uc3m | Universidad Carlos III de Madrid
Departamento de Ingeniería Telemática

SEGURIDAD EN REDES DE COMUNICACIONES

Celeste Campo (celeste@it.uc3m.es)
Carlos García Rubio (cgr@it.uc3m.es)

ÍNDICE

BIBLIOGRAFÍA

- Básica:
  - Capítulo 8 de A.S. Tanenbaum: "Computer Networks", 5 Ed., Prentice Hall, 2011.
  - Capítulo 8 de J.F. Kurose, K.W Ross: “Computer Networking: A Top-Down Approach”, 6 Ed., Pearson 2013.

INTRODUCCIÓN

- Inicialmente Internet red académica
  - poca preocupación por la seguridad
- Ahora se utiliza para:
  - transacciones económicas
    - operaciones bancarias, comercio electrónico, etc...
  - red empresarial
  - red gubernamental

⇒ cobra importancia la **seguridad**

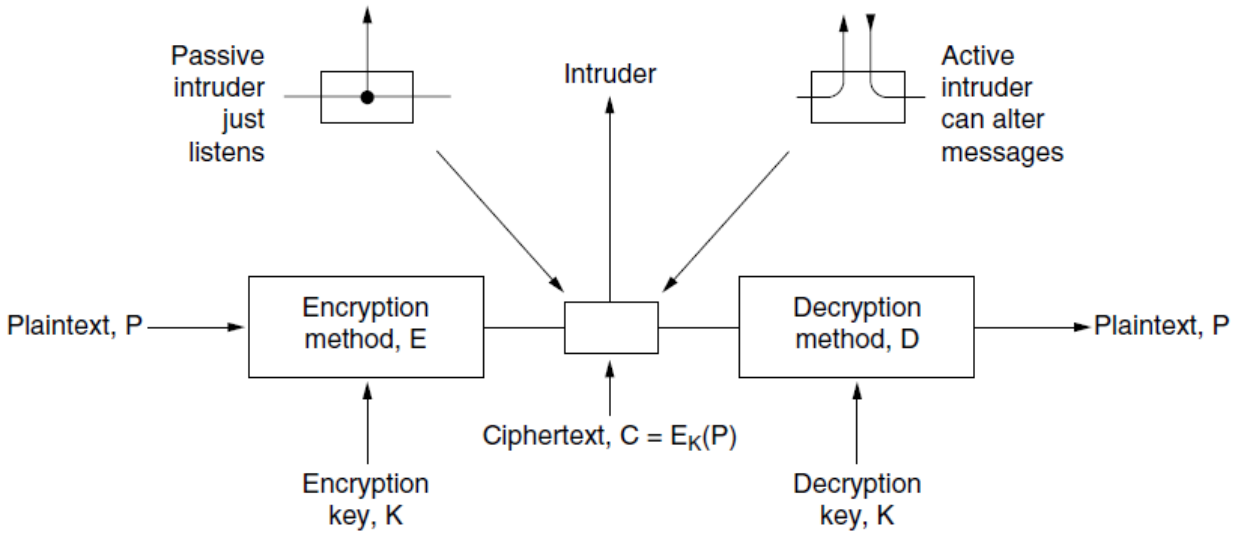
- Historia de las técnicas de cifrado:
  - muy ligada a la comunicación entre militares
    - numerosos avances durante la 2ª GM y guerra fría
    - considerado "munición" en muchos países
  - se ha extendido al dominio civil
  - empieza a ser utilizado en sistemas telemáticos

CONCEPTOS BÁSICOS

- Dos tipos de amenaza a la seguridad:
  - Desvelar información
  - Modificar información

⇒ Dos **tipos de intruso**:

  - pasivo (escucha)
  - activo



Seguridad en redes de comunicaciones

- Implica cuatro áreas:
  - **Secreto**: que la información sólo pueda ser recibida por usuarios autorizados.
  - **Autenticación** (compulsa): determinar con quién está uno hablando antes de revelar información clasificada.
  - **Integridad** de las transmisiones: asegurar que lo que se transmite y lo que se recibe es lo mismo.
  - Protección contra **incumplimientos** de contratos concertados electrónicamente.

¿A qué nivel de la torre OSI debe ir la seguridad?

- A todos los niveles:
  - Físico: cables protegidos con gas.
  - Enlace: en cada enlace punto a punto entre sistemas.
  - Red: *firewalls*.
  - Transporte: cifrado de conexiones.
- Pero si se quiere algo más que secreto, hay que hacerlo a **nivel de aplicación**.

DEFINICIONES

- **Cifrado**: desfiguración de la información que se transmite para hacerla irreconocible al observador intermedio.
- **Autenticación**: añadir información de control protegida a los mensajes para evitar que sea alterada o para asegurar la identidad del remitente.
- Principio de *Kerckhoff*: los algoritmos de cifrado deben ser públicos, las claves secretas
- **$C = E_K(P)$**
  - Texto original (*plaintext*) = P
  - Función de cifrado = E
    - Se supone conocida de todos.
  - Clave = K parámetro de la función de cifrado
    - es lo que es secreto (... o no)
    - es importante su tamaño
      - más trabajo para descifrarla
  - Texto cifrado = C

Definiciones

- **Criptografía**: arte de idear claves
- **Criptoanálisis**: arte de “romper” (averiguar) claves.
- **Criptología** = criptografía + criptoanálisis

Escenarios de criptoanálisis

- Cuando un intruso intenta romper una clave pueden darse tres situaciones:
  - que sólo tenga texto cifrado
  - que conozca algunas parejas de texto original - texto cifrado
  - que pueda obtener texto cifrado correspondiente al texto original que desee
- Podríamos suponer la primera
- Sin embargo es más común la segunda
- Lo conservador es suponer la tercera situación
- **Seguridad del cifrado**:
  - seguridad incondicional
    - Sólo es posible con sistema Venam (de adición de cintas): clave de misma longitud que mensaje, aleatoria y de un uso
  - seguridad computacional
    - conceptualmente descifrable, pero se necesitaría capacidad de proceso > que todos los procesadores del universo

Métodos clásicos de cifrado

- Cifrado por sustitución:
  - Cada letra se sustituye por otra.
    - Cifrado César:
      - trasladar tres letras el abecedario: a por D, b por E, c por F, ...
      - generalizando, trasladar k letras
  - En general, cualquier sustitución de alfabeto (letra por letra)
    - $26! = 4 \times 10^{26}$
      - a 1 μs por clave, 10^{13} años para probarlas
    - Pero no es tan bueno. Se puede atacar:
      - basándose en propiedades estadísticas del idioma
        - letras más frecuentes
        - grupos de dos letras
        - grupos de tres letras
      - buscando palabras probables en el texto con letras repetidas…

Métodos clásicos de cifrado (cont.)

