)**
 - los elementos de interconexión (ej., routers)
- Medidas de seguridad relacionadas con el **AP**:
 - la mayor amenaza que involucra al AP es el acceso no autorizado a la red
 - la forma de evitarlo es usando **mecanismos de autenticación** para los dispositivos que se quieren conectar a la red



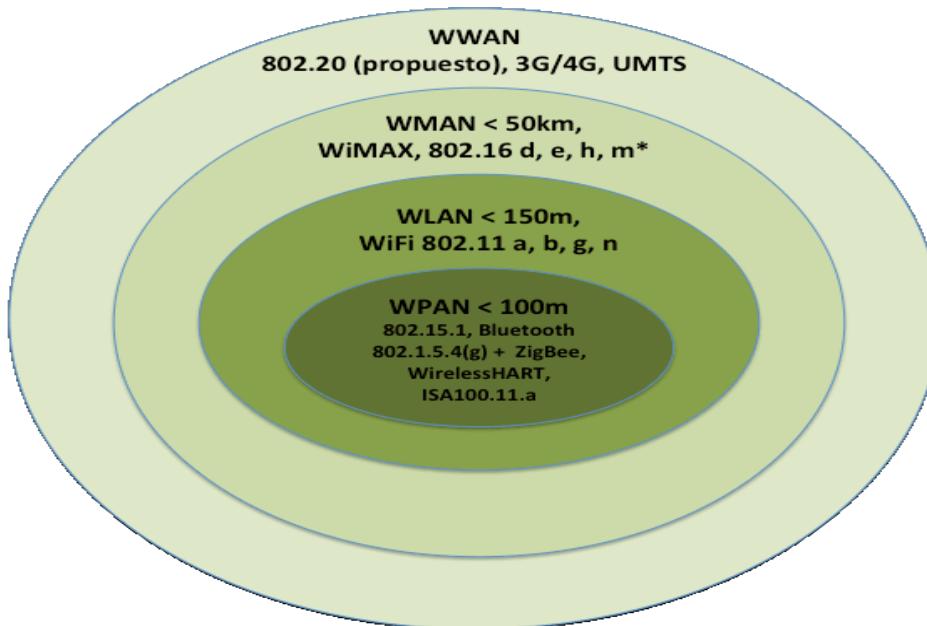
Consideraciones generales

- Las **medidas de seguridad** se pueden clasificar de acuerdo a:
 - la transmisión inalámbrica
 - el puntos de acceso (AP)
 - **los elementos de interconexión (ej., routers)**
- Medidas de seguridad relacionadas con **los elementos de la red**:
 - **cifrado**
 - integrado en los routers inalámbricos para el tráfico entre routers
 - **deshabilitar el broadcast de identificación**
 - sólo los dispositivos autorizados podrán conocer la identidad de los routers
 - **dejar que sólo equipos específicos se conecten** a la red
 - sólo direcciones cuyas direcciones MAC sean conocidas
 - **cambiar el identificador (el SID)**
 - sobre todo el por defecto que el router trae de fábrica
 - **cambiar el password**
 - El por defecto o el preestablecido para la administración del router
 - Hacer frecuente *rekeying*



Redes inalámbricas IEEE 802.11

- IEEE 802 es un comité que ha desarrollado estándares para diferentes tipos de redes LAN y WAN
- IEEE 802.11 es un capítulo de ese comité, y su objetivo es el desarrollo de un protocolo y las especificaciones de transmisión para LANs inalámbricas



Redes inalámbricas IEEE 802.11

- Diferencias entre una red cableada y una red inalámbrica:

	Cableada	Inalámbrica
Ventajas	Robustez Ancho de banda Equipos baratos Fiables al contexto (ruido, vibración, interferencias u obstáculos)	Rapidez y bajo coste de instalación Bajo coste de mantenimiento Movilidad Conexión para múltiples usuarios/dispositivos
Inconvenientes	Coste de mantenimiento Vulnerabilidad a amenazas físicas (principalmente) Dificultad para control en local	Vulnerables a múltiples tipos de amenazas No fiable para contextos inestables, ruidosos o con altas interferencias Coexistencia para interactuar varias redes inalámbricas

