

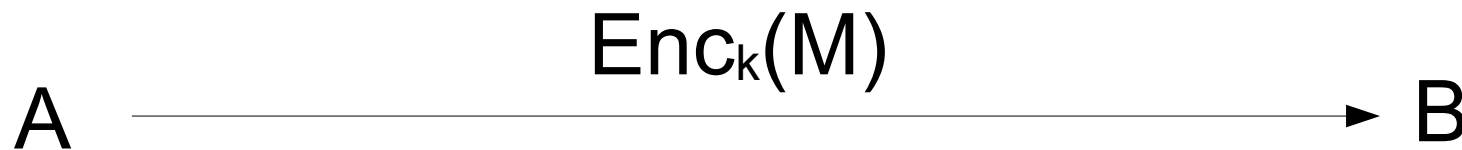


# Criptografía y Seguridad

Criptografía:  
Protocolos

# Distribución de claves (repaso)

- Un criptosistema CCA-Secure permite enviar información manteniendo:
  - Confidencialidad
  - Integridad



- Pero requiere que ambas partes conozcan una misma clave

# Intercambio de claves (repaso)

- Es un protocolo  $\Pi(n)$ , ejecutado por dos partes:
  - $\Pi: (n) \rightarrow \text{Tran}, k_a, k_b$
  - No tiene entrada (salvo el parámetro de seguridad)
  - La salida del protocolo es
    - Un conjunto de mensajes intercambiados
    - Una clave  $k_a$  conocida solo por una de las partes
    - Una clave  $k_b$  conocida solo por la otra parte
- Condición fundamental:
  - $k_a = k_b$

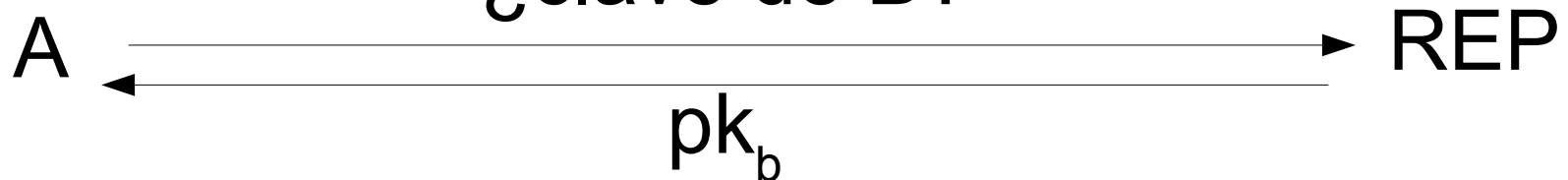
# Ataques Activos

- Un atacante puede
  - Omitir mensajes
  - Reescribir contenido de mensajes
  - Reordenar mensajes
  - Repetir mensajes
- Los esquema de intercambio vistos hasta ahora no sirven con atacantes activos
  - Ataques Man In The Middle

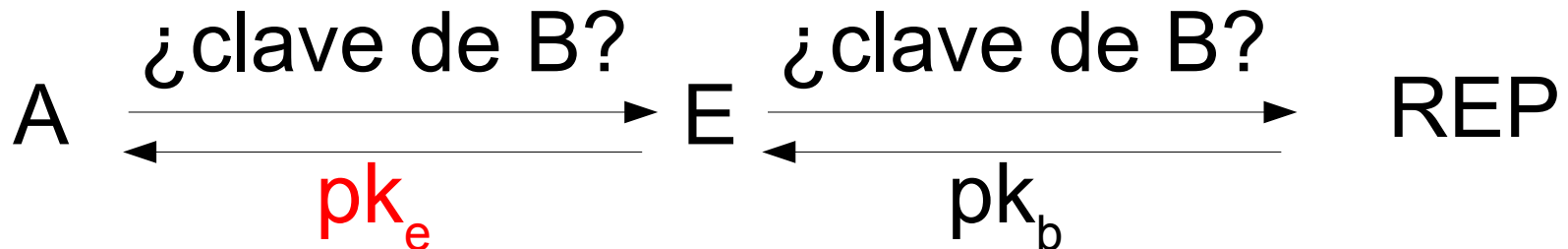
# Problemas

- ¿De donde se obtienen las claves públicas?
  - Se confía implícitamente en la asociación de las claves con la identidad de la personas
- Si se vulnera esa asociación puede realizarse un ataque Man-In-The-Middle:

- ¿Como consigue A la clave pública de B?  
¿clave de B?



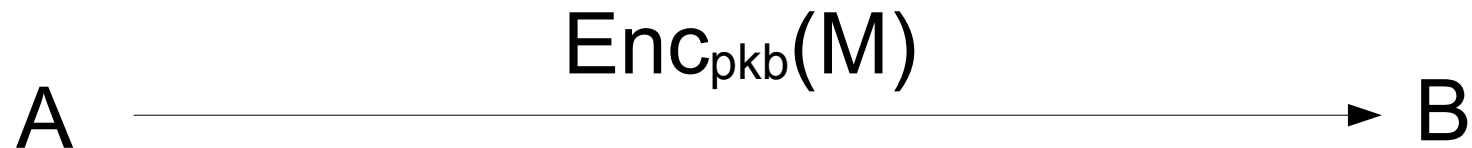
- Lo que pasa en realidad:



# Man in the middle

- Una vez que A tiene una clave equivocada, intenta comunicarse con B:

- Lo que A imagina:



- Lo que pasa en realidad:



- El problema entra en la esfera de administración de claves

# Infraestructura de claves (PKI)

- Objetivo: Asociar identidad a las claves
  - Se busca evitar problemas de suplantación de identidad (man in the middle o spoofing)
  - No es aplicable a los criptosistemas simétricos
- Motivo
  - La selección de claves depende de las partes involucradas en la comunicación
  - Usar la clave equivocada significa que no hay garantías de confidencialidad o integridad.

# Certificados

- Son mensajes que contienen al menos:
  - Información de identidad (ej: nombre)
  - Clave pública asociada
  - Fecha de emisión
  - Intervalo de validez
  - Tipo de uso de la clave
    - Firma de mensajes
    - Cifrado de emails
    - Firma de certificados
    - Cifrado de sitios web
- Los certificados están a su vez firmados digitalmente por una autoridad competente



# Uso de certificados

- A solicita certificado de B
  - Puede ser provisto B o cualquier otra entidad
- A puede verificar la identidad del certificado
  - CN=Common Name
  - Emails: CN=<dirección de email>
  - Servidores: CN=<IP> o <hostname>
  - Empresas: CN=Razón social
- A puede obtener la clave pública de B
  - Parte del certificado mismo
  - Incluye información del tipo de clave
    - Ej: RSA-2048, DSA-EC 320, etc

# Uso de certificados (2)

- A puede verificar la validez del certificado
  - Comprobando el tipo de uso permitido
  - Comprobando la fecha de vigencia
- A puede verificar la integridad del certificado
  - Validando la firma digital de la autoridad que lo certifica

La validación de una firma digital requiere una clave pública  
¿Cómo se obtiene?