- Cifrado por transposición:
  - Consiste en reordenar las letras

M	E	G	A	B	U	C	K
7	4	5	1	2	8	3	6
p	l	e	a	s	e	t	r
a	n	s	f	e	r	o	n
e	m	i	l	l	i	o	n
d	o	l	l	a	r	s	t
o	m	y	s	w	i	s	s
b	a	n	k	a	c	c	o
u	n	t	s	i	x	t	w
o	t	w	o	a	b	c	d

Plaintext

pleasetransferonemilliondollarsto
myswissbankaccountsixtwotwo

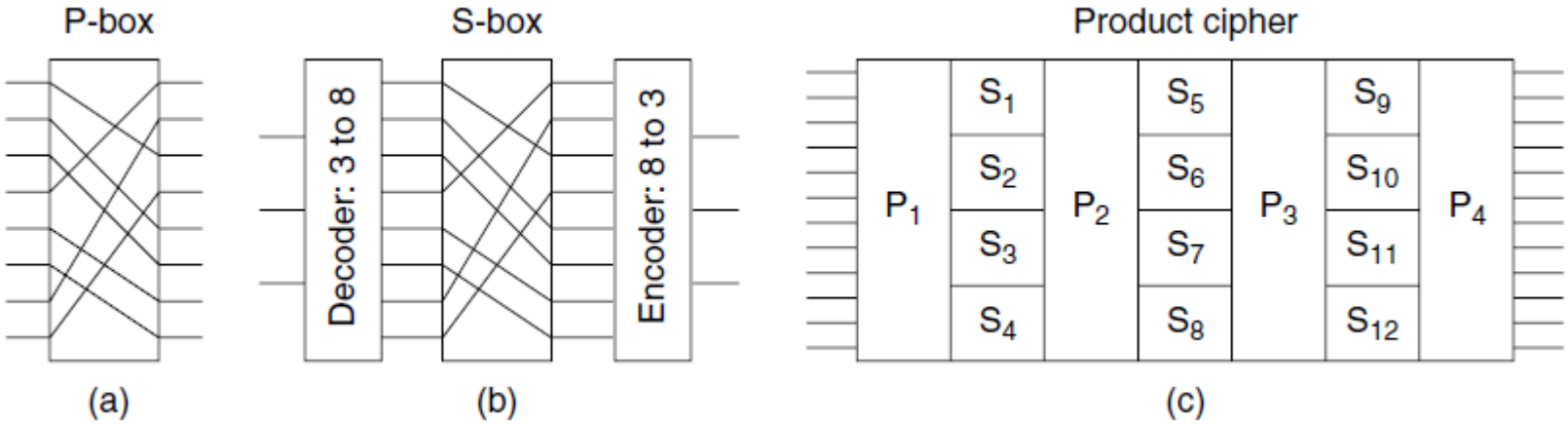
Ciphertext

AFLLSKSOSELAWAIATOOSSCTCLNMOMANT
ESILYNTWRNNTSOWDPAEDOBUOERIRICXB

  - Es posible atacarlo
    - sabiendo que es cifrado por transposición
    - conociendo alguna palabra larga
    - observando parejas de letras posibles

Métodos clásicos de cifrado (cont.)

- Cifrado producto:
  - Consiste en combinar sustituciones (S-box) y transposiciones (P-box, de permutación) en etapas sucesivas.
  - Más robusto.



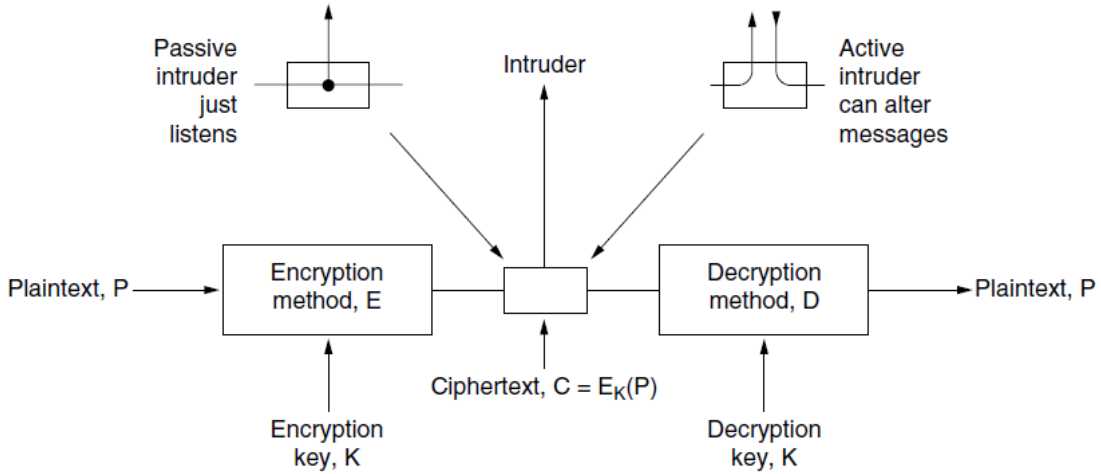
  - El objetivo es hacer el algoritmo de cifrado tan complicado que incluso con cantidades enormes de texto cifrado sea imposible descifrarlo.

Dos principios básicos de la criptografía

- Los mensajes originales deben contener **redundancia:**
  - si no, cualquier texto cifrado que introduzca un intruso activo dará lugar a un texto descifrado correcto.
  - aunque no demasiada redundancia, porque si no puede facilitarse el criptoanálisis.
- Debe evitarse que la repetición de un mensaje anterior sea un mensaje válido
  - **sello de tiempo**, guardar los mensajes durante un tiempo para comprobar duplicados

ALGORITMOS DE CLAVE SIMÉTRICA

- Usan la misma clave para cifrar y descifrar.



- Los más importantes son sistemas de cifrado producto que trabajan en bloques.
  - Toman un bloque de n bits de texto plano como entrada.
  - Lo transforman mediante el uso de la clave en un bloque de n bits de texto cifrado.
  - Por lo general, n puede valer de 64 a 256.

CIFRADO DES

- Finales -60: necesidad de norma de cifrado para intercomunicación de los sistemas informáticos de las administraciones civiles del gobierno norteamericano.
- Requisitos:
  - bien conocido (de dominio público)
  - económico
  - pequeño
- 1975: petición de propuestas
- Se elige DES (*Data Encryption Standard*)
  - desarrollada por IBM
  - basado en Lucifer (experiencia previa)
- Aprobado a mediados de 1977

DES: Funcionamiento

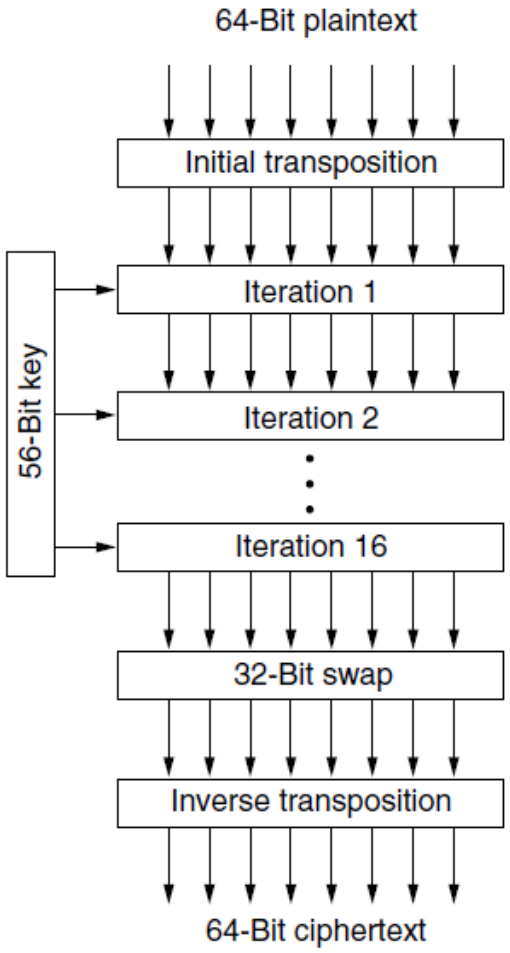
- Divide el texto en bloques de 64 bits.
- Varios modos de trabajo
  - cifrar cada bloque por separado, encadenadamente, ...
- La clave es también de 64 bits
  - aunque sólo se usan los 56 de menos peso
    - los otros 8 de paridad