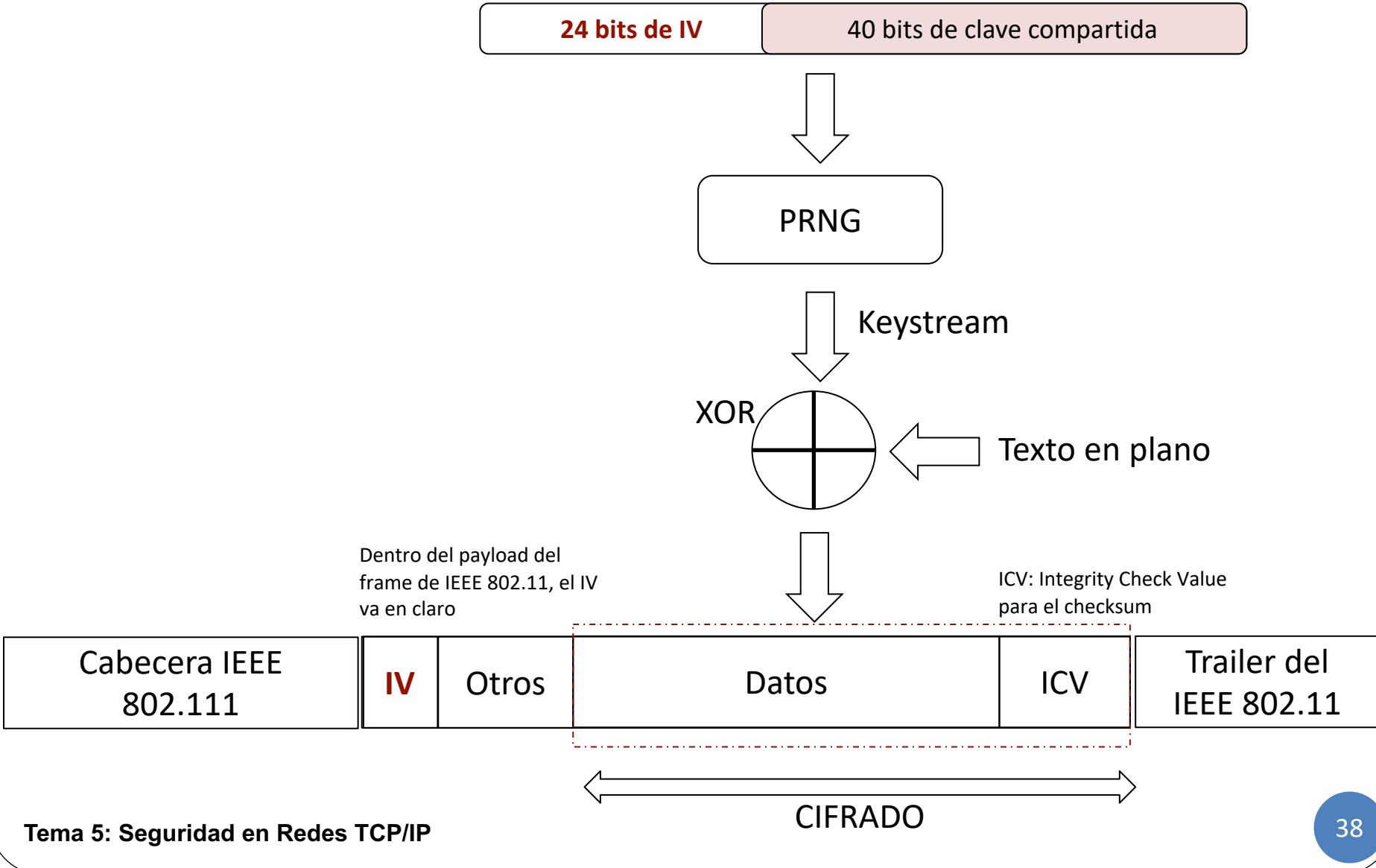
Redes inalámbricas IEEE 802.11

- En lo que a **seguridad** se refiere, hay dos características importantes que distinguen las LAN cableadas y las inalámbricas:
 - en una LAN cableada hay una forma intrínseca de **autenticación** de las estaciones porque están directamente interconectadas a esa red
 - una LAN cableada proporciona un cierto grado de **confidencialidad de los datos** porque la recepción de los datos está limitada a las estaciones conectadas a esa red
- Estas diferencias ponen de manifiesto la necesidad de servicios y mecanismos de seguridad más robustos en las LAN inalámbricas
 - La especificación original de IEEE 802.11 requería, por tanto, un conjunto de características de seguridad:
 - **Privacidad de los datos (confidencialidad) y autenticación**, por lo que se definió el protocolo **WEP (Wired Equivalent Privacy)**

IEEE 802.11: WEP (Wired Equivalent Privacy)

- El protocolo WEP se especificó con el objetivo de proporcionar unos **niveles de seguridad y confidencialidad comparables al de las LAN cableadas**
 - limitado a la comunicación entre la estación y el punto de acceso
- Se basa:
 - en la especificación de una clave de 64 bits que comparten los dispositivos de la red, y de esos 64 bits:
 - 40 corresponden a la clave secreta, y
 - **24 bits para el vector de inicialización (IV)**
 - Además, WEP se basa en el algoritmo de cifrado RC4 (algoritmo en flujo)
- WEP tiene varias vulnerabilidades:
 - **uso de IV débiles**, que posibilitan que, a partir de un número de paquetes cifrados, recuperar la clave secreta
 - Debido a la reutilización de IVs y a la corta longitud del IV

Redes inalámbricas IEEE 802.11



IEEE 802.11i: WPA (Wi-Fi Protected Access)