# Cadenas de firmas

- ¡Las autoridades certificantes tienen a su vez un certificado!
  - Se acostumbra incluir el certificado de la autoridad junto con cada certificado
- ¿Quien valida el certificado de la AC?
  - ¡Otra AC!
- AC Raices
  - Son aquellas autoridades que firman su propio certificado.
  - Son el punto de confianza del sistema

# Cadenas de firmas (2)

- Concepto de AC Raíces
  - Confiar en una única autoridad certificante
  - Dicha autoridad delega en otras la capacidad de firmar certificados
- Ejemplos:
  - Certificado de A, firmado por CA3 ( $A \ll CA3 \gg$ )
    - $C_a = C'_a \parallel C_{CA3} \parallel C_{CA2} \parallel C_{CA1}$
  - Certificado de B, firmado por CA1 ( $B \ll CA1 \gg$ )
    - $C_b = C'_b \parallel C_{CA1}$

# Validación entre C.A.

- No existe una única C.A. Raíz
- Si A y B tienen dos C.A. diferentes, ¿como valida A el certificado de B?
  - Opción 1: A confía en la C.A. de B
  - Opción 2: Las C.A. Se certifican entre sí
    - Cada C.A. Emite un certificado de la identidad de la otra
- En la práctica:
  - Existe una lista de C.A.s reconocidas:
    - En el sistema operativo
    - En los navegadores
    - En runtimes como JVM

# Certificados X.509

- Estándar de certificados digitales
- Incluye:
  - Versión
  - Número de serie
  - Identificador de algoritmo de firma
  - Nombre del emisor (C.A.)
  - Intervalo de validez
  - Nombre del sujeto (C.N. - common name)
  - Clave pública del sujeto
  - Firma: firma del hash de todo lo anterior
- Forman cadenas de certificados

# X.509

```
1.Certificate:
2.  Data:
3.    Version: 3 (0x2)
4.    Serial Number: 1 (0x1)
5.    Signature Algorithm: md5WithRSAEncryption
6.    Issuer: C=ZA, ST=Western Cape, L=Cape Town, O=Thawte Consulting cc,
7.           OU=Certification Services Division,
8.           CN=Thawte Server CA/Email=server-certs@thawte.com
9.    Validity
10.       Not Before: Aug  1 00:00:00 1996 GMT
11.       Not After : Dec 31 23:59:59 2020 GMT
12.       Subject: C=ZA, ST=Western Cape, L=Cape Town, O=Thawte Consulting cc,
13.              OU=Certification Services Division,
14.              CN=Thawte Server CA/Email=server-certs@thawte.com
15.       Subject Public Key Info:
16.         Public Key Algorithm: rsaEncryption
17.         RSA Public Key: (1024 bit)
18.         Modulus (1024 bit):
19.           00:d3:a4:50:6e:c8:ff:56:6b:e6:cf:5d:b6:ea:0c:
20.           68:75:47:a2:aa:c2:da:84:25:fc:a8:f4:47:51:da:
21.           85:b5:20:74:94:86:1e:0f:75:c9:e9:08:61:f5:06:
22.           6d:30:6e:15:19:02:e9:52:c0:62:db:4d:99:9e:e2:
23.           6a:0c:44:38:cd:fe:be:e3:64:09:70:c5:fe:b1:6b:
24.           29:b6:2f:49:c8:3b:d4:27:04:25:10:97:2f:e7:90:
25.           6d:c0:28:42:99:d7:4c:43:de:c3:f5:21:6d:54:9f:
26.           5d:c3:58:e1:c0:e4:d9:5b:b0:b8:dc:b4:7b:df:36:
27.           3a:c2:b5:66:22:12:d6:87:0d
28.         Exponent: 65537 (0x10001)
29.       X509v3 extensions:
30.         X509v3 Basic Constraints: critical
31.         CA:TRUE
32.       Signature Algorithm: md5WithRSAEncryption
33.         07:fa:4c:69:5c:fb:95:cc:46:ee:85:83:4d:21:30:8e:ca:d9:
34.         a8:6f:49:1a:e6:da:51:e3:60:70:6c:84:61:11:a1:1a:c8:48:
35.         3e:59:43:7d:4f:95:3d:a1:8b:b7:0b:62:98:7a:75:8a:dd:88:
36.         4e:4e:9e:40:db:a8:cc:32:74:b9:6f:0d:c6:e3:b3:44:0b:d9:
37.         8a:6f:9a:29:9b:99:18:28:3b:d1:e3:40:28:9a:5a:3c:d5:b5:
38.         e7:20:1b:8b:ca:a4:ab:8d:e9:51:d9:e2:4c:2c:59:a9:da:b9:
39.         b2:75:1b:f6:42:f2:ef:c7:f2:18:f9:89:bc:a3:ff:8a:23:2e:
40.         70:47
```

# X.509

```
1. Certificate
2.   Data
3.     Issuer: C=ZA, ST=Western Cape, L=Cape Town, O=Thawte Consulting cc,
4.             OU=Certification Services Division,
5.             CN=Thawte Server CA/Email=server-certs@thawte.com
6.
7.     CN=Thawte Server CA/Email=server-certs@thawte.com
8.
9.     Validity
10.      Not Before: Aug  1 00:00:00 1996 GMT
11.      Not After : Dec 31 23:59:59 2020 GMT
12.
13.     Subject: C=ZA, ST=Western Cape, L=Cape Town, O=Thawte Consulting cc,
14.             OU=Certification Services Division,
15.             CN=Thawte Server CA/Email=server-certs@thawte.com
16.
17.
18.     Modulus (1024 bit):
19.       00:d3:a4:50:6e:c8:ff:56:6b:e6:cf:5d:b6:ea:0c:
20.       68:75:47:a2:aa:c2:da:84:25:fc:a8:f4:47:51:da:
21.       05:15:00:54:04:06:10:00:75:00:00:00:61:05:06:
22.
23.     Validity
24.      Not Before: Aug  1 00:00:00 1996 GMT
25.      Not After : Dec 31 23:59:59 2020 GMT
26.
27.
28.     Exponent: 65537 (0x10001)
29.     X509v3 extensions:
30.       X509v3 Basic Constraints: critical
31.       CA:TRUE
32.
33.     X509v3 extensions:
34.       X509v3 Basic Constraints: critical
35.       CA:TRUE
36.
37.
38.     b2:75:1b:f6:42:f2:ef:c7:f2:18:f9:89:bc:a3:ff:8a:23:2e:
39.     70:47
```