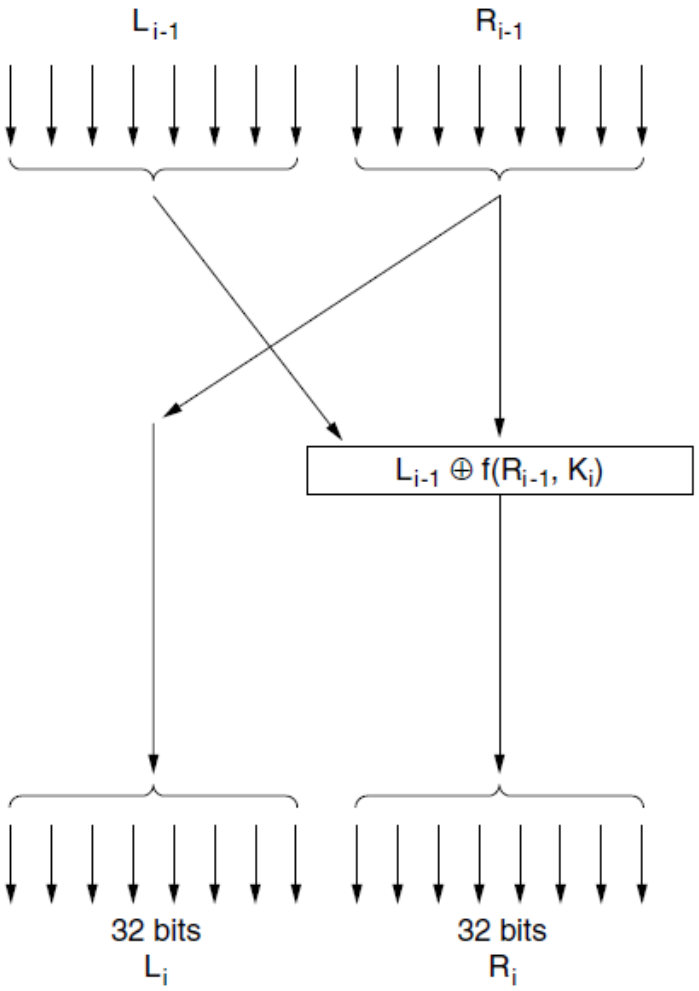
  $\Rightarrow 2^{56} = 10^{17}$ posibles claves
- Algoritmo de cifrado = serie de transposiciones y sustituciones en función de la clave de cifrado.
- Se descifra con la misma clave y algoritmo inverso
- Se utiliza la misma clave para descifrar
  - debe permanecer secreta
    - es un algoritmo de **clave secreta**
- El algoritmo se ha generalizado mucho
  - ha sido implementado en circuito integrado

DES: Algoritmo de cifrado

- Transposición inicial independiente de clave
  - se desconoce su propósito
- A la clave se le aplica una transposición
- 16 etapas intermedias idénticas
  - se obtiene clave 48 bits
    - dividiendo la clave en dos bloques de 28
    - se rota cada bloque un nº de bits (0, 1 ó 2) dependiente de la etapa en que se esté
    - se eligen 48 de los 56 bits (según etapa)
  - 32 bits izda salida = 32 bits izda entrada
  - 32 bits dcha salida =
    - se calcula función dependiente de clave 48 bits
      - a partir de los 32 se obtienen 48 bits
        - transposición + duplicación
      - OR exclusivo con la clave
      - dividido en 8 grupos de 6 bits, a cada uno se le aplica una sustitución de 4 bits
      - cada grupo 4 bits una transposición
    - salida = OR exclusivo de 32 bit izda y resultado de la función
- Transposición inversa final

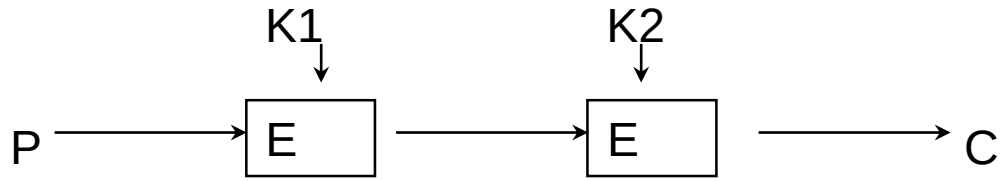
DES: Algoritmo de cifrado (cont.)



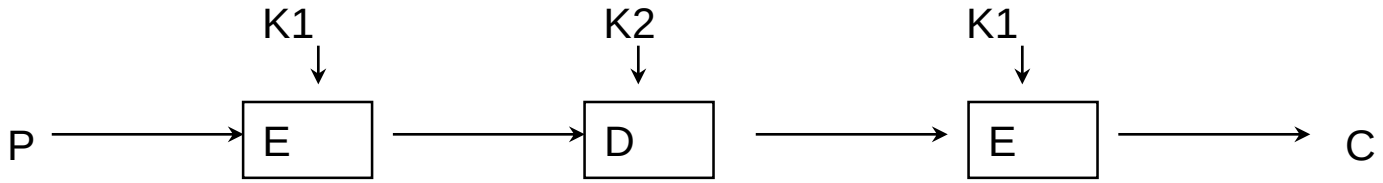


DES: Ruptura del cifrado

- La agencia de seguridad americana (NSA) “invitó“ a IBM a reducir la clave de 128 bits (Lucifer) a 56 bits
  - 2^{56} es poco
- Soluciones:
  - Cifrar dos veces
    - fácil de romper (*meet-in-the middle)*



  - Triple DES
    - sólo dos claves (no tres), da suficiente espacio de claves
    - EDE (no EEE) para compatibilidad
    - ¡Es bastante seguro!



CIFRADO AES

- En 1997 el NIST (*Nacional Institute of Standards and Technology*) invitó a investigadores de todo el mundo a enviar propuestas para un nuevo estándar de cifrado.
  - Se llamaría AES (del inglés *Advanced Encryption Standard*)
- Requisitos:
  - Debía ser un sistema de cifrado de bloques simétrico.
  - Diseño público.
  - Soportar longitudes de claves de 128, 192 y 256 bits.
  - Debe ser posible implementarlo en HW y SW.
  - El algoritmo debe ser público.
- De 15 propuestas se eligió en 2001 la llamada *Rijndael*, de Joan Daemen y Vincent Rijmen (belgas).

