

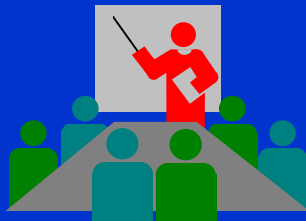
# Capítulo 16

## Autenticación y Firma Digital

### Seguridad Informática y Criptografía



v 4.1



Material Docente de  
Libre Distribución

Ultima actualización del archivo: 01/03/06  
Este archivo tiene: 63 diapositivas

Dr. Jorge Ramió Aguirre  
Universidad Politécnica de Madrid

Este archivo forma parte de un curso completo sobre Seguridad Informática y Criptografía. Se autoriza el uso, reproducción en computador y su impresión en papel, sólo con fines docentes y/o personales, respetando los créditos del autor. Queda prohibida su comercialización, excepto la edición en venta en el Departamento de Publicaciones de la Escuela Universitaria de Informática de la Universidad Politécnica de Madrid, España.

# ¿Interesa confidencialidad o integridad?

- **Confidencialidad**

- Para lograrla se **cifra** el mensaje  $M$ , número  $N$  o clave  $K$  obteniendo un criptograma.

- **Integridad**

- Para lograrla se **firma** un hash del mensaje  $h(M)$ , añadiendo una marca al mensaje o criptograma.

Si bien en ciertos escenarios es muy importante mantener el secreto de la información, siempre que ésta lo requiera, en muchos casos tiene quizás más trascendencia el poder certificar la autenticidad entre cliente y servidor como ocurre en Internet.

# Algunos problemas de integridad

## a) Autenticidad del emisor

¿Cómo comprueba Benito (**B**) que el mensaje recibido del emisor que dice ser Adela (**A**) es efectivamente de esa persona?

## b) Integridad del mensaje

¿Cómo comprueba Benito (**B**) que el mensaje recibido del emisor Adela (**A**) es el auténtico y no un mensaje falso?

## c) Actualidad del mensaje

¿Cómo comprueba Benito (**B**) que el mensaje recibido del emisor Adela (**A**) es actual, no un mensaje de fecha anterior reenviado?

## Más problemas de integridad

### d) No repudio del emisor

¿Cómo comprueba Benito (**B**) que el mensaje enviado por el emisor Adela (**A**) -y que niega haberlo enviado- efectivamente ha llegado?

### e) No repudio del receptor

¿Cómo comprueba Benito (**B**) que el mensaje enviado al receptor Adela (**A**) -y que niega haberlo recibido- efectivamente se envió?

### d) Usurpación de identidad del emisor/receptor

¿Cómo comprueba Benito (**B**) que Adela (**A**), Carmen (**C**) u otros usuarios están enviando mensajes firmados como Benito (**B**)?

# Primer escenario de integridad

Escenario de desconfianza

**1ª Solución.** Uso de una tercera parte de confianza activa. Un juez tendrá una clave  $K_A$  con la que se comunica con **A** y una clave  $K_B$  con la que se comunica con **B**.



Si **A** envía un mensaje  $M$  a **B**:

**A** cifra  $M$  con la clave  $K_A \Rightarrow E_{K_A}(M)$  y lo envía al juez. Este comprueba la integridad de **A**, lo descifra y envía a **B**, cifrado con  $K_B$ , el mensaje  $M$ , la identidad de **A** y la firma  $E_{K_A}(M)$ :  $E_{K_B}\{M, A, E_{K_A}(M)\}$ . Ambos confían en el juez y ante cualquier duda éste puede comprobar la identidad de **A** ante **B** descifrando  $E_{K_A}(M)$ .

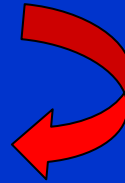
Se usará criptografía simétrica

## Segundo escenario de integridad

Escenario de desconfianza

**2ª Solución.** Uso de una tercera parte de confianza no siempre activa. Esta parte sólo actúa cuando se produce un conflicto entre los interlocutores, quienes se autentican a través de ella que les certifica.

Se usará criptografía asimétrica



En este caso la figura del juez se conoce como una **Autoridad de Certificación**

Habrà una aceptación del sistema de autenticación, tanto por convencimiento propio de los usuarios como por su confianza en los algoritmos.

# Funciones y esquemas de autenticación

- **Autenticación mediante el cifrado de mensajes con criptografía simétrica**
  - La cifra de datos podría servir como autenticación.
- **Autenticación con MAC Message Authentication Code o checksum**
  - Una función pública y una clave secreta producen un valor de longitud fija que es válida como autenticador.
- **Autenticación mediante funciones hash**
  - Una función pública reduce el mensaje a una longitud de valor hash que sirve como autenticador de integridad.
- **Autenticación mediante firma digital del mensaje con criptografía asimétrica**
  - Una función pública y un par de claves, pública y privada inversas en un cuerpo, permiten la autenticación completa.

## Autenticación con sistemas simétricos

Si la clave de un sistema simétrico es segura, es decir no está en entredicho, podemos afirmar que, además de la confidencialidad que nos entrega dicha cifra, se comprueban también simultáneamente la integridad del mensaje y autenticidad del emisor, en tanto que sólo el usuario emisor (en quien se confía por el modelo, tipo de cifra y en su clave pública) puede generar ese mensaje.

---

Con los sistemas de cifra simétricos no podremos realizar una autenticación fácil y completa: emisor y mensaje. Veremos un procedimiento algo complejo de autenticación con cifra simétrica como es el caso de Kerberos y un esquema de firma digital.



# Los problemas de la autenticación simétrica

No obstante, subyacen los problemas característicos de un criptosistema: **¿cómo asegurar que la clave simétrica entre emisor y receptor es segura?** o lo que es lo mismo, **¿cómo intercambiar claves de forma segura?**

El intercambio de claves de forma segura ya hemos visto que se logra eficientemente sólo a través de sistemas asimétricos. Las herramientas más usuales para autenticación serán los códigos de autenticación de mensajes MACs y el sistema Kerberos con cifras simétricas. Kerberos también permite el intercambio seguro de una clave de sesión aunque es más complejo y pesado que el algoritmo de Diffie y Hellman.