# X.509

1. Certificate:

2.     Da

3.     Subject Public Key Info:

4.         Public Key Algorithm: rsaEncryption

5.         RSA Public Key: (1024 bit)

6.             Modulus (1024 bit):

7.                 00:d3:a4:50:6e:c8:ff:56:6b:e6:cf:5d:b6:ea:0c:

8.                 68:75:47:a2:aa:c2:da:84:25:fc:a8:f4:47:51:da:

9.                 85:b5:20:74:94:86:1e:0f:75:c9:e9:08:61:f5:06:

10.                6d:30:6e:15:19:02:e9:52:c0:62:db:4d:99:9e:e2:

11.                6a:0c:44:38:cd:fe:be:e3:64:09:70:c5:fe:b1:6b:

12.                29:b6:2f:49:c8:3b:d4:27:04:25:10:97:2f:e7:90:

13.                6d:c0:28:42:99:d7:4c:43:de:c3:f5:21:6d:54:9f:

14.                5d:c3:58:e1:c0:e4:d9:5b:b0:b8:dc:b4:7b:df:36:

15.                3a:c2:b5:66:22:12:d6:87:0d

16.             Exponent: 65537 (0x10001)

17.             5d:c3:58:e1:c0:e4:d9:5b:b0:b8:dc:b4:7b:df:36:

18.     Signature Algorithm: md5WithRSAEncryption

19.         07:fa:4c:69:5c:fb:95:cc:46:ee:85:83:4d:21:30:8e:ca:d9:

20.         a8:6f:49:1a:e6:da:51:e3:60:70:6c:84:61:11:a1:1a:c8:48:

21.         3e:59:43:7d:4f:95:3d:a1:8b:b7:0b:62:98:7a:75:8a:dd:88:

22.         4e:4e:9e:40:db:a8:cc:32:74:b9:6f:0d:c6:e3:b3:44:0b:d9:

23.         8a:6f:9a:29:9b:99:18:28:3b:d1:e3:40:28:9a:5a:3c:d5:b5:

24.         e7:20:1b:8b:ca:a4:ab:8d:e9:51:d9:e2:4c:2c:59:a9:da:b9:

25.         b2:75:1b:f6:42:f2:ef:c7:f2:18:f9:89:bc:a3:ff:8a:23:2e:

26.         70:47

# Verificación de certificados X.509

- Obtener la clave pública del emisor
  - En la cadena de certificados o del sistema si es raíz
  - Si es necesario, verificar recursivamente el certificado del emisor
- Verificar integridad del certificado
  - Utilizando el algoritmo especificado y la clave pública del emisor
- Verificar intervalo de validez
  - Debe estar vigente.
  - La autoridad certificante debe estar vigente al comienzo del periodo de vigencia.

# Verificación de certificados X.509

- Verificar identidad de certificado
  - Comparar el CN con el que espera la aplicación
  - Esto depende del uso del certificado
- Verificar el uso del certificado
  - Validar que el certificado esté autorizado para el uso que se le quiere dar

# Revocación de claves

- Necesidad de invalidar una clave ANTES de su expiración
  - Fue averiguada por un atacante
  - Cambio anticipado (cambio de dueño de clave)
- Problemas
  - No revocar claves que no deben ser revocadas
    - Evitar que se revoquen claves sin autorización
  - Propagar la información para evitar futuras comunicaciones con dicha clave

# Listas de revocación

- Listas de certificados revocados
  - Similar a listas de números de tarjeta de crédito robadas
  - Listado actual: certificados vigentes y revocados
  - Listado histórico: certificados expirados y revocados
- X.509: Solo el emisor de un certificado puede revocarlo
  - Se agrega al CRL de la autoridad certificante global
  - La lista se puede obtener para validar offline o puede ser validada online

# Resumen certificados

- Asocian una clave publica con una identidad
- Se pueden validar offline
  - Cuidado con posibles revocaciones
- Son certificados por otra entidad
- Forman cadenas de confianza
- Resuelven el tema de integridad de las claves publicas

# Ejercicio

- Describir todas las validaciones necesarias:



Common name: www.itba.edu.ar  
SANs: www.itba.edu.ar  
Valid from June 6, 2017 to July 7, 2018  
Serial Number: 0dd6836dc41e256a7f3c79e00b4672fd  
Signature Algorithm: sha256WithRSAEncryption  
Issuer: Amazon



Common name: Amazon  
Organization: Amazon Org. Unit: Server CA 1B  
Location: US  
Valid from October 21, 2015 to October 18, 2025  
Serial Number: 067f94578587e8ac77deb253325bbc998b560d  
Signature Algorithm: sha256WithRSAEncryption  
Issuer: Amazon Root CA 1



Common name: Amazon Root CA 1  
Organization: Amazon  
Location: US  
Valid from May 25, 2015 to December 30, 2037  
Serial Number: 067f944a2a27cdf3fac2ae2b01f908eeb9c4c6  
Signature Algorithm: sha256WithRSAEncryption  
Issuer: Starfield Services Root Certificate Authority - G2



Common name: Starfield Services Root Certificate Authority - G2  
Organization: Starfield Technologies, Inc.  
Location: Scottsdale, Arizona, US  
Valid from September 1, 2009 to June 28, 2034  
Serial Number: 12037640545166866303 (0xa70e4a4c3482b77f)  
Signature Algorithm: sha256WithRSAEncryption  
Issuer: Starfield Technologies, Inc.