- Ante esa debilidad, IEEE introdujo una serie de mejoras recogidas e integradas dentro de la versión **IEEE 802.11i**
 - y, por tanto, dentro del protocolo **WPA (Wi-Fi Protected Access)**
- Las mejoras de WPA son:
 - El protocolo **TKIP (Temporal Key Integrity Protocol)**
 - Inicialmente se basaba de RC4, pero usando IVs de 48 bits (el doble de los 24 bits del IV de WEP)
 - Como la mejora de TIK seguía proporcionando los mismos problemas de seguridad, se propuso más tarde el uso del **algoritmo de cifrado AES**
 - También propuso la adopción del protocolo de autenticación **802.1X** (desarrollado inicialmente para LAN cableadas), conocido como **EAP (Extensible Authentication Protocol)**

IEEE 802.11i: WPA (Wi-Fi Protected Access)

- 802.11i asegura tres principales servicios de seguridad:
 - **Autenticación**
 - Autenticación entre un cliente (la estación) y el servidor de autenticación (AS)
 - El AS proporciona autenticación mutua y generación de claves temporales para la confidencialidad e integridad
 - Las claves temporales se usan entre el cliente y el AP
 - **Control de acceso - basado en “puertos”**
 - Controla el acceso de acuerdo a puertos
 - *“Hasta que el cliente no se autentique y no tenga las claves, no podrá acceder al sistema”*, por lo que el AP bloquea todos los puertos excepto el requerido para el IEEE 802.1X / EAP
 - **Confidencialidad con integridad de mensaje**
 - Una vez autenticado el cliente y generados las claves, como se indica arriba, el cliente y el AP pueden intercambiar datos de manera segura
 - Los datos se cifran con TKIP / AES y aplica una función MAC que asegura que los datos no han sido alterados

IEEE 802.11i: WPA (Wi-Fi Protected Access)