AES: Funcionamiento

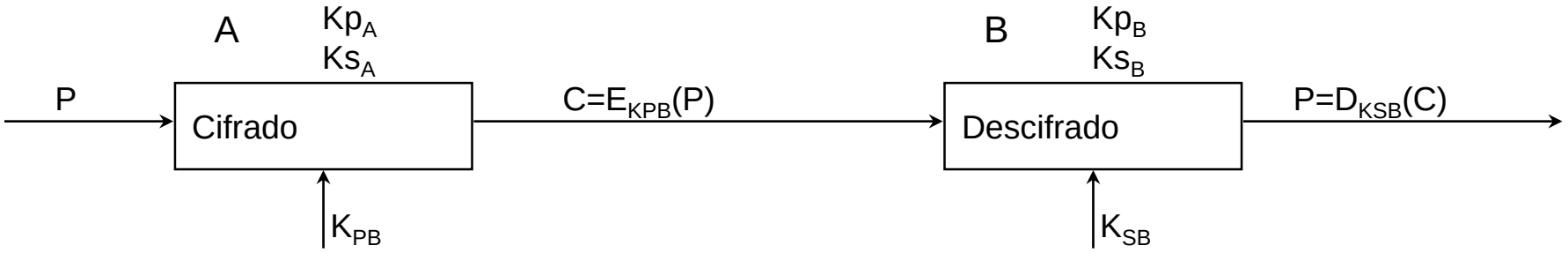
- Al igual que el DES, Rijndael utiliza sustituciones y permutaciones en múltiples rondas (entre 10 y 14 dependiendo del tamaño de clave y de bloque).
- Soporta longitudes de clave y tamaños de bloque de 128 a 256 bits.
  - Lo más habitual, bloque de 128 bits con clave de 128 bits y bloque de 128 bits con clave de 256 bits.
  - 128 bits da una seguridad suficiente: da un espacio de claves de $2^{128} \approx 3 \times 10^{38}$ claves.
- Desde una perspectiva matemática, Rijndael se basa en la teoría de campos de Galois.
- El algoritmo se ha diseñado para gran seguridad y velocidad.
  - Implementación SW en una máquina de 2 GHz alcanza tasa de encriptación de 700 Mbps (en HW aún más rápido).

Otros algoritmos de clave simétrica

Sistema de cifrado	Autor	Longitud de clave	Comentarios
DES	IBM	56 bits	Muy débil para usarlo en la actualidad.
RC4	Ronald Rivest	1-2048 bits	Precaución: algunas claves son débiles.
RC5	Ronald Rivest	128-256 bits	Bueno, pero patentado.
AES (Rijndael)	Daemen y Rijmen	128-256 bits	La mejor opción.
Serpent	Anderson, Biham, Knudsen	128-256 bits	Muy sólido.
Triple DES	IBM	168 bits	Bueno, pero se está volviendo anticuado.
Twofish	Bruce Schneier	128-256 bits	Muy sólido; se utiliza mucho.

ALGORITMOS DE CLAVE PÚBLICA

- Punto débil de los sistemas tradicionales de cifrado: **mantener secreta la clave**.
  - debe ser conocida por el destino
  - pero no por los intrusos
- 1976: Diffie y Hellman (Standford) proponen el uso de dos claves:
  - clave de cifrado (Kp)
    - pública
    - $C=E_{Kp}(P)$
  - clave de descifrado (Ks)
    - privada (secreta)
    - muy difícil (imposible) de obtener a partir de la clave de cifrado
    - $P=D_{Ks}(C)$



ALGORITMO RSA

- Desarrollado en 1978 en el MIT por Rivest, Shamir y Adleman
- Algoritmo:
  - Elegir p, q primos (> 10^100)
  - n= p x q
  - z= (p-1) x (q-1)
  - Elegir d primo con respecto a z
  - Encontrar un número e | (e x d) mod z = 1
  - Kp = (e, n)
  - Ks = (d, n)
- Algoritmo de cifrado:
  - P (en binario) tiene que estar en [0,n)

  $\Rightarrow$ hay que dividir el texto en bloques de k bits, con $2^k < n$
  - $C = (P^e)$ MOD n
- Algoritmo de descifrado:
  - $P = (C^d)$ MOD n

RSA: Ejemplo

- p, q primos
- n= p x q
- z= (p-1) x (q-1)
- Elegir d primo con respecto a z
- Encontrar un número e | (e x d) mod z = 1

- Kp = (e, n)
- Ks = (d, n)

- C = (P^e) MOD n
- P = (C^d) MOD n

- p=3, q= 11
- n= 33
- z= 20
- d= 7

- (e x 7) mod 20 = 1 $\Rightarrow$ e = 3

- Kp= (3, 33)
- Ks= (7, 33)

- C= P^3 mod 33
- P= C^7 mod 33

RSA: Ejemplo

Plaintext (P)			Ciphertext (C)		After decryption	
Symbolic	Numeric	P^3	P^3 (mod 33)	C^7	C^7 (mod 33)	Symbolic
S	19	6859	28	13492928512	19	S
U	21	9261	21	1801088541	21	U
Z	26	17576	20	1280000000	26	Z
A	01	1	1	1	01	A
N	14	2744	5	78125	14	N
N	14	2744	5	78125	14	N
E	05	125	26	8031810176	05	E
Sender's computation				Receiver's computation		

RSA: Consideraciones

- Interesa n grande $\Rightarrow$ p y q grandes:
  - cada bloque "P" podrá ser de más bits
  - n muy grandes son difíciles de factorizar
    - 100 dígitos -> 1 semana
    - 150 dígitos -> 1000 años
    - 200 dígitos -> 1 millón de años
- Para codificar sucesivos bloques podemos utilizar cualquier **modo de funcionamiento** de los vistos en DES.
  - En la práctica no suele ser necesario, pues se utiliza sólo para codificar pequeños bloques de información, debido a que:
- El algoritmo tiene gran **coste** computacional

RSA: Uso habitual

- RSA se utiliza al principio de la comunicación, para distribuir la clave de un algoritmo de clave secreta
  - AES, triple DES...
- A continuación se cifra con clave secreta
  - la clave secreta sólo se utilizará en esta sesión
  - Se dificulta criptoanálisis
    - Hay menos texto cifrado disponible
  - Menos daños si un intruso averigua la clave

FIRMA DIGITAL

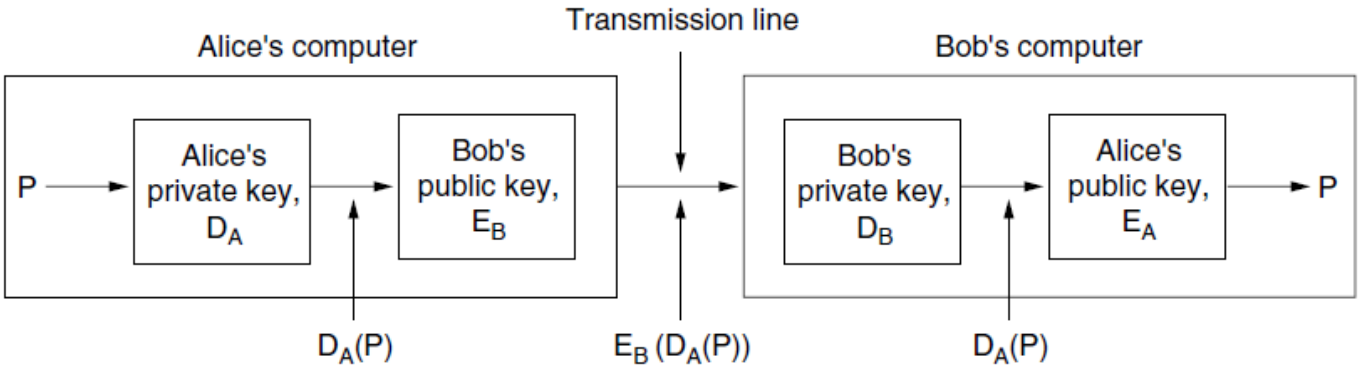
- Para tener validez legal muchos documentos necesitan firmarse
  - a mano
  - ¿cómo sustituirla firma en los mensajes electrónicos?