## Autenticación con MAC o checksum

- Los dos extremos de la comunicación **A** y **B** comparten una única clave secreta que no está en entredicho.
- El MAC o checksum será una función que se aplica al mensaje  $M$  junto a una clave secreta  $K$  de la forma  $C_K(M)$ .
- **A** envía el mensaje en claro y el Message Authentication Code (MAC) o Checksum  $C_K(M)$  a **B**.
- **Integridad**: el receptor **B** puede estar seguro de que nadie ha modificado el mensaje durante la transmisión pues el valor  $C_K(M)$  en destino coincide con el enviado por **A**.
- **Autenticación**: como solamente el emisor **A** y el receptor **B** comparten la clave secreta  $K$ , se asegura la autenticación de ambos usuarios. Por lo tanto, la condición a cumplir es que la clave  $K$  debe ser única y segura.

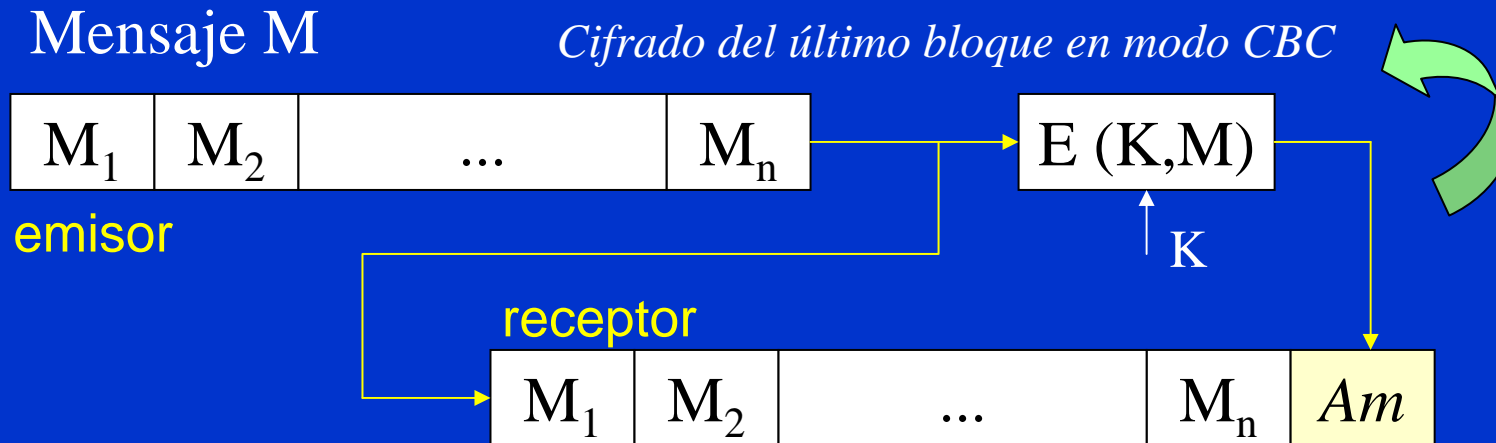
# Message Authentication Code con DES

- Se inserta un código al final del mensaje  $M$  transmitido en claro, consistente en la cifra con la clave secreta de los últimos bytes del texto, por ejemplo 64 bits de DES o bien usando Triple DES.
- En destino, con el mismo algoritmo y la clave secreta, se realiza la cifra y se comparan estos últimos bloques.
- Como la cifra es CBC, encadenamiento de bloques cifrados, esos bytes cifrados dependen de **todo** el mensaje por lo que cualquier modificación será detectada al no coincidir los resultados del cifrado de  $M$  en emisión y recepción.

<http://www.faqs.org/rfcs/rfc3537.html>



# Esquema de autenticación MAC con DES



$E(K, M) = Am \in M_R$      $M_R$ : espacio de marcas de autenticación

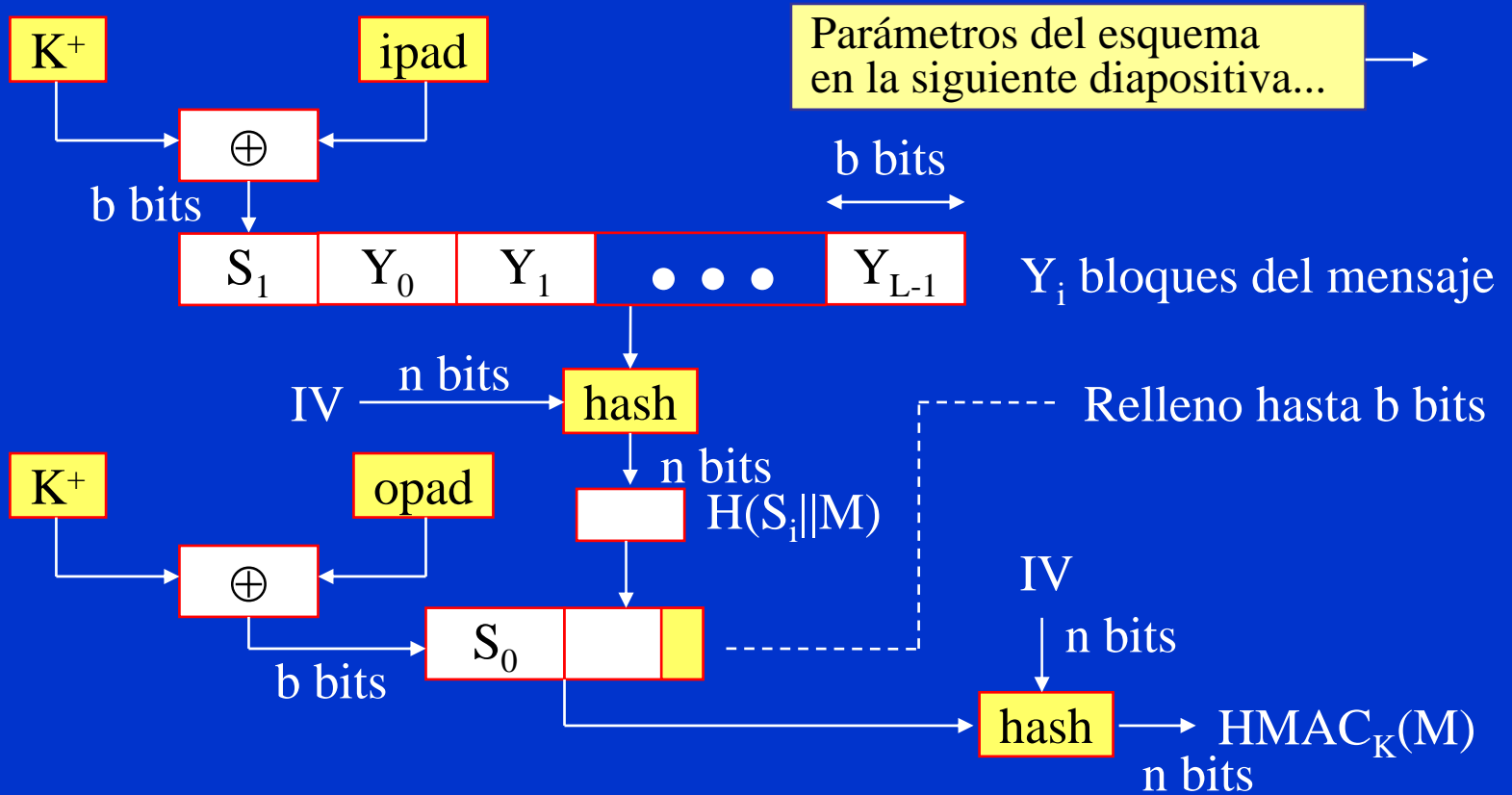
¿Cuáles son las debilidades de este modelo?