# Intercambio simétrico de claves

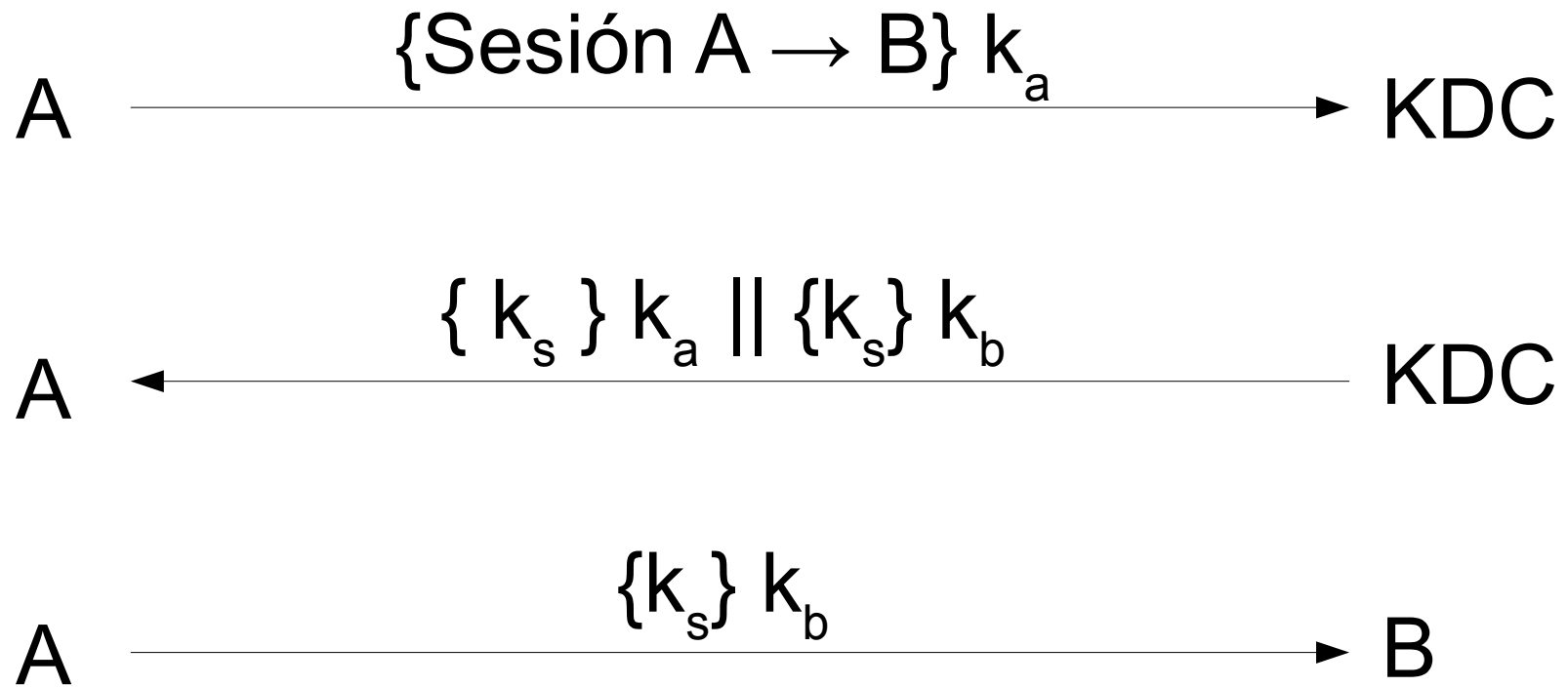
Equivalente simétrico a DH



# Protocolo Needham-Schroeder

- Protocolo de intercambio de claves simétrico
  - Es el protocolo utilizado por Kerberos y Active Directory (con modificaciones)
  - Requiere un servicio centralizado (KDC)
- Genera claves de sesión entre pares
  - Parte de la hipótesis de que cada entidad comparte una clave con el KDC