- Concretamente:
 - Autenticación
 - A través de un servidor de autenticación en el AP o en un servicio externo, ej. RADIUS
 - Aplica el protocolo EAP para la autenticación
 - MAC/MIC (Message Integrity Code) usando HMAC con SHA-1 o MD5, TKIP-MIC, etc.
 - Control de acceso - basado en “puertos”
 - Controla el acceso de acuerdo a puertos
 - Confidencialidad con integridad de mensaje
 - Aplica TKIP-RC4, CCM (AES-CTR)
 - Generación de clave
 - Aplica HMAC-SHA-1

IEEE 802.11i: WPA (Wi-Fi Protected Access)

- Las operaciones de seguridad (autenticación, control de acceso y confidencialidad) en el 802.11i se distribuyen entre 5 fases distintas:

1. Descubrimiento

- el AP utiliza mensajes llamados *beacons* y *probe responses* para anunciar su política de seguridad
- la estación los utiliza para identificar al AP y se asocia con él seleccionando un *cipher suite* y un mecanismo de autenticación

2. Autenticación

- la estación y el servidor de autenticación (AS) verifican la identidad del otro
- hasta que la autenticación no ha finalizado, el AP bloquea cualquier tráfico entre la estación y el AS, salvo que el tráfico esté relacionado con el propio proceso de autenticación

IEEE 802.11i: WPA (Wi-Fi Protected Access)

3. Generación y distribución de claves

- el AS y la estación realizan diferentes operaciones para generar claves criptográficas, las cuales se almacenan en el AP y la estación
- Esto quiere decir que la comunicación segura se realiza entre el AP y la estación:
 - Estación → AP → Estación