- La Marca  $Am$  son 16, 32 ó 64 bits.
- Es un tamaño muy pequeño y podría dar lugar a colisiones: mensajes distintos con iguales MACs.

# Autenticación con funciones hash HMAC

- Las funciones hash vistas (MD5, SHA-1, etc.) no han sido diseñadas para la autenticación al carecer de clave secreta.
- No obstante, son interesantes puesto que su velocidad es mayor que muchos cifradores de bloque, su código fuente es abierto y sus propiedades y funcionamiento son muy conocidos.
- La RFC 2104 propone el uso de una autenticación en entornos seguros como SSL mediante una operación MAC en la que intervenga una función hash: su nombre es HMAC.
- HMAC usa entonces una función hash (MD5, SHA-1, etc.) como una caja negra, una clave  $K$  en lo posible mayor que el tamaño del hash en bits, una función xor y operaciones de relleno de bits, tal como se muestra en el siguiente esquema.

# Esquema de HMAC



<http://csrc.nist.gov/publications/fips/fips198/fips-198a.pdf>



# Parámetros, valores típicos y salida HMAC

M = mensaje de entrada incluyendo el relleno.

H = alguna función hash como MD5 (128 bits) o SHA-1 (160 bits).

$Y_i$  = bloque  $i$ ésimo de M.

L = número de bloques en M.

b = número de bits en cada bloque (512).

n = longitud del resumen del hash ocupado en el sistema (128 / 160 bits).

K = clave secreta (160 bits) aunque se recomienda sea mayor que n. Si la clave K es mayor que n, esta clave se hace pasar antes por la función hash para reducirla a una clave de n bits.

$K^+$  = clave K con ceros añadidos a la izquierda para alcanzar b bits.

ipad = 00110110 octeto repetido b/8 (64) veces.

opad = 01011010 octeto repetido b/8 (64) veces.

Salida HMAC:  $HMAC_K = h\{(K^+ \oplus opad) \parallel h[(K^+ \oplus ipad) \parallel M]\}$

# Operaciones y seguridad de HMAC

- Añadir ceros a la izquierda de  $K$  hasta obtener  $K^+$ , una cadena de  $b$  bits.
- Sumar or exclusivo  $K^+$  con la cadena  $ipad$  para obtener  $S_1$  de  $b$  bits.
- Añadir a  $S_1$  el mensaje  $M$  como bloques  $Y_i$  de  $b$  bits cada uno.
- Aplicar la función hash elegida a la cadena de bits antes formada, con el vector inicial  $IV$  de  $n$  bits para generar un hash de  $n$  bits.
- Sumar or exclusivo  $K^+$  con la cadena  $opad$  para obtener  $S_0$  de  $b$  bits.
- Añadir a  $S_0$  el hash anterior, rellenando este último hasta  $b$  bits.
- Aplicar la función hash elegida a los dos bloques de  $b$  bits generados en el paso anterior, con vector  $IV$  de  $n$  bits para generar un hash de  $n$  bits.
- El resultado de este último hash es el HMAC del mensaje  $M$  con la clave  $K$ , es decir  $HMAC_K(M)$ .

Aunque la seguridad de HMAC está directamente relacionada con el hash utilizado, el modelo presentado no es seguro. Hay otras configuraciones más desarrolladas que mejoran esta característica.



# Autenticación con cifra simétrica y un KDC

- Utilización de un KDC (**K**ey **D**istribution **C**entre) o Distribuidor de Claves de Confianza, que comparte una clave maestra con los clientes.
- Cada parte, usuario o entidad de la red comparte una clave secreta y única o clave maestra con el KDC.
- El KDC se ocupa de distribuir una clave de sesión que va a ser utilizada en la conexión entre dos partes.
- La clave de sesión se protege con la clave maestra de los participantes de una comunicación.
- Entre los esquemas propuestos están el de Needham-Schroeder y el de Denning-Sacco. Uno de los más conocidos y populares será Kerberos.

<http://www.lsv.ens-cachan.fr/spore/nssk.html>



<http://www.isi.edu/gost/publications/kerberos-neuman-tso.html>



# Propuesta de Needham y Schroeder (1978)

1. **A** → KDC:  $ID_A || ID_B || N_1$
2. KDC → **A**:  $E_{KA}[K_S || ID_B || N_1 || E_{KB}[K_S || ID_A]]$
3. **A** → **B**:  $E_{KB}[K_S || ID_A]$
4. **B** → **A**:  $E_{KS}[N_2]$
5. **A** → **B**:  $E_{KS}[f(N_2)]$

$f(N_2)$  es una función conocida por **A** y **B**.

$ID_A$  = Nombre de **A** o identificador de **A**

$ID_B$  = Nombre de **B** o identificador de **B**

$N_X$  = Valor nonce

$K_S$  = Clave de sesión

$E_{KX}$  = Cifrado con clave secreta de **X**

$K_X$  = Clave secreta de **X** (**A** o **B**)

¿Qué sucede si no se realizan los pasos 4 y 5?