# Primera aproximación

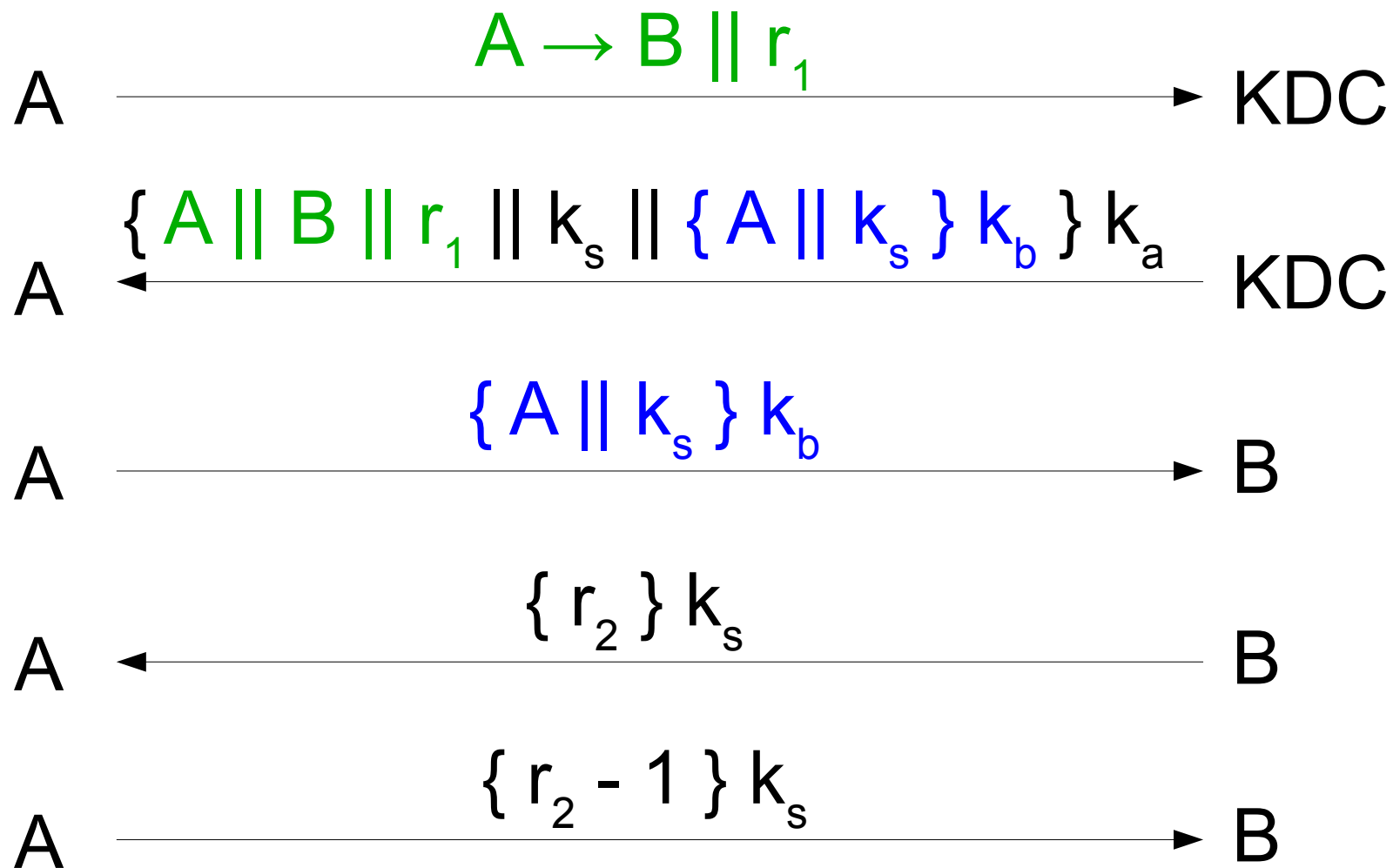


Para simplificar, se utiliza la nomenclatura:  
 $\text{Enc}_k(M) = \{ M \} k$

# Problemas

- Repetición (Replay):
  - Un atacante puede grabar mensajes de A hacia B y luego reenviar  $\{k_s\} k_b$  y los mensajes siguientes
  - B no tiene forma de saber que no está hablando con A
- Reuso de claves (Key Reuse)
  - Un atacante graba el mensaje de C a A, y cuando A quiere iniciar otra conversación se lo envía
  - A y B estarán utilizando la misma clave de sesión

# Segunda aproximación



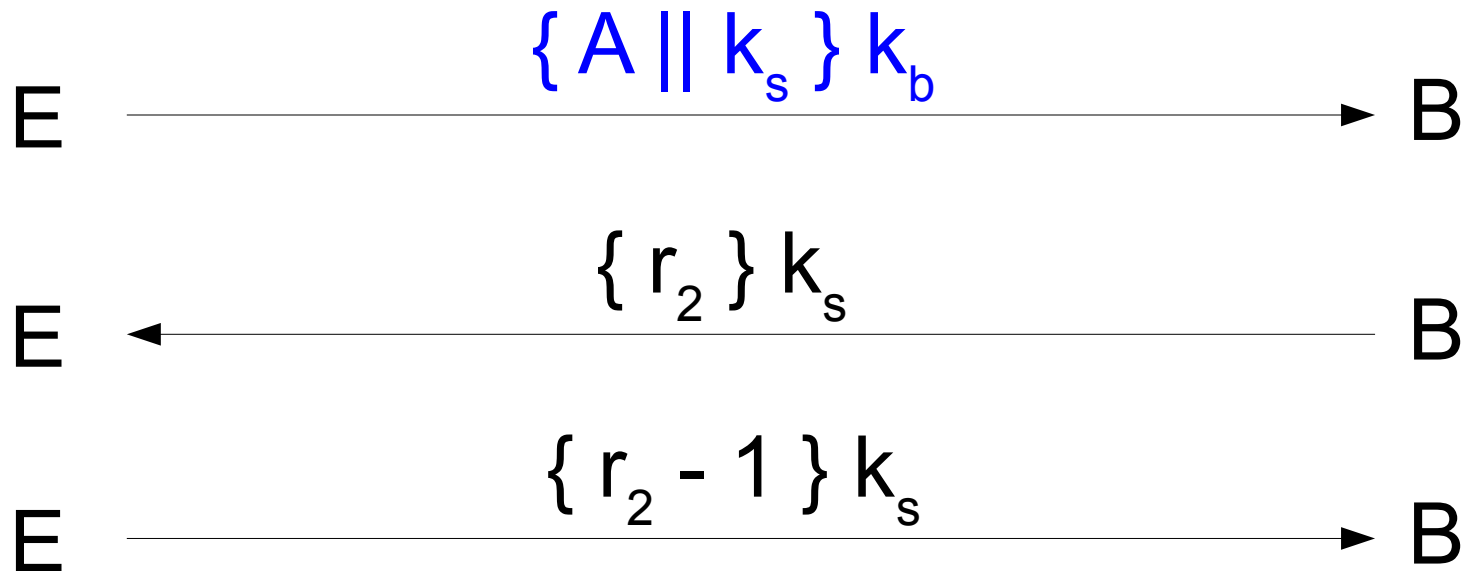
# Explicación

- Segundo mensaje
  - Encriptado con clave compartida A-C
    - El mensaje proviene de C
  - No es repetición
    - El número  $r_1$  recibido coincide con el enviado
- Cuarto mensaje
  - Solo B puede enviarlo
    - Avisa a A de un intento de comunicación
- Quinto mensaje
  - A confirma la comunicación
  - B sabe que no es una repetición (por  $r_2$ )

# Problema

Escenario: El atacante (E) obtiene una clave de sesión antigua  $k_s$

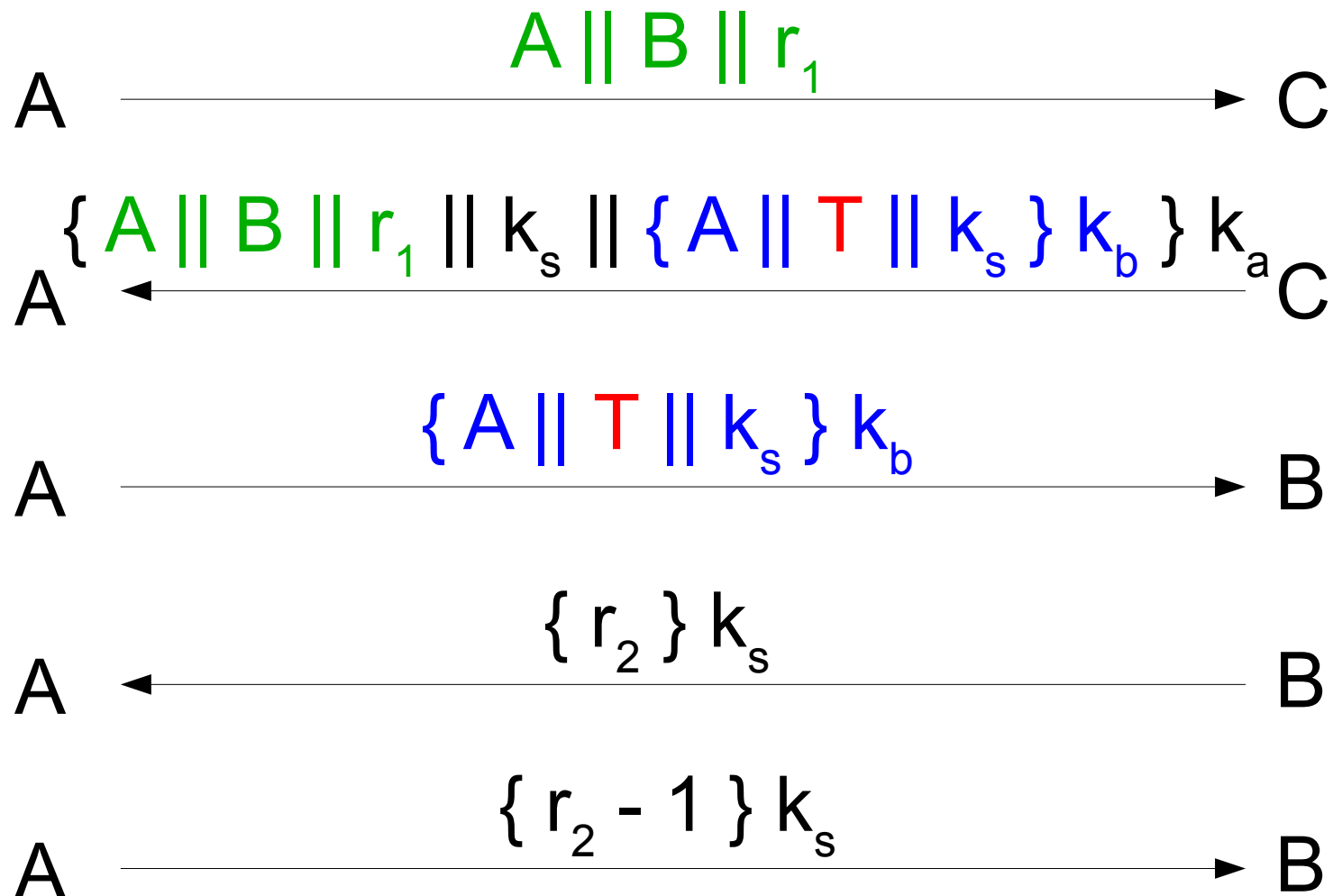
- Comienza por el tercer paso:



E logra impersonar a A!

# Modificación Demming-Sacco

- Arreglo propuesto: Usar timestamps



# Canal seguro

Confidencialidad e integridad  
sobre un canal inseguro

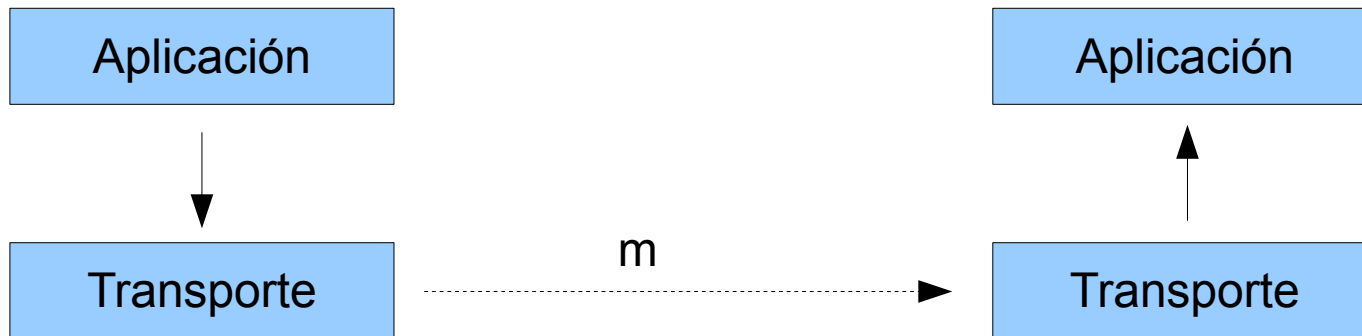


# SSL / TLS

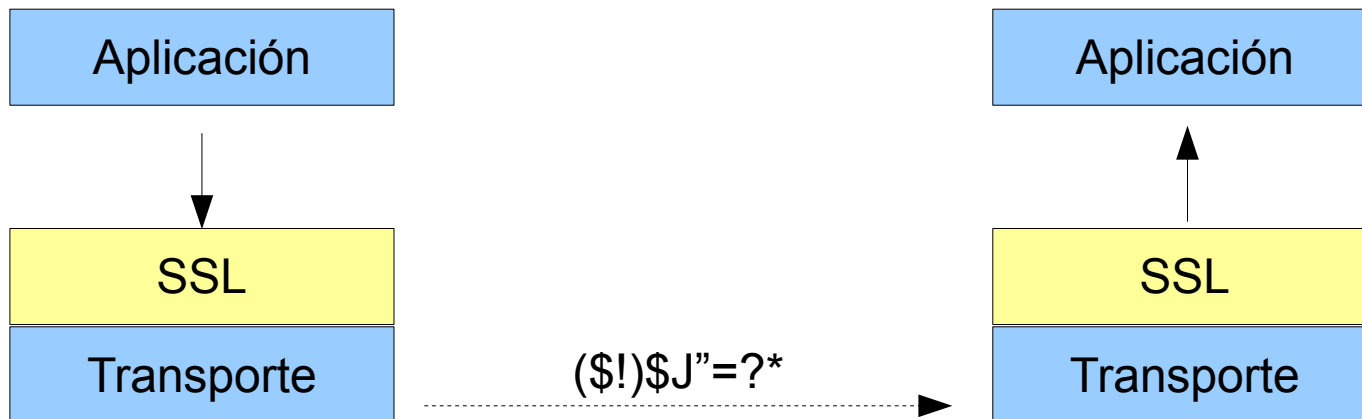
- Ofrece Seguridad a nivel transporte
- Utiliza un canal de transporte confiable (TCP?)
- Provee confidencialidad, integridad y autenticación de los participantes (origen y destino)
- SSL – Secure Socket Layer
  - Versión inicial, creada por Netscape
- TLS – Transport Layer Security
  - Evolución de SSL (TLS 1.2 = SSL 3.3)
  - Mejora deficiencias encontradas
  - Estándar actual