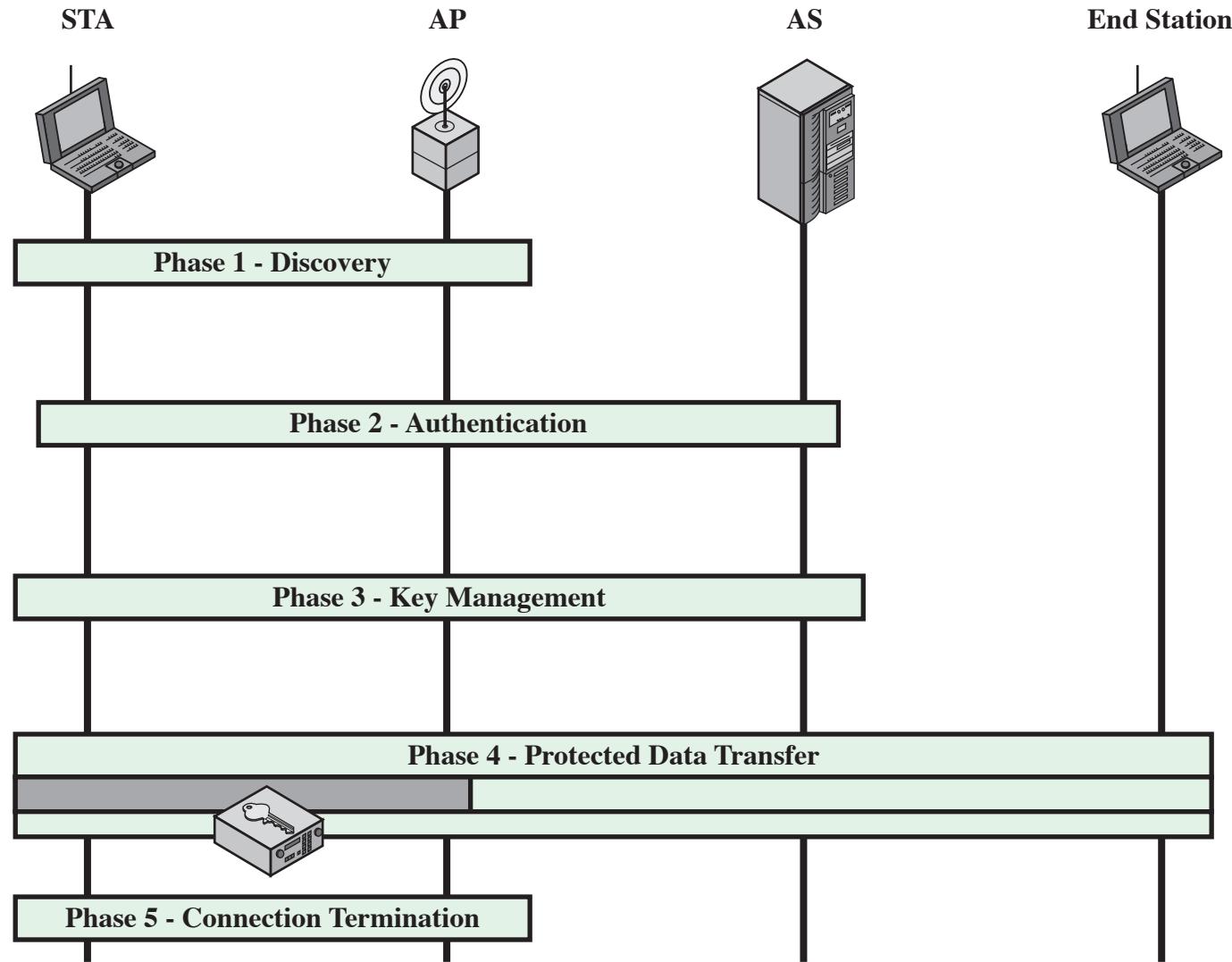
4. Transferencia segura de datos

- Como se indica arriba, la estación origen y la estación final intercambian datos a través del AP, pero la transferencia sólo se realiza de forma segura (cifrada) entre la estación de origen y el AP