Un intruso **C** podría capturar el mensaje en el paso 3 y repetirlo, interfiriendo así las operaciones de **B**.

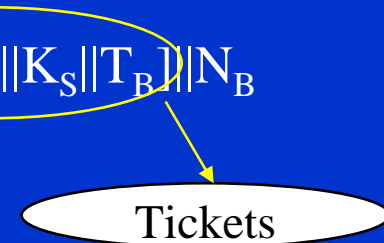


## Más debilidades de la propuesta N-S

- Algo más improbable es que un intruso **C** tenga una clave de sesión antigua y repita el mensaje del paso 3 enviado a **B**, quien no tiene porqué recordar todas las claves de sesión usadas previamente con **A**.
- Si **C** captura el mensaje del paso 4, podrá suplantar la respuesta de **A** del paso 5, enviando a **B** mensajes que parecen venir de **A** con clave de sesión autenticada.
- Denning propone la inclusión de un valor timestamp en los pasos 2 y 3. La utilización de timestamps requiere la sincronización de relojes, pero la utilización de valores nonce es también vulnerable. Se propone como solución óptima dar una duración a la clave de sesión (tickets).

# Solución propuesta por Denning y tickets

1.	<b>A</b> → KDC:	$ID_A    ID_B$	<b>Denning</b>  
2.	KDC → <b>A</b> :	$E_{KA}[K_S    ID_B    T    E_{KB}[K_S    ID_A    T]]$	
3.	<b>A</b> → <b>B</b> :	$E_{KB}[K_S    ID_A    T]$	
4.	<b>B</b> → <b>A</b> :	$E_{KS}[N_1]$	
5.	<b>A</b> → <b>B</b> :	$E_{KS}[f(N_1)]$	

1.	<b>A</b> → <b>B</b> :	$ID_A    N_A$	<b>Tickets</b>  
2.	<b>B</b> → KDC:	$ID_B    N_B    E_{KB}[ID_A    N_A    T_B]$	
3.	KDC → <b>A</b> :	$E_{KA}[ID_B    N_A    K_S    T_B]    (E_{KB}[ID_A    K_S    T_B])    N_B$	
4.	<b>A</b> → <b>B</b> :	$E_{KB}[ID_A    K_S    T_B]    E_{KS}[N_B]$	

Solución al problema del timestamp

## Autenticación en un sentido

- El correo electrónico es un ejemplo de aplicación que requiere este tipo de autenticación:
  - No es necesario que el emisor y el receptor estén conectados al mismo tiempo.
  - El receptor necesita confirmar de alguna forma que el emisor del mensaje es quien dice ser.

1.  $A \rightarrow KDC:$   $ID_A || ID_B || N_1$
2.  $KDC \rightarrow A:$   $E_{KA}[K_S || ID_B || N_1 || E_{KB}[K_S || ID_A]]$
3.  $A \rightarrow B:$   $E_{KB}[K_S || ID_A] || E_{KS}[M]$

Se garantiza así que sólo el receptor puede leer el mensaje y autentica además al emisor. Es vulnerable a repeticiones.

# Autenticación con Kerberos



Kerberos fue un proyecto desarrollado en 1988 por el Massachusetts Institute of Technology MIT, siendo su objetivo la implementación de un servicio de autenticación.



- ❖ Perro de tres cabezas y cola de serpiente según mitología griega, guardián de la entrada del Templo de Hades.
- ❖ Tres componentes guardarán esa puerta: Autenticación, Contabilidad y Auditoría. Las dos últimas cabezas del proyecto nunca han sido implementadas.

<http://web.mit.edu/kerberos/www/>



# Características y fases de Kerberos

- Está basado en el protocolo de distribución de claves de Needham & Schroeder.
- Usa un intercambio de claves con una tercera parte de confianza.
- Fases de autenticación:
  - Obtención de credenciales
    - Tickets
    - Autenticadores
  - Petición de autenticación frente un servicio.
  - Presentación de las credenciales al servidor final.

# Elementos del diálogo de autenticación

- **Usuario, cliente y servidor:** persona o máquina que necesita autenticarse en la red.
- **Principal:** entidad o cliente que usa Kerberos y que ha sido previamente autenticado por un servidor de autenticación.
- **Servicio y servidor:** servicio es una entidad abstracta (por ejemplo correo) y servidor el proceso que lo ejecuta.
- **Clave y password:** función aplicada a la clave de usuario.
- **Credenciales:** lo que se utiliza para autenticarse.
- **Maestros y esclavos:** la máquina que hospeda la base de datos es el master y esclavos son las máquinas que poseen copias de ésta.
- **Dominio:** nombre de entidad administrativa que mantiene los datos de autenticación.



## El ticket en la credencial de Kerberos

- Existen dos tipos de credenciales utilizadas en el modelo de Kerberos: **tickets** y **autenticadores**.
- Es necesario un ticket para pasar de una forma segura la identidad del cliente entre el servidor de autenticación y el servidor final.

$$[s, c, \text{addr}, \text{timestamp}, \text{life}, K_{S,C}]K_S$$

s = nombre del servidor

c = nombre del cliente

addr = dirección Internet del cliente

timestamp = fecha de iniciación del ticket

life = duración

$K_S$  = clave de sesión aleatoria

# El autenticador en la credencial de Kerberos

- El autenticador contiene información adicional que, comparada con el ticket, prueba que el cliente que presenta el ticket es el mismo a quien se le entregó,

$$[c, \text{addr}, \text{timestamp}]K_{S,C}$$

$c$  = nombre del cliente

$\text{addr}$  = dirección Internet de la estación de trabajo

$\text{timestamp}$  = fecha de iniciación del ticket

$K_{S,C}$  = clave de sesión que es parte del ticket

Los objetivos de las credenciales son minimizar el número de veces que un usuario tiene que introducir la password así como eliminar el problema de enviar la password en texto plano por la red.