# Concepto

- Conexión normal, insegura

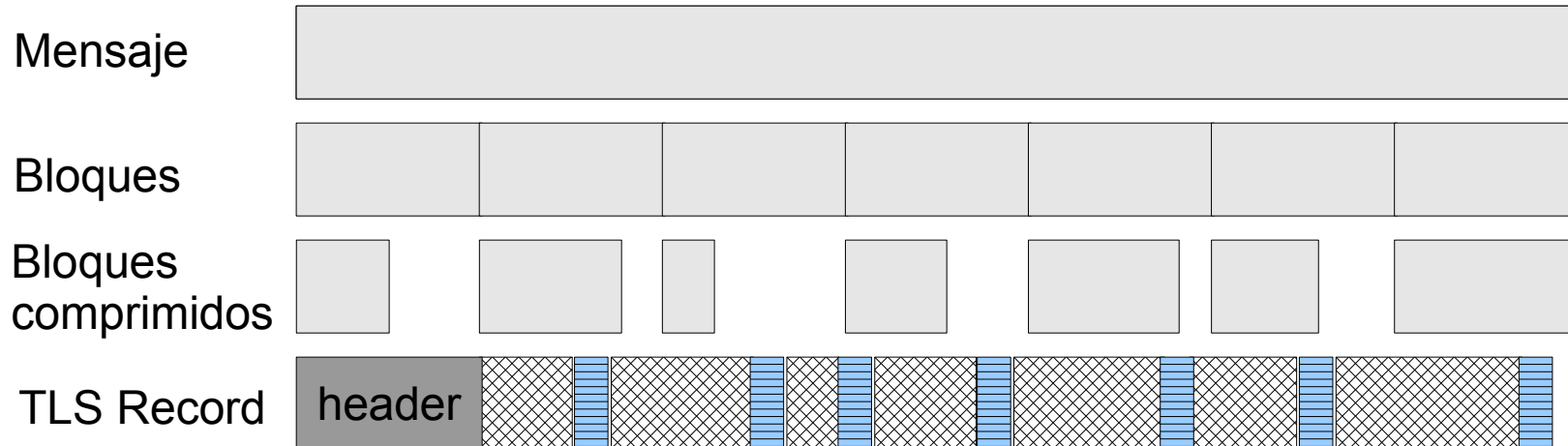


- Conexión con SSL, protegida



# TLS Record

- TLS recibe un mensaje a enviar
- Lo divide en bloques de no más de  $2^{16}$  bytes
- Comprime cada bloque, y calcula su hash
- Encripta cada bloque y hash
- Lo envía al servicio inferior (por ejemplo, TCP)



# TLS – Tipo de mensajes

- TLS handshake
  - Configuración inicial
  - Autenticación
  - Negociación de material criptográfico
- TLS application data
  - Contiene información del protocolo encapsulado
- TLS alert
  - Informa de eventos y problemas
- TLS Change cipher spec
  - Concluye una renegociación de claves

# TLS - Criptografía

- Intercambio de clave:
  - RSA
  - Diffie Hellman, certificado RSA o DSS
  - Diffie Hellman, anonymous
  - ECDH – Diffie Hellman sobre curvas elípticas
  - Kerberos
- Hay variantes obsoletas que utilizan 512 bits
  - Definidas por problemas de exportación de material criptográfico en los EE.UU.

# TLS - Criptografía

- Cifrado simétrico
  - ~~RC4~~
  - ChaCha20 (TLS 1.2)
  - ~~DES-CBC~~
  - Triple DES (3DES-EDE-CBC)
  - AES; AES-CBC, AES-CCM, **AES-GCM**
  - IDEA CBC
  - ARIA: ARIA-CBC, ARIA-CCM, ARIA-GCM
- También hay variantes obsoletas, con claves de 40 bits

# TLS - Criptografía

- Hash
  - HMAC-MD5
  - HMAC-SHA1
  - HMAC-SHA2 (256/384)
  - AEAD (Authenticated Encryption with Associated Data)
- Autenticación
  - Firmas RSA
  - Firmas DSS

# Sesión TLS

- Es una asociación entre dos pares
  - Puede soportar múltiples conexiones
- Información:
  - Identificador unico de sesion
  - Certificado X.509v3 del otro extremo (opcional)
  - Método de compresión acordado
  - Método de encriptación y MAC acordado
  - 'Master Secret': clave de sesión compartida de 48 bytes

Sesiones en SSL: Uso transparente por eficiencia.  
NO es un protocolo de sesión!!!



# Conexión TLS

- Describe como intercambiar datos
- Contiene
  - Secuencia de bits aleatoria de calidad criptográfica
  - Claves de escritura para ambos lados (diferentes)
  - Claves para MACs (hashes con clave) para ambos lados
  - IVs si fuesen necesarios
  - Número de secuencia para cliente y servidor

# TLS Handshake – parte 1

- ClientHello:



- ServerHello:



$V_c$  = Version cliente     $V$  = min (Version Cliente, Version Servidor)

$r_1, r_2$  = nonces (Timestamp y 28 bytes aleatorios)

$S_{id}$  = ID de la sesion. 0 para iniciar una nueva

Ciphers = Lista de stacks criptográficos disponibles

Cipher = Stack criptográfico elegido

Comps = Metodos de compresión disponibles

Comp = Método de compresión elegido

# TLS Handshake – parte 2

- Certificate (si el servidor es autenticado):

$\{ \text{Certificado} \}$

C  $\leftarrow$  \_\_\_\_\_ S

- ServerKeyExchange

$\{ p \parallel \{ \text{hash}(r_1 \parallel r_2 \parallel p) \} k_s \}$

C  $\leftarrow$  \_\_\_\_\_ S

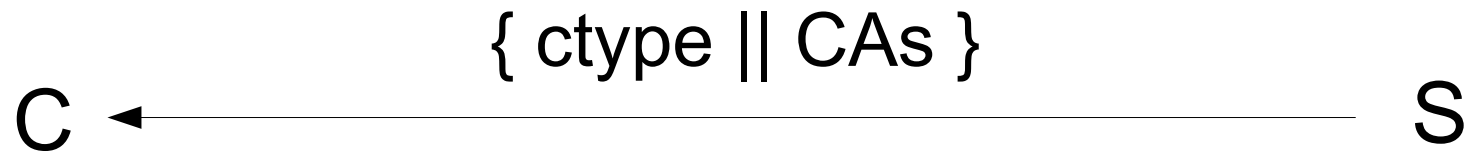
$r_1, r_2$  = nonces utilizados durante la parte inicial (evitan replay)

$p$  = parámetros criptográficos ( $e, n$  – para RSA,  $p, g, g^a$  para DH,  $r$  para fortaleza)

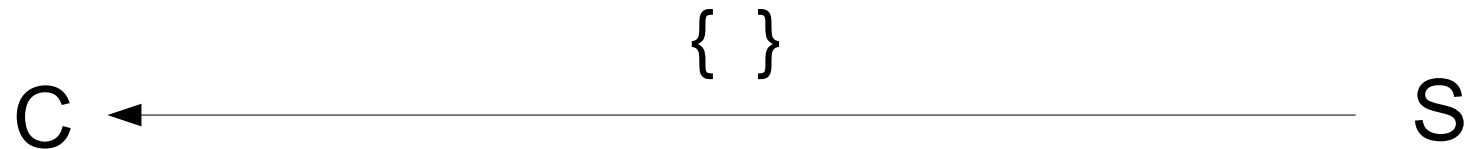
$K_s$  = Clave privada del servidor correspondiente a la pública del certificado