 - Estación → AP → Estación

5. Finalización de la conexión

- Entre la estación y el AP

IEEE 802.11i: WPA (Wi-Fi Protected Access)



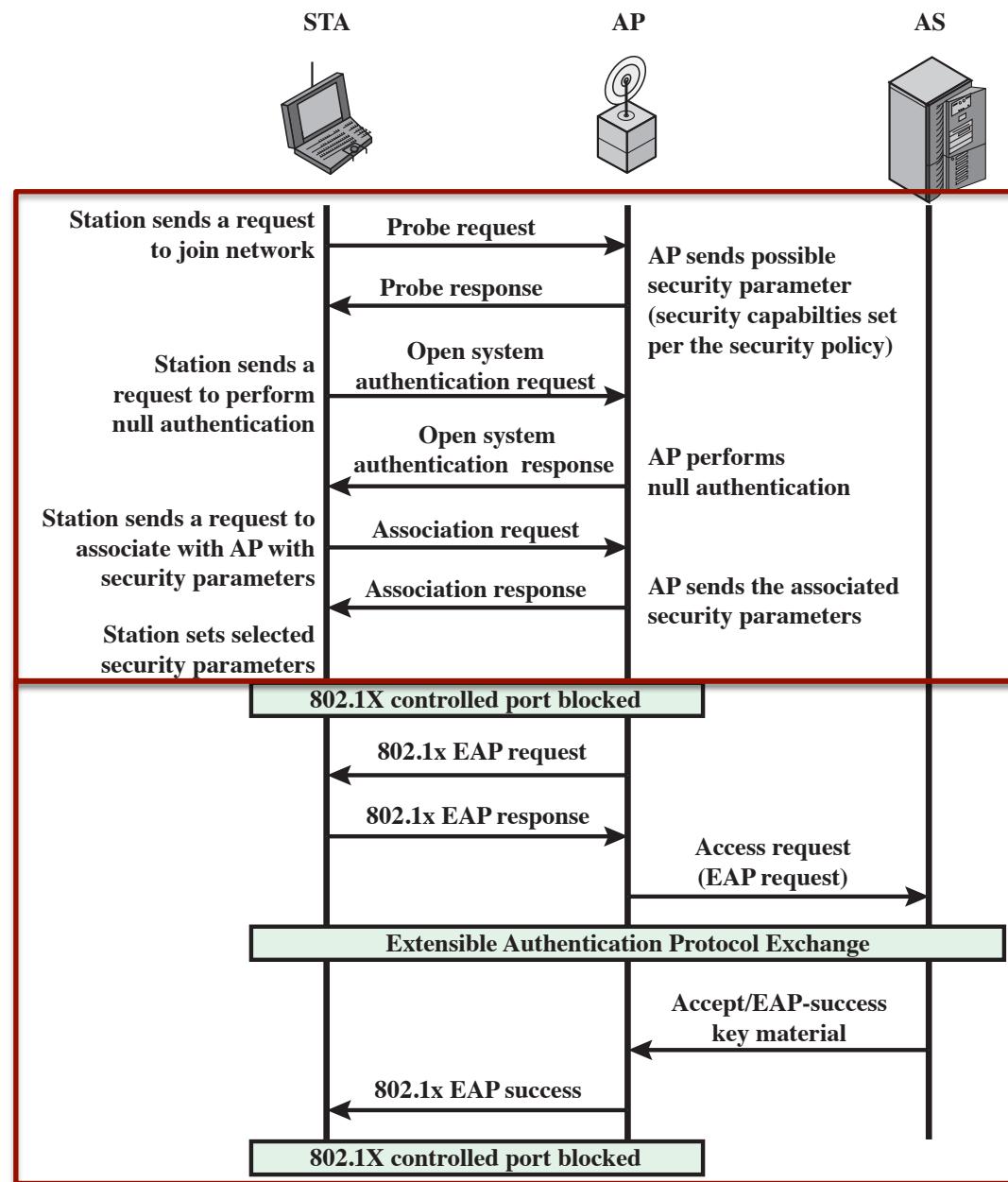
IEEE 802.11i: WPA (Wi-Fi Protected Access)