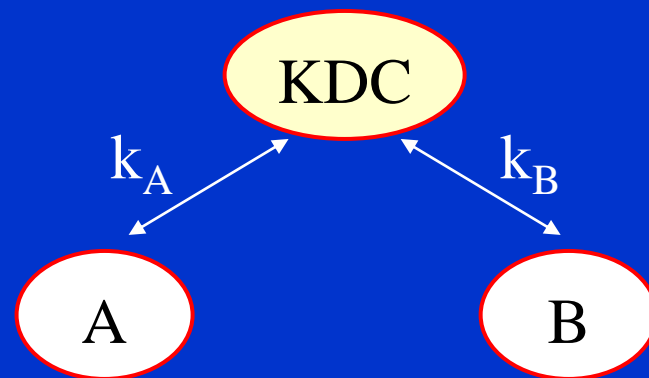
# Ejemplo de autenticación con Kerberos (1)

Se autenticará el usuario **A** ante el usuario **B** teniendo como tercera parte de confianza al centro KDC. Usarán el algoritmo DES.

Paso 1. **A** pide una credencial a KDC:

$A \rightarrow KDC \Rightarrow ID(A), ID(B), NA$

Envía el número aleatorio  $NA$  para que nadie pueda suplantar su identidad.



Paso 2. KDC genera una clave de sesión  $K_S$  con período de validez  $L$  y una credencial  $c$  para el usuario **B**:

$m = E_{k_A}[K_S, L, NA, ID(B)]; \quad c = E_{k_B}[K_S, L, ID(A)]$

$KDC \rightarrow A \Rightarrow (m, c)$

## Ejemplo de autenticación con Kerberos (2)

Paso 3. **A** descifra con su clave  $k_A$  el valor  $m$ :

$$D_{k_A}[E_{k_A}[K_S, L, NA, ID(B)]] = K_S, L, NA, ID(B)$$

Y luego con la clave de sesión entregada por KDC generará un autenticador  $a$  para el usuario **B** junto con un sello temporal  $T_A$ .

$$a = E_{K_S}[ID(A), T_A]$$

**A**  $\rightarrow$  **B**  $\Rightarrow (c, a)$

Paso 4. **B** descifra con su clave  $k_B$   $c$ :

$$D_{k_B}[E_{k_B}[K_S, L, ID(A)]] = K_S, L, ID(A)$$

Ahora que tiene  $K_S$  descifra el valor  $a$ :

$$D_{K_S}[E_{K_S}[ID(A), T_A]] = ID(A), T_A$$

Comprueba que coinciden los identificadores  $ID(A)$  del usuario **A** y que  $T_A$  está dentro del rango de validez  $L$  dado por KDC.

## Ejemplo de autenticación con Kerberos (3)

Paso 5. **B** calcula una función acordada con **A** por ejemplo  $(T_A + 1)$  y con la clave de sesión  $K_S$  se lo envía cifrado:

$$\mathbf{B} \rightarrow \mathbf{A} \quad \Rightarrow \mathbf{h} = \mathbf{E}_{K_S}[T_A + 1]$$

Paso 6. **A** descifra  $h$  con la clave de sesión  $K_S$ :

$$\mathbf{D}_{K_S}[\mathbf{E}_{K_S}[T_A + 1]] = T_A + 1$$

Comprueba que coincide con lo acordado lo que le demuestra que **B** ha recibido correctamente la clave de sesión  $K_S$ .

La importancia del valor de  $T_A$  es que permitirá más autenticaciones dentro del período de validez  $L$  sin que sea necesario la intervención de la tercera parte de confianza KDC.

## Resumen del ejemplo de autenticación

- Los valores de **c** y **m** aseguran la confidencialidad en la transmisión de la clave de sesión  $K_S$ .
- Los valores de **a** y **h** aseguran la confirmación de la recepción correcta de la clave de sesión  $K_S$  por **B**.
- En este protocolo sólo se autentica **A** ante **B**.
- Existen extensiones del algoritmo que permiten una autenticación doble entre **A** y **B**.
- Otra opción es que los usuarios **A** y **B** compartan una clave  $K_S$  sin que su valor sea conocido por KDC.

<http://www.mug.org.ar/Infraestructura/ArticInfraestructura/300.aspx>



## El hash en la autenticación asimétrica

- $h(M)$  es el resultado de un algoritmo que con una entrada  $M$  de cualquier tamaño, produce salida de tamaño fijo.
- El emisor **A** envía el mensaje  $M$  y la función hash  $h(M)$  a **B**, quien aplica la misma función al mensaje  $M'$  recibido. Si los mensajes son iguales entonces se asegura que los hash también son iguales. No obstante, el hecho de que los hash coincidan no significa que los mensajes  $M$  y  $M'$  sean iguales, como ya hemos visto en un capítulo anterior.
- **Integridad**: el receptor **B** puede confiar en que nadie ha modificado el mensaje durante la transmisión pues el valor  $h(M)$  en destino coincide con el enviado por **A**.
- **Autenticación**: es imposible la autenticación de usuario, salvo que se use un modelo con clave tipo HMAC ya visto.

# Características de una firma digital

Requisitos de la firma digital:

- a) Debe ser fácil de generar.
- b) Será irrevocable, no rechazable por su propietario.
- c) Será única, sólo posible de generar por su propietario.
- d) Será fácil de autenticar o reconocer por su propietario y los usuarios receptores.
- e) Debe depender del mensaje y del autor. →

Son condiciones mucho más fuertes que las de una firma manuscrita...

Esta última propiedad es muy importante pues protege ante la falsificación de los mensajes



## Firmas digitales simétricas

- ✓ Su uso usará una única clave secreta. Luego, persiste el problema de comunicar la clave entre los comunicantes.
- ✓ Normalmente usarán la función XOR para realizar la transformación.
- ✓ Podremos firmar mediante cifra en flujo y en bloque.
- ✓ Son muy rápidas por el tipo de cifra o bien por el uso de bloques más pequeños que en otros tipos de firma.
- ✓ **Algoritmos de firma:** Rabin, Lamport-Diffie, Desmedt, Matyas-Meyer basada en el DES ...