# TLS Handshake – parte 2

- Certificate request (si el cliente es autenticado):



- ServerHelloDone



Ctype = tipo de certificados permitidos, desde el punto de vista algoritmico  
Cas = lista de autoridades certificadoras válidas

# TLS Handshake – parte 3

- Client Certificate (solo si fue solicitado)

C  $\xrightarrow{\{ \text{Certificado} \}}$  S

- Client Key Exchange (version RSA)

C  $\xrightarrow{\{ V \parallel \text{PRE}_{\text{rand}} \} K_s}$  S

- Client Key Exchange (version DH)

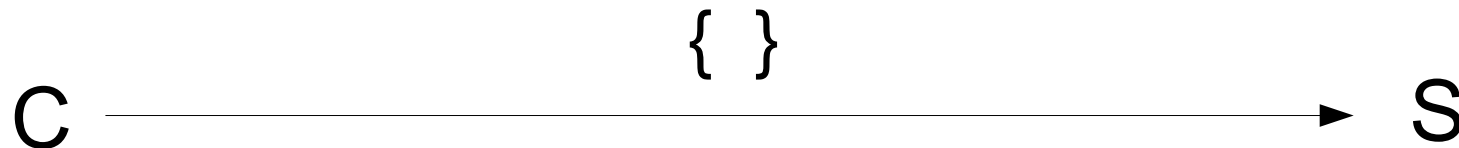
C  $\xrightarrow{\{ g^b \bmod p \} K_s}$  S

$\text{PRE} = g^{ab} \bmod p$

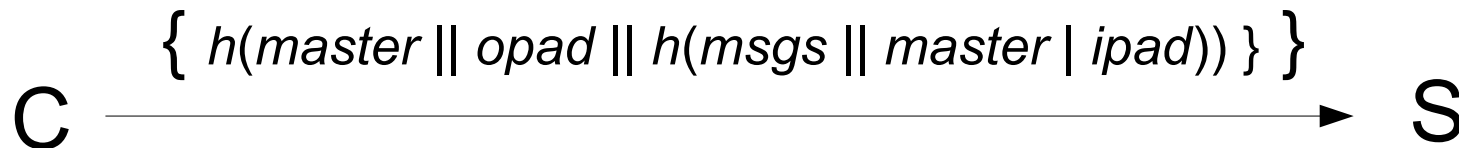
V = Version informada originalmente por el cliente (previene downgrade attacks)  
g, p = parametros de DH informados por el servidor

# TLS Handshake – parte 4

- Change Cipher Spec



- Finish



Master = MD5(pre || SHA('A' || pre ||  $r_1$  ||  $r_2$ ) ||  
MD5(pre || SHA('BB' || pre ||  $r_1$  ||  $r_2$ ) ||  
MD5(pre || SHA('CCC' || pre ||  $r_1$  ||  $r_2$ ))

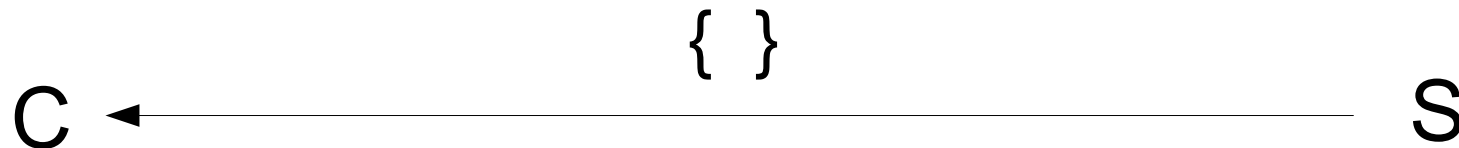
Opad = 01011100 (binario, repetido durante todo un bloque – 20 bytes)

Ipad = 00110110 (binario, repetido durante todo un bloque – 20 bytes)

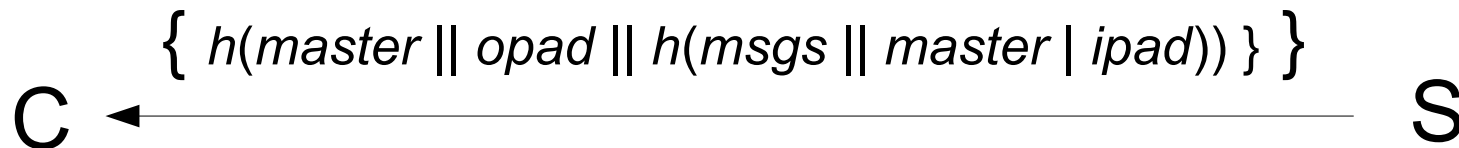
msgs = concatenación de todos los mensajes intercambiados hasta el momento

# TLS Handshake – parte 4

- Change Cipher Spec



- Finish



# TLS Change Cipher Spec

- Utilizado durante el handshake
- Puede aparecer en cualquier momento
- Implica una renegociación (cambio) de las claves de sesion utilizadas
- Lo puede solicitar tanto el cliente como el servidor
- No tiene contenido. Es un tipo de mensaje del protocolo



# TLS Alert

- Envía eventos fuera de banda
  - Pueden ser de advertencia o fatales
  - Un evento fatal invalida la conexión
- CloseNotify: Evento que indica el fin de una sesión
  - No se enviarán mensajes nuevos
  - Si llega un mensaje nuevo será ignorado

# TLS Alert

- Errores fatales
  - Mensaje no esperado,
  - MAC incorrecto,
  - error al descomprimir,
  - error en el handshake,
  - parámetro ilegal
- Advertencias o errores (a elección de quien lo recibe y el contexto):
  - No hay certificado, certificado no válido
  - Certificado no soportado, expirado o revocado

# TLS - Panorama

- Estandar de comunicación segura en servicios web (https)
- Depende de infraestructura de PKI
- No es utilizado frecuentemente para validar clientes
- Soportado por todos los navegadores
  - Diferente respuesta frente a alertas
- Problemas de diseño (en los clientes) hicieron que pierda parte de su utilidad
  - Aún así, es la alternativa mayormente adoptada.

# Lectura Recomendada

## Capítulo 11 Computer Security Art and Science

Matt Bishop

---

RFC 5246 – TLS v1.2  
<http://tools.ietf.org/html/rfc5246>

Descripción de vulnerabilidad encontrada en la  
renegociación de claves  
<http://www.g-sec.lu/practicaltls.pdf>