$\Rightarrow$ firma digital

- Requisitos de la firma digital:
  - que el receptor pueda verificar la identidad del remitente.
  - que el remitente no pueda repudiar posteriormente su mensaje.
  - que no se pueda falsificar (por el receptor)

Firma digital con clave pública

- Supongamos un cifrado de clave pública en el que E(D(P))=P
  - No siempre cierto
    - Sí es cierto siempre que D(E(P))=P
  - Sí lo cumple RSA, por ejemplo
    - hay otros (DSS: Digital Signature Standard)
- La firma digital con clave pública funciona:



- B tiene P y Da(P). Eso sólo es posible si se lo mandó A.

Firma digtal por cifrado de porciones del mensaje

- La firma digital que hemos visto cifre todo el mensaje
  - puede no ser necesario
  - ineficiente
- Basta con cifrar una porción del mensaje suficientemente larga
  - m bits
  - m a partir de 128 bits
  - coste del ataque $2^{m/2}$
    - ataque del cumpleaños
- Para obtener esa porción del mensaje se usan los algoritmos de resumen de mensaje (*message digest)*.

ALGORITMOS DE RESUMEN DE MENSAJE

- Se basan en una función de *hash* unidireccional , MD, que
  - Toma una parte arbitrariamente grande de texto plano.
  - A partir de ella calcula una cadena de bits de longitud fija
- Cuatro propiedades importantes de la función de *hash*:
  - Dado P, es fácil calcular MD (P).
  - Dado MD (P), es imposible encontrar P.
  - Dado P, no es posible encontrar P′ tal que MD (P′) = MD (P).
  - Un cambio en la entrada de incluso 1 bit produce una salida muy diferente.
- Para cumplir el tercer criterio, la longitud del resumen debe ser al menos 128 bits.

Algoritmos de resumen de mensaje

- Hay muchas formas de elegir ese subconjunto del mensaje.
  - MD5 (Rivest, 1992)
    - 128 bits
    - Se sigue usando aunque se le han encontrado debilidades.
  - SHA-1 (Secure Hash Algorithm)
    - 160 bits
    - más rápido
    - El código C completo de SHA-1 se proporciona en el RFC 3174.
  - SHA-2
    - produce hashes de 224, 256, 384 y 512 bits.

ADMINISTRACIÓN DE CLAVES PÚBLICAS

- Mediante la criptografía de clave pública, las personas que **no** comparten una clave común pueden de todos modos:
  - Comunicarse de forma segura.
  - Firmar mensajes, sin la presencia de un tercero de confianza.
  - Gracias a los resúmenes de mensajes firmados, verificar de una manera fácil y segura la integridad de los mensajes recibidos.
- El problema es cómo obtiene cada uno la clave pública del otro.
  - Se necesita un mecanismo para asegurar que las claves públicas se puedan intercambiar de manera segura.
  - La solución son las autoridades de certificación.

Certificados

- Una Autoridad de Certificación (CA, *Certification Authority*) es una organización que certifica claves públicas.
  - Certifica que una clave pública pertenece a una cierta persona, empresa u organización.
    - Emite certificados que enlazan una clave pública con el nombre de esa persona, empresa u organización.
  - La CA firma el hash SHA-1 del certificado con la clave privada de la CA
    - Cualquiera que conozca la clave pública de la CA puede comprobar su autenticidad.

Certificados X.509

- Ejemplo de certificado:

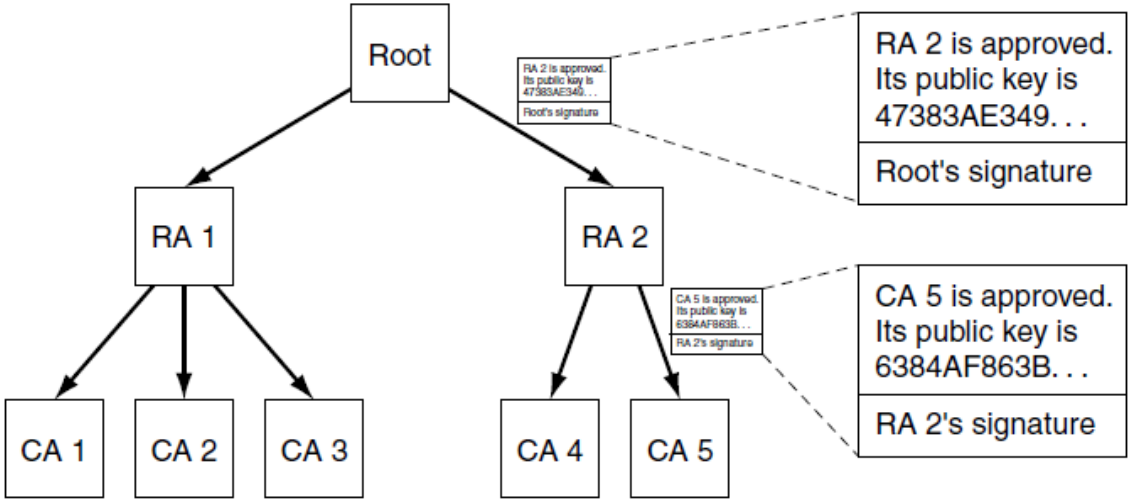
I hereby certify that the public key
19836A8B03030CF83737E3837837FC3s87092827262643FFA82710382828282A
belongs to
Robert John Smith
12345 University Avenue
Berkeley, CA 94702
Birthday: July 4, 1958
Email: bob@superdupernet.com

SHA-1 hash of the above certificate signed with the CA's private key

- El formato de certificado adoptado por el IETF es el estándar X.509 de ITU (RFC 2459, actualizada en 5280).
- Hay también certificados de atributos, que enlazan una clave pública no con una identidad sino con un atributo.
  - Por ejemplo, "Esta clave pública pertenece a alguien mayor de 18 años".
  - Útil por razones de privacidad,

Infraestructura de clave pública

- No es viable que una CA emita todos los certificados del mundo.
- Por esta razón se ha desarrollado una forma para certificar claves públicas: la Infraestructura de Clave Pública, (PKI del inglés *Public Key Infrastructure*).
- En la figura se muestra un ejemplo de dos niveles, pero pueden ser más o menos.



Infraestructura de clave pública

- En la figura anterior:
  - La CA de nivel superior (raíz) certifica a las autoridades CA de segundo nivel, a las que llamaremos autoridades regionales, RA.
    - Cuando la raíz autoriza una nueva RA, genera un certificado X.509 donde indica que ha aprobado la RA, e incluye en él la clave pública de la RA.
  - Las RA certifican a las CA finales.
    - Cuando una RA aprueba una nueva CA, produce y firma un certificado que indica su aprobación y que contiene la clave pública de la CA.
  - Las CA finales emiten los certificados X.509
- La cadena de certificados que va a la raíz se denomina cadena de confianza o ruta de certificación.