<http://research.microsoft.com/users/lamport/>



[http://www.deas.harvard.edu/ourfaculty/profile/Michael\\_Rabin](http://www.deas.harvard.edu/ourfaculty/profile/Michael_Rabin)



# Firma digital de Desmedt

- Propuesta por Yvo G. Desmedt en 1985.
- Se basa en el cifrado de Vernam.
- **Elementos:** parámetro  $q$  en bits, un mensaje  $M$  de longitud  $n$  bits y una clave  $H$  de  $n*q$  bits:
- $H = \{ [h_1^{(1)}, \dots, h_1^{(q)}], \dots, [h_n^{(1)}, \dots, h_n^{(q)}] \}$ 
  - $\forall i$  se cumplirá que  $[h_i^{(1)}, \dots, h_i^{(q)}] \neq (0, \dots, 0)$
- **Algoritmo:**  $r_j = \bigoplus_{i=1}^n m_i * h_i^{(j)} = m_1 * h_1^{(j)} \oplus \dots \oplus m_n * h_n^{(j)}$  (con  $j = 1, \dots, q$ )
- **Verificación de la firma:** con el mensaje recibido se realiza el mismo proceso del firmado hecho en emisión y se comprueba que se obtiene una firma idéntica.

<http://www.cs.fsu.edu/~desmedt/>

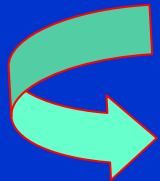


## Ejemplo de Firma digital de Desmedt

- Sea: mensaje  $M = 11011$  de  $n = 5$  bits, el parámetro  $q$  de 3 bits y la clave aleatoria  $H = (111, 100, 110, 001, 111)$ .
  - $r_1 = 1*1 \oplus 1*1 \oplus 0*1 \oplus 1*0 \oplus 1*1 = 1$
  - $r_2 = 1*1 \oplus 1*0 \oplus 0*1 \oplus 1*0 \oplus 1*1 = 0$
  - $r_3 = 1*1 \oplus 1*0 \oplus 0*0 \oplus 1*1 \oplus 1*1 = 1$
- Firma Desmedt: 101
- Como el receptor tiene esa clave  $H$ , que realiza la misma operación en destino y compara ambas firmas.
- ✓ Como son sistemas sin uso actual, sólo se muestra aquí este sencillo esquema. El lector interesado encontrará una amplia información en el libro de Criptografía Digital de José Pastor y Miguel Angel Sarasa que se menciona en el capítulo 21 de este libro electrónico.

# Autenticación con sistemas asimétricos

Al existir una clave pública y otra privada que son inversas, se autentica el mensaje y al emisor.



Permite la **firma digital**, única para cada mensaje

## Problema:

Los sistemas de cifra asimétricos son muy lentos y el mensaje podría tener miles o millones de bytes ...

## Solución:

Se genera un resumen del mensaje, representativo del mismo, con una función hash imposible de invertir. La función hash comprime un mensaje de longitud variable a uno de longitud fija y pequeña.

# Firma digital RSA de A hacia B

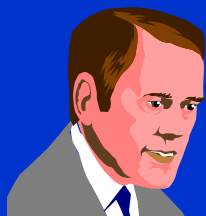
Clave Pública ( $n_A, e_A$ ) Clave Privada ( $d_A$ )



Adela

**Algoritmo:** Rúbrica:  $r_A h(M) = h(M)^{d_A} \text{ mod } n_A$

A envía el mensaje M en claro (o cifrado) al destinatario B junto a la rúbrica:  $\{M, r_A h(M)\}$



Benito

El destinatario B tiene la clave pública  $e_A, n_A$  de A y descifra  $r_A h(M) \Rightarrow \{(h(M)^{d_A})^{e_A} \text{ mod } n_A\}$  obteniendo así  $h(M)$ . Como recibe el mensaje  $M'$ , calcula la función hash  $h(M')$  y compara:

Si  $h(M') = h(M)$  se acepta la firma.



[http://www.enstimac.fr/Perl/perl5.6.1/site\\_perl/5.6.1/Crypt/RSA.html](http://www.enstimac.fr/Perl/perl5.6.1/site_perl/5.6.1/Crypt/RSA.html)



## Valores y tamaños típicos de firmas

- En los siguientes ejemplos, por limitación del tamaño de los primos elegidos, se firmarán sólo bloques de una cantidad determinada de bits en función del cuerpo de trabajo.
- No obstante, en los sistemas reales esto no es así puesto que las funciones hash ya vistas entregarán -por lo general- resúmenes comprendidos entre 128 y 160 bits y, por otra parte, el cuerpo de trabajo de la cifra asimétrica para la firma digital será como mínimo de 1.024 bits.
- Por lo tanto el resumen que se firma es menor que el cuerpo de cifra o, lo que es lo mismo, el número que se cifra será parte del conjunto de restos de ese grupo.

# Ejemplo de firma digital RSA (B → A)



Benito

Hola. Te envío el documento. Saludos, Beni.



Adela

Sea  $h(M) = F3A9$  (16 bits)

**Claves Benito**  
 $n_B = 65.669$   
 $e_B = 35, d_B = 53.771$

$2^{16} < 65.669 < 2^{17}$   
 Forzaremos firmar bloques de 16 bits

**Claves Adela**  
 $n_A = 66.331$   
 $e_A = 25, d_A = 18.377$

## Firma

$$h(M) = F3A9_{16} = 62.377_{10}$$

$$r_{h(M)} = h(M)^{d_B} \bmod n_B$$

$$r_{h(M)} = 62.377^{53.771} \bmod 65.669 = 24.622$$

Benito envía el par  $(M, r) = (M, 24.622)$

Nota: los primos que usa Benito son (97, 677) y Adela (113, 587)

# Comprobación la firma RSA por A



Benito

## Claves Benito

$$n_B = 65.669$$

$$e_B = 35, d_B = 53.771$$

## Claves Adela

$$n_A = 66.331$$

$$e_A = 25, d_A = 18.377$$



Adela

Teníamos que:  $h(M) = F3A9_{16} = 62.377_{10}$

$$r_{h(M)} = h(M)^{d_B} \bmod n_B \quad r_{h(M)} = 62.377^{53.771} \bmod 65.669 = 24.622$$

Benito había enviado a Adela el par  $(M, r) = (M, 24.622)$

Adela recibe un mensaje  $M'$  junto con una rúbrica  $r = 24.622$ :

- Calcula  $r^{e_B} \bmod n_B = 24.622^{35} \bmod 65.669 = 62.377$ .
- Calcula el resumen de  $M'$  es decir  $h(M')$  y lo compara con  $h(M)$ .
- Si los mensajes  $M$  y  $M'$  son iguales, entonces  $h(M) = h(M')$  y se acepta la firma como válida.
- **Recuerde:** El hecho de que  $h(M) = h(M')$  **no implica** que  $M = M'$ .