- Detalles de la **fase de autenticación**:
 - permite la autenticación mutua entre la estación y un servidor de autenticación (AS), como por ejemplo: un servidor RADIUS
 - la autenticación está diseñada para permitir sólo a estaciones autorizadas el uso de la red y garantizarles que están comunicando con una red legítima
 - el protocolo de autenticación utilizado es el **EAP (Extensible Authentication Protocol)**, definido en el estándar 802.1X
 - este estándar define los términos: *suplicant* (el cliente), *authenticator* (el AP) y *authentication server* (el AS)
 - el *authentication server* puede ser un servidor por sí mismo, o puede ejecutarse dentro del propio *authenticator* (el AP)

IEEE 802.1X / EAP

Descubrimiento

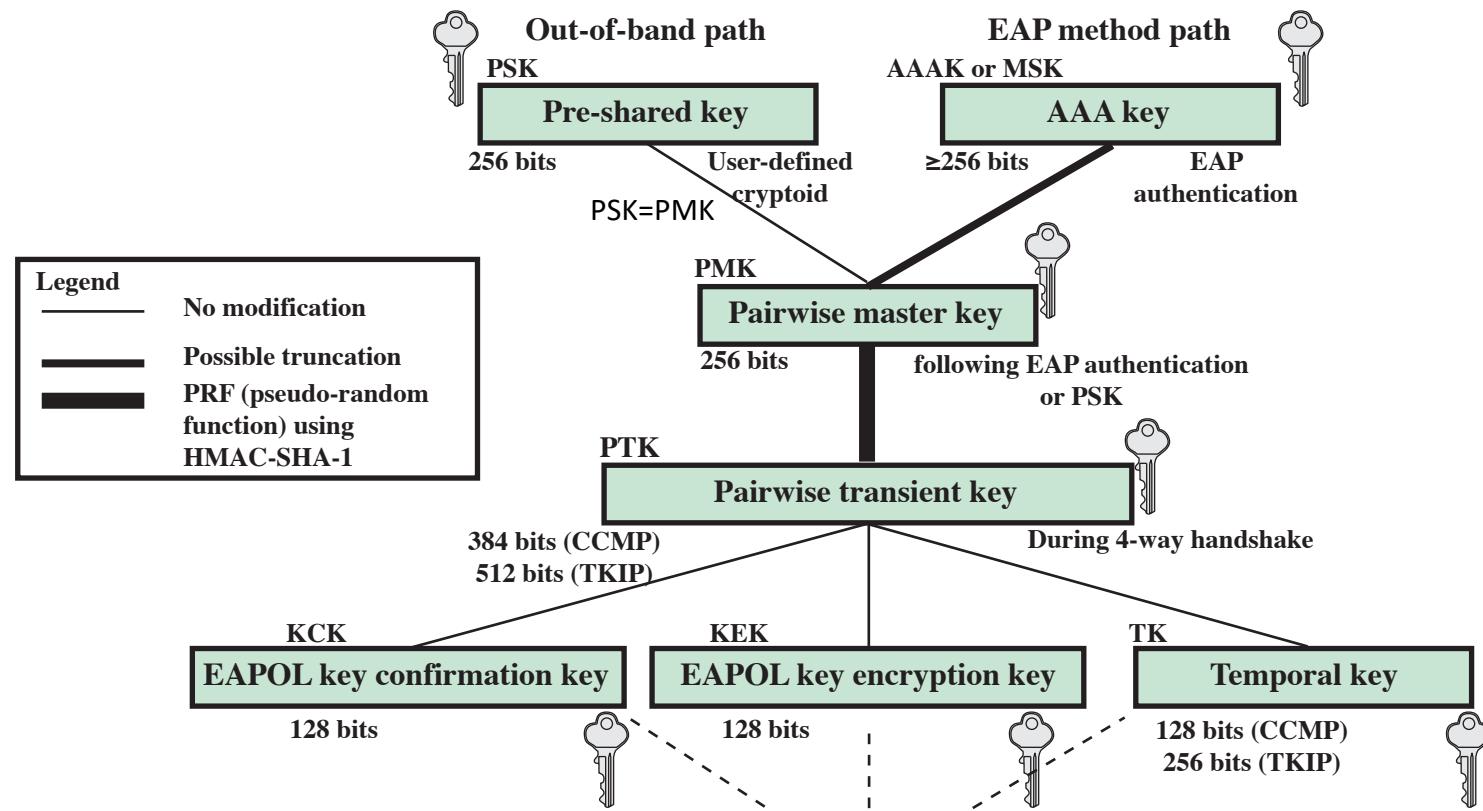
Autenticación con el AS



IEEE 802.11i Phases of Operation:
Capability Discovery, Authentication, and Association

IEEE 802.11i: WPA (Wi-Fi Protected Access)

- Detalles de la fase de generación de claves:



$$\text{CCMP} = 128 (\text{KCK}) + 128 (\text{KEK}) + 128 (\text{TK})$$

$$\text{TKIP} = 128 (\text{KCK}) + 128 (\text{KEK}) + 256 (\text{TK})$$

IEEE 802.11i: WPA (Wi-Fi Protected Access)

- Como se aprecia de la figura anterior, la **fase de generación de claves** se basa en la composición global de varias claves organizadas según una jerarquía
- Es decir, se crean claves temporales de sesión:
 - La clave **PMK** (Pairwise Master Key), la cual depende del método de autenticación
 - Si se aplica una PSK (Pre-Shared Key), entonces: PMK = PSK, cuyo valor puede derivar de una frase secreta de 8-23 caracteres, o una cadena de 256-bit
 - Solución adecuada para entornos pequeños sin servidor configurado
 - Si se usa un servidor AS, entonces PMK puede derivar de la MK (Master Key) del SA
 - Con la PMK se genera una clave temporal para cifrado denominada **PTK** (Pairwise Transient Key).
 - Su longitud depende del protocolo de cifrado (TKIP-512, CCMP-384)
 - La PTK está basado de varias claves dedicadas:
 - KCK (Key Confirmation Key – 128 bits): usada para la **autenticación de mensajes (MIC)**
 - KEK (Key Encryption Key – 128 bits): usada para garantizar la **confidencialidad de los datos**
 - TK (Temporary Key – 256/128 bits): clave para realizar otras operaciones de cifrado datos en modo TKIP o CCMP