Infraestructura de clave pública

- Todos tienen que conocer la clave pública de la raíz.
- ¿Quién hace de CA raíz?
- La solución no es tener una sola raíz, sino muchas, cada una con sus propias autoridades RA y CA.
  - Los navegadores web modernos vienen precargados con claves públicas para cerca de 100 raíces, algunas veces llamadas anclas de confianza.
    - Queda a criterio del usuario confiar en que el fabricante del navegador elige bien los CA raíz
    - La mayoría de los navegadores permiten ver las claves de la raíz y eliminar las que nos parezcan sospechosas.
  - De esta forma, es posible evitar tener una sola autoridad mundial de confianza.

Revocación

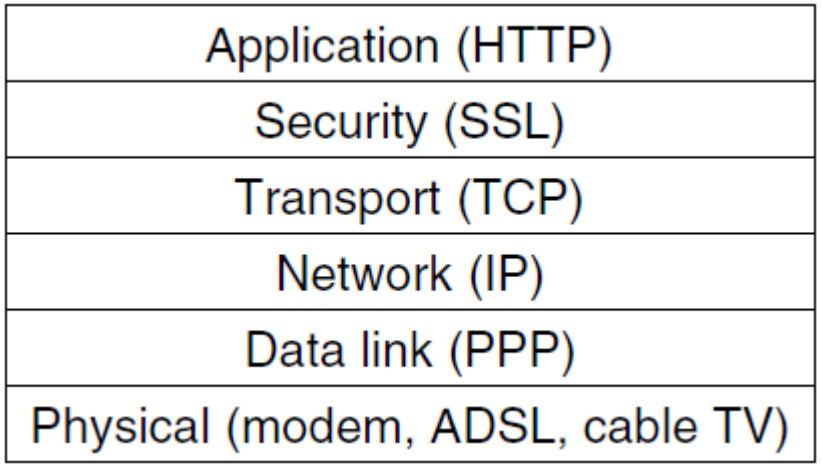
- El otorgante de un certificado podría decidir revocarlo por varios motivos:
  - Porque la persona u organización que lo posee ha abusado de él en cierta manera.
  - Porque la clave privada el sujeto se ha expuesto .
  - O porque la clave privada de la CA ha sido comprometida.
- PKI aborda este problema a través de las Listas de Revocación de Certificados (CRL).
  - Por un lado, los certificados tienen un tiempo de expiración.
  - Para los que no han expirado y se desea revocar, las CA emiten periódicamente una CRL con los números de serie de todos los certificados revocados.
- Esto obliga a que un usuario antes de usar un certificado debe obtener la CRL para ver si ha sido revocado.
  - Y debería volver a hacerlo cada vez que lo vaya a usar.

SEGURIDAD EN LA COMUNICACIÓN

- El IETF durante muchos años dejó de lado la seguridad en Internet.
- Después, agregarla no era fácil : ¿dónde colocarla?
  - La mayoría de los expertos en seguridad creían que tenía que llevarse a cabo extremo a extremo (en la capa de aplicación).
    - Problema: requiere cambiar todas las aplicaciones para que sean conscientes de la seguridad.
  - El siguiente enfoque es colocar el encriptado en la capa de transporte o en una nueva capa entre la capa de aplicación y la de transporte
    - SSL/TLS
    - Se conserva el enfoque de extremo a extremo pero no hay que cambiar las aplicaciones.

SSL/TLS

- SSL (*Secure Sockets Layer*) fue introducido en 1995 por Netscape
  - Entonces dominaba el mercado de los navegadores.
- Se trata de una nueva capa entre TCP y el nivel de aplicación que cifra la comunicación.



SSL/TLS

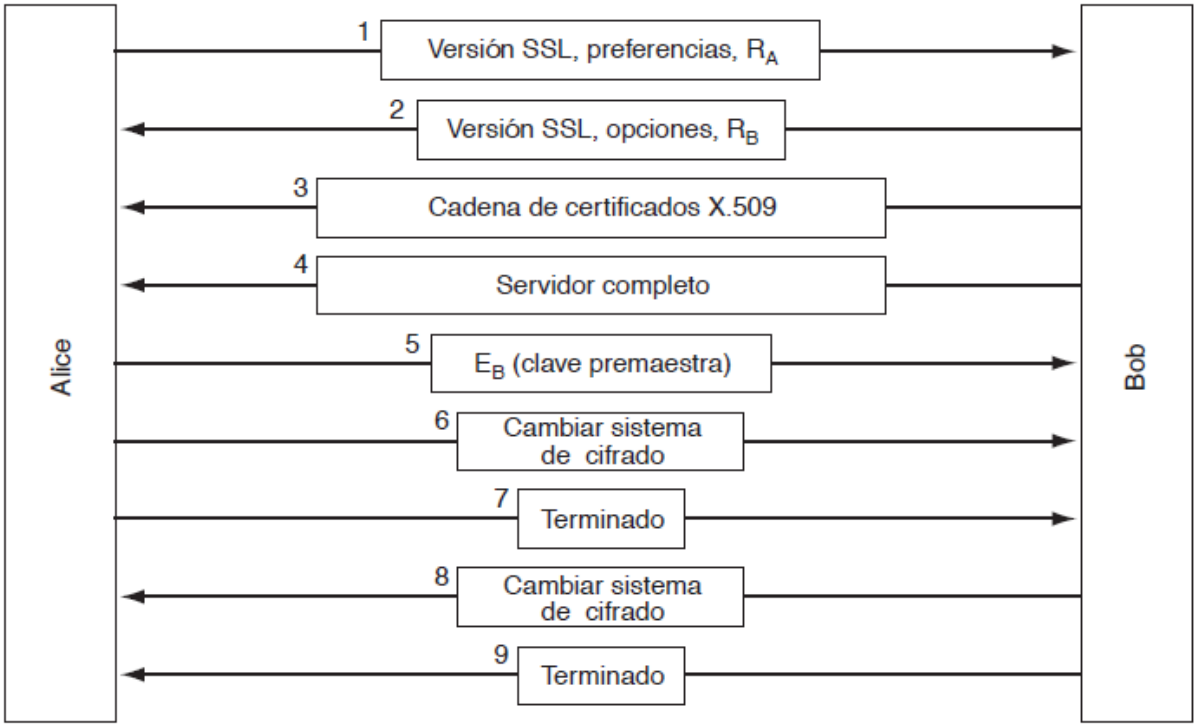
- En 1996, Netscape entregó el SSL a IETF para su estandarización. El resultado fue TLS (*Transport Layer Security*
- TLS se basa en la versión 3 de SSL.
  - Los cambios en SSL fueron relativamente pequeños, pero suficientes como para que SSL versión 3 y TLS no puedan interoperar.
  - Debido a esta incompatibilidad, la mayoría de los navegadores implementan ambos protocolos. A esto se le conoce como SSL/TLS.
- TLS versión 1.0 RFC 2246 (enero 1999).
- TLS versión 1.1 RFC 4346 (abril 2006).
- TLS versión 1.2 RFC 5246 (agosto de 2008).
- TLS versión 1.3 RFC 8446 (agosto 2018).