# Posible vulnerabilidad de la firma RSA

Al trabajar en un grupo multiplicativo, se da el siguiente escenario:

Si se conoce la firma de dos mensajes  $M_1$  y  $M_2$ , se puede firmar un tercer mensaje  $M_3$  producto de los dos anteriores sin necesidad de conocer la clave privada del firmante.

Sean las claves del firmante  $e$ ,  $d$  y  $n = p \cdot q$ . Luego:

$$r_{M_1} = M_1^d \bmod n \quad \text{y} \quad r_{M_2} = M_2^d \bmod n \quad \text{Si ahora } M_3 = M_1 M_2:$$

$$r_{M_3} = (M_1 M_2)^d \bmod n = M_1^d M_2^d \bmod n = r_{M_1} r_{M_2} \bmod n$$

**Comentario:** si la firma se realiza sobre un texto es prácticamente imposible que el producto de dos textos dé un nuevo texto con sentido. No obstante, sabemos que este tipo de firmas asimétricas se realizan sobre un número resultado de un hash por lo que podría darse una situación así, aunque por el tamaño del hash sería bastante poco probable este tipo de colisión.

## Ejemplo de firma RSA con colisión

Sean las claves  $n = 15.833 = 71 \cdot 223$ ,  $e = 17$ ,  $d = 7.313$ .

Sea  $M_1 = 8.643$  y  $M_2 = 10.014$  (ambos elementos de  $n$ )

Firma de  $M_1$ :  $r_{M_1} = 8.643^{7.313} \bmod 15.833 = 840$

Firma de  $M_2$ :  $r_{M_2} = 10.014^{7.313} \bmod 15.833 = 5.049$

Sea  $M_3 = M_1 M_2 = 8.643 \cdot 10.014 \bmod 15.833 = 7.824$

Firma de  $M_3$ :  $r_{M_3} = 7.824^{7.313} \bmod 15.833 = 13.749$

... y  $r_{M_1} r_{M_2} = 840 \cdot 5.049 \bmod 15.833 = 13.749$



Y. Desmedt, W. de Jonge y D. Chaum presentan en el año 1986 una variante de la firma RSA que evita este problema de la multiplicatividad.

# Firma digital ElGamal de A hacia B



Adela

ElGamal: El usuario A generaba un número aleatorio  $a$  (clave privada) del cuerpo  $p$ . La clave pública es  $\alpha^a \text{ mod } p$ , con  $\alpha$  generador.

## Algoritmo de firma:

Firma:  $(r, s)$

1° El usuario **A** genera un número aleatorio  $h$ , que será primo relativo con  $\phi(p)$ :  $h / \text{mcd} \{h, \phi(p)\} = 1$

2° Calcula  $h^{-1} = \text{inv} \{h, \phi(p)\}$

$$M = a*r + h*s \text{ mod } \phi(p) \quad \therefore$$

$$s = (M - a*r) * \text{inv}[h, \phi(p)] \text{ mod } \phi(p)$$

3° Calcula  $r = \alpha^h \text{ mod } p$

4° Resuelve la siguiente congruencia: \_\_\_\_\_

[http://www.math.clemson.edu/faculty/Gao/crypto\\_mod/node5.html](http://www.math.clemson.edu/faculty/Gao/crypto_mod/node5.html)



# Comprobación de firma ElGamal por B

## Algoritmo comprobación de firma

1° El usuario **B** recibe el par  $(r, s)$  y calcula:

$$r^s \bmod p \quad \text{y} \quad (\alpha^a)^r \bmod p$$

2° Calcula  $k = [(\alpha^a)^r * r^s] \bmod p$

Como  $r$  era igual a  $\alpha^h \bmod p$  entonces:

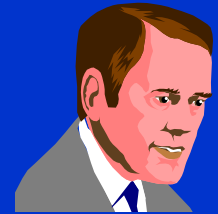
$$k = [(\alpha^{ar} * \alpha^{hs}) \bmod p = \alpha^{(ar + hs)} \bmod p = \alpha^\beta \bmod p$$

3° Como  $M = (a*r + h*s) \bmod \phi(p)$  y  $\alpha$  es una **raíz primitiva** de  $p$  (esto debe cumplirse) entonces:

$$\alpha^\beta = \alpha^\gamma \quad \text{ssi} \quad \beta = \gamma \bmod (p-1)$$

4° Comprueba que  $k = \alpha^M \bmod p \longrightarrow$

Si  $k = [(\alpha^a)^r * r^s] \bmod p$   
es igual a  $\alpha^M \bmod p \dots$



**Benito**

Conoce:  $p$  y  $(\alpha^a) \bmod p$

Se acepta la firma

# Ejemplo de firma ElGamal (B → A)



Benito

¡Hola otra vez! Soy Benito de nuevo. Saludos.



Adela

Sea  $h(M) = A69B$  (16 bits)

**Claves Benito**  
 $p_B = 79.903$     $\alpha = 10$   
 $\alpha^b \text{ mod } p = 3.631$   
 **$b = 20$     $h = 31$**

$2^{16} < 79.903 < 2^{17}$   
 Forzaremos firmar bloques de 16 bits

## Firma

- 1)  $h^{-1} = \text{inv}[h, \phi(p)] = \text{inv}(31, 79.902) = 5.155$
- 2)  $r = \alpha^h \text{ mod } p = 10^{31} \text{ mod } 79.903 = 11.755$
- 3)  $s = [h(M) - b*r]*[\text{inv}(h, \phi(p))] \text{ mod } \phi(p)$       $h(M) = A69B_{16} = 42.651_{10}$
- 4)  $s = [42.651 - 20*11.755]*5.155 \text{ mod } 79.902$
- 5)  $s = 68.539$     Luego, la firma será  $\longrightarrow$   $(r, s) = (11.755, 68.539)$

# Comprobación de firma ElGamal por A



Benito

## Claves Benito

$$p_B = 79.903 \quad \alpha = 10$$

$$\alpha^b \text{ mod } p = 3.631$$

$$b = 20 \quad h = 31$$

$$h(M) = A69B = 42.651$$



Adela

Adela recibe de Benito el par  $(r, s) = (11.755, 68.539)$

## Comprobación de la firma:

$$1) r^s \text{ mod } p = 11.755^{68.539} \text{ mod } 79.903 = 66.404$$

$$2) (\alpha^b)^r \text{ mod } p = 3.631^{11.755} \text{ mod } 79.903 = 12.023$$

$$3) (\alpha^b)^r * r^s \text{ mod } p = (12.023 * 66.404) \text{ mod } 79.903 = 64.419 = k$$

$$4) \alpha^{h(M)} \text{ mod } p = 10^{42.651} \text{ mod } 79.903 = 64.419$$

Como hay igualdad  
se acepta la firma



# Importancia de $\alpha$ en la firma de ElGamal



Benito

## Claves Benito

$$p_B = 79.903 \quad \alpha = 10$$

$$\alpha^b \text{ mod } p = 3.631$$

$$b = 20 \quad h = 31$$

$\alpha = 10$  es un generador del cuerpo  $p = 79.903$  puesto que:

$$p-1 = 79.902 = 2 \cdot 3^2 \cdot 23 \cdot 193$$

$$q_1 = 2; q_2 = 3; q_3 = 23; q_4 = 193$$

y se cumple  $10^{(p-1)/q_i} \text{ mod } p \neq 1$

$$10^{39.951} \text{ mod } 79.903 = 79.902$$

$$10^{26.634} \text{ mod } 79.903 = 71.324$$

$$10^{3.474} \text{ mod } 79.903 = 2.631$$

$$10^{414} \text{ mod } 79.903 = 41.829$$

Si se elige  $\alpha = 11$ , para el exponente 39.951 se obtiene un 1 por lo que no es raíz.

Compruebe que en este caso no nos servirá  $\alpha$  para la firma digital porque será **imposible** comprobarla mediante  $k = \alpha^M \text{ mod } p$ .

# Estándares de firma digital

El peor inconveniente de la firma propuesta por ElGamal es que duplica el tamaño del mensaje  $M$  al enviar un par  $(r, s)$  en  $Z_p$  y  $\phi(p)$ . No obstante, se solucionará con el algoritmo denominado DSS.

1991: National Institute of Standards and Technology (NIST) propone el DSA, Digital Signature Algorithm, una variante de los algoritmos de ElGamal y Schnoor.

1994: Se establece como estándar el DSA y se conoce como DSS, Digital Signature Standard.

1996: La administración de los Estados Unidos permite la exportación de Clipper 3.11 en donde viene inmerso el DSS, que usa una función hash de tipo SHS, Secure Hash Standard.

<http://www.itl.nist.gov/fipspubs/fip186.htm>





# Digital Signature Standard DSS

Parámetros públicos de la firma:

- Un número primo  $p$  ( $512 \text{ bits} < p < 1024 \text{ bits}$ )
- Un número primo  $q$  (160 bits) divisor de  $p-1$
- Un generador  $\alpha$  “de orden  $q$ ” del grupo  $p$



Generador de orden  $q$  es aquella raíz  $\alpha$  en el cuerpo  $Z_p$  de forma que  $q$  es un entero que verifica:

$$\alpha^q \text{ mod } p = 1$$

Así, para todo  $t$ :

$$\alpha^t = \alpha^{t \text{ (mod } q)} \text{ mod } p$$

**Elección de parámetros:** primero se busca el primo  $p$ , luego un primo  $q$  que sea divisor de  $(p-1)$  y luego un valor  $g$  en  $p$ , de forma que si  $\alpha = g^{(p-1)/q} \text{ mod } p \neq 1$ , éste será el generador... →

# La elección de los parámetros en DSS

- Elegir un número primo  $2^{159} \text{ bits} < q \leq 2^{160} \text{ bits}$ .
- Elegir  $0 \leq t \leq 8$  y encontrar un número primo  $p$  que esté en  $2^{511+64t} \leq p \leq 2^{512+64t}$  y que además  $q$  divida a  $(p-1)$ .
- Elegir un generador  $\alpha$  de orden  $q$  de la siguiente forma:
  - Elegir  $g$ , un elemento de  $p$ , de forma que se cumpla la condición  $\alpha = g^{(p-1)/q} \text{ mod } p \neq 1$ . Recuerde que esta raíz  $\alpha$  **no** será la misma que la conocida hasta ahora.
- Elegir como clave privada un valor aleatorio  $x$  dentro del cuerpo del primo  $q$ .
- Calcular la clave pública  $y = \alpha^x \text{ mod } p$ .
- Hacer público los parámetros  $q, p, \alpha, y$ .

## Generación de firma DSS ( $A \rightarrow B$ )

- Valores públicos de  $A$ : primos  $p$ ,  $q$  y el generador  $\alpha$
- Clave secreta de la firma:  $x$  ( $1 \leq x \leq q$ ) aleatorio
- Clave pública de la firma:  $y = \alpha^x \bmod p$
- Para encontrar  $\alpha$  y firmar un mensaje  $1 \leq M \leq p$ ,  $A$  ya ha elegido un valor aleatorio  $1 \leq k \leq q$

Luego el firmante  $A$  calcula:

- $r = (\alpha^k \bmod p) \bmod q$
- $s = [(M + x*r) * \text{inv}(k, q)] \bmod q$
- La firma digital de  $A$  sobre  $M$  será el par  $(r, s)$

## Comprobación de firma DSS por B

- **B** recibe el par  $(r, s)$
- Luego calcula:
  - $w = \text{inv}(s, q)$
  - $u = M * w \text{ mod } q$
  - $v = r * w \text{ mod } q$
- Comprueba que se cumple:
  - $r = (\alpha^u y^v \text{ mod } p) \text{ mod } q$
- Si se cumple, **B** acepta la firma como válida.

Observe que la comprobación de la firma se hace sobre **r**, un valor en el que no interviene el valor del mensaje  $M$  y que, además, la firma se hace ahora sobre valores de  $q$  y no de  $p$ .

La firma DSS tendrá un tamaño menor que  $q$  al reducirse  $(r, s)$  a dicho módulo, siendo  $q \ll p$ .

# Ejemplo de firma DSS (B → A)



**Benito**

Compruebe que son correctos  $p, q$ . Y que  $g = 19$  genera  $\alpha = 17$ .

## Firma

- 1)  $\text{inv}(k, q) = \text{inv}(12, 37) = 34