SSL/TLS

- SSL/TLS se propuso inicialmente para el protocolo HTTP.
  - Cuando se utiliza HTTP sobre SSL, se conoce como HTTPS (HTTP Seguro), aunque sea el protocolo HTTP estándar.
  - Algunas veces los servidores HTTPS usan el puerto 443 en lugar del 80 habitual de TCP.
- Pero hoy en día TLS se usa con muchos otros protocolos de nivel de aplicación.
  - SMTP, POP, IMAP, FTP, etc.
  - En algunos casos se usa un puerto distinto y la comunicación se cifra desde el principio, en otros se pasa a cifrar con TLS cuando se ejecuta un comando STARTLS.

TLS: Funcionamiento

- TLS está formado de dos subprotocolos, uno para establecer una conexión segura y otro para utilizarla.
- Para establecer una conexión segura:
  - (esto se hace normalmente justo a continuación del establecimiento de la conexión TCP)



TLS: Establecimiento de conexión segura

- Establecimiento de conexión segura:
  1. *ClientHello*: el cliente especifica la versión TSL que tiene y sus preferencias con respecto a los algoritmos criptográficos y de compresión. También envía un reto R_A.
  2. *ServerHello*: el servidor selecciona unos algoritmos de los que soporta el cliente y envía otro reto R_B.
  3. Certificados: el servidor envía su certificado de clave pública.
     - Si el certificado no está firmado por alguna autoridad bien conocida, también puede enviar una cadena de certificados que pueden seguirse hasta encontrar una (por ejemplo, uno de los precargados habitualmente en los navegadores).
  4. *ServerHelloDone*: el servidor indica que ha terminado y es el turno del cliente

TLS: Establecimiento de conexión segura

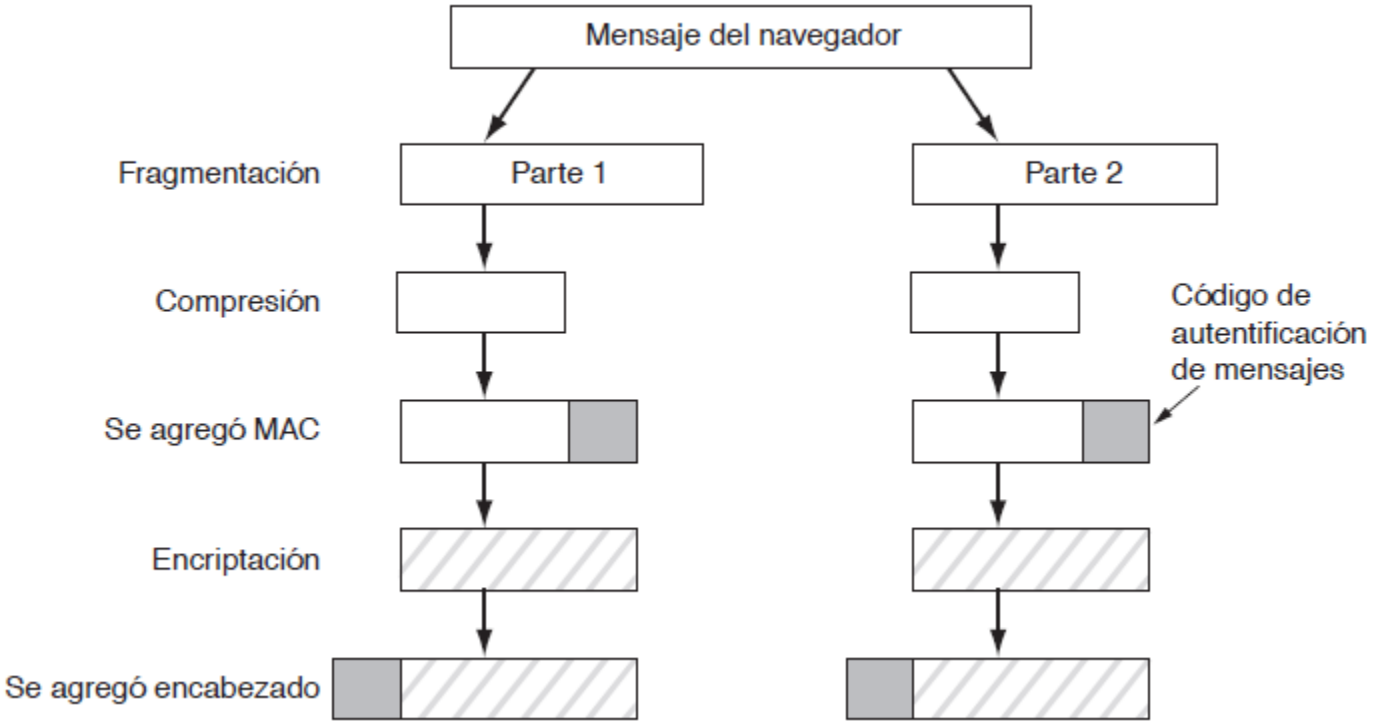
- Establecimiento de conexión segura:
  5. *ClientKeyExchange*: el cliente elige una clave premaestra aleatoria de 384 bits y la envía al servidor cifrada con la clave pública de el servidor.
     - La clave de sesión se obtiene a partir de esta clave premaestra y los dos retos. Tanto cliente como servidor la pueden calcular
  6. *ChangeCipherSpec*: el cliente indica que se pase a cifrar.
  7. *Finished*: el cliente indica el final del proceso de establecimiento de conexión segura
  8. y 9. *ChangeCipherSpec* y *Finished*: el servidor confirma los mensajes anteriores.

TLS: Envío de datos

- El cliente sabe quién es el servidor, pero el servidor no sabe quién es el cliente → En este punto el servidor puede solicitar un certificado al cliente o autenticarse con nombre de usuario y contraseña.
  - TLS no define cómo hacer esto, se haría con otro protocolo, por ejemplo en ESMTP con AUTH.
- A partir de aquí, para el envío de datos se usa otro subprotocolo.
  - Se fragmentan los datos en bloques de 16 KB máximo.
  - Se comprime cada bloque.
  - Se genera un resumen del mensaje.
  - Se cifra el conjunto usando la clave de sesión con un algoritmo de clave secreta.
  - Se envía.

TLS: Envío de datos

- Envío de datos:



TLS: Protocolos de clave pública

- Protocolos de clave pública para autenticación e intercambio de claves soportados en SSL/TLS

Algoritmo	SSL 2.0	SSL 3.0	TLS 1.0	TLS 1.1	TLS 1.2	Status
RSA	Sí	Sí	Sí	Sí	Sí	Definido para TLS 1.2 en RFCs
DH-RSA	No	Sí	Sí	Sí	Sí	
DHE-RSA (forward secrecy)						
ECDH-RSA	No	No	Sí	Sí	Sí	
ECDHE-RSA (forward secrecy)						
DH-DSS	No	Sí	Sí	Sí	Sí	
DHE-DSS (forward secrecy)						
ECDH-ECDSA	No	No	Sí	Sí	Sí	
ECDHE-ECDSA (forward secrecy)						
DH-ANON (inseguro)	No	No	Sí	Sí	Sí	
ECDH-ANON (inseguro)	No	No	Sí	Sí	Sí	
GOST R 34.10-94 / 34.10-2001[14]	No	No	Sí	Sí	Sí	Propuesto en borradores RFC

TLS: Protocolos de clave simétrica

- Protocolos de clave simétrica para cifrado de bloques soportados en SSL/TLS

Cifrado	Versión del Protocolo				
	SSL 2.0	SSL 3.0 note 1 note 2 note 3	TLS 1.0 note 1 note 3	TLS 1.1 note 1	TLS 1.2 note 1
AES CBC note 4	—	—	depende	Seguro	Seguro
AES GCM 15 note 5	—	—	—	—	Seguro
AES CCM 16 note 5	—	—	—	—	Seguro
Camellia CBC 17 note 4	—	—	depende	Seguro	Seguro
Camellia GCM 18 note 5	—	—	—	—	Seguro
SEED CBC 19 note 4	—	—	depende	Seguro	Seguro
ChaCha20+Poly1305 20 note 5	—	—	—	—	Seguro
IDEA CBC note 4 note 6	Inseguro	depende	depende	Seguro	—
3DES CBC note 4 note 7	Inseguro	depende	depende	depende	depende
DES CBC note 4 note 6	Inseguro	Inseguro	Inseguro	Inseguro	—
RC2 CBC note 4 note 6	Inseguro	Inseguro	Inseguro	Inseguro	—
RC4 note 8	Inseguro	Inseguro	Inseguro	Inseguro	Inseguro

TLS: ALPN

- Algunos protocolos han reservado puertos para la versión segura (sobre TLS) del protocolo.
  - Por ejemplo HTTPS (443), IMAP sobre TLS (993), POP sobre TLS (995), SMTP sobre TLS (465)…
- Si usamos TLS sobre un puerto no bien conocido, TLS dispone de una extensión denominada ALPN (*Application Layer Protocol Negotiation*) :
  - Permite durante el establecimiento de la conexión segura negociar el protocolo de nivel de aplicación que van a usar cliente y servidor.
    - En el *ClientHello* el cliente envía la lista de protocolos de nivel de aplicación soportados (*ProtocolNameList*).
    - En el *ServerHello* el servidor devuelve el protocolo de nivel de aplicación seleccionado (*ProtocolName*).

OpenSSL

- Proyecto de software libre que implemente SSL/TLS.
- Podemos usarlo para abrir una sesión SSL/TLS con un servidor.
  - Similar a lo que hacemos con telnet para los protocolos de nivel de aplicación sobre TCP en claro

Ejemplo 1 OpenSSL

```
~>openssl s_client -connect pop.gmail.com:995
CONNECTED(00000003)
depth=2 /C=US/O=GeoTrust Inc./CN=GeoTrust Global CA
verify error:num=20:unable to get local issuer certificate
verify return:0
---
Certificate chain
 0 s:/C=US/ST=California/L=Mountain View/O=Google Inc/CN=pop.gmail.com
   i:/C=US/O=Google Inc/CN=Google Internet Authority G2
 1 s:/C=US/O=Google Inc/CN=Google Internet Authority G2
   i:/C=US/O=GeoTrust Inc./CN=GeoTrust Global CA
 2 s:/C=US/O=GeoTrust Inc./CN=GeoTrust Global CA
   i:/C=US/O=Equifax/OU=Equifax Secure Certificate Authority
---
Server certificate
-----BEGIN CERTIFICATE-----
[...]
-----END CERTIFICATE-----
```

Ejemplo 1 OpenSSL

```
subject=/C=US/ST=California/L=Mountain View/O=Google Inc/CN=pop.gmail.com
issuer=/C=US/O=Google Inc/CN=Google Internet Authority G2
---
No client certificate CA names sent
---
SSL handshake has read 3238 bytes and written 447 bytes
---
New, TLSv1/SSLv3, Cipher is AES128-SHA
Server public key is 2048 bit
Compression: NONE
Expansion: NONE
SSL-Session:
    Protocol  : TLSv1
    Cipher    : AES128-SHA
    Session-ID: FFEC487570FF7F654089FE9FAE2B378CF9EBD9B6116124B556CB73E6E8AA9B35
    Session-ID-ctx:
    Master-Key:
    83B5E119017A8D2E9E57B1EA96D47603D52E7DC5C373FF4F6912B15A9CDC911CE9470049DE41A9
    67777CA64005356D46
```

Ejemplo 1 OpenSSL

```
Key-Arg   : None
    Start Time: 1449114175
    Timeout   : 300 (sec)
    Verify return code: 20 (unable to get local issuer certificate)
---
+OK Gpop ready for requests from 163.117.139.253 t203mb34342019wle
USER cgr@it.uc3m.es
+OK send PASS
PASS XXXXXXXXXX
+OK Welcome.
STAT
+OK 347 12306594
QUIT
DONE
```

Ejemplo 2 OpenSSL

```
~>openssl s_client -connect aulaglobal.uc3m.es:443
CONNECTED(00000003)
depth=2 /C=US/ST=New Jersey/L=Jersey City/O=The USERTRUST Network/CN=USERTrust RSA
    Certification Authority
verify error:num=20:unable to get local issuer certificate
verify return:0
---
Certificate chain
 0 s:/OU=Domain Control Validated/CN=aulaglobal.uc3m.es
   i:/C=NL/ST=Noord-Holland/L=Amsterdam/O=TERENA/CN=TERENA SSL CA 2
 1 s:/C=NL/ST=Noord-Holland/L=Amsterdam/O=TERENA/CN=TERENA SSL CA 2
   i:/C=US/ST=New Jersey/L=Jersey City/O=The USERTRUST Network/CN=USERTrust RSA
    Certification Authority
 2 s:/C=US/ST=New Jersey/L=Jersey City/O=The USERTRUST Network/CN=USERTrust RSA
    Certification Authority
   i:/C=SE/O=AddTrust AB/OU=AddTrust External TTP Network/CN=AddTrust External CA
    Root
---
Server certificate
```

Ejemplo 2 OpenSSL

```
----BEGIN CERTIFICATE-----
[...]
-----END CERTIFICATE-----
subject=/OU=Domain Control Validated/CN=aulaglobal.uc3m.es
issuer=/C=NL/ST=Noord-Holland/L=Amsterdam/O=TERENA/CN=TERENA SSL CA 2
---
No client certificate CA names sent
---
SSL handshake has read 4827 bytes and written 319 bytes
---
New, TLSv1/SSLv3, Cipher is DHE-RSA-AES256-SHA
Server public key is 2048 bit
Compression: NONE
Expansion: NONE
SSL-Session:
    Protocol  : TLSv1
    Cipher    : DHE-RSA-AES256-SHA
    Session-ID: 87C234499CABFEE592D526B3F203513F4E4DC1434847D58B4186638080D9C60A
```

Ejemplo 2 OpenSSL

```
Session-ID-ctx:
    Master-Key:
    50FAFDAD6B26F3ABF9498E48CE6F69C1183F77F4D12B32A92121603093D163FD7E0CF08B1ADCCD
    A9A1EC3B6491131EAD
    Key-Arg   : None
    Start Time: 1449114699
    Timeout   : 300 (sec)
    Verify return code: 20 (unable to get local issuer certificate)
---
GET / HTTP/1.1
Host: aulaglobal.uc3m.es
Connection: close
```

Ejemplo 2 OpenSSL

```
HTTP/1.1 303 See Other
Date: Thu, 03 Dec 2015 03:51:46 GMT
Server: Apache
Set-Cookie: MoodleSessionag3=9m09h2rhrm67kiru0okpvr2nf6; path=/
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Location: https://aulaglobal.uc3m.es/login/index.php
Content-Language: es-es
Content-Length: 620
Connection: close
Content-Type: text/html

[...]
closed
violin:~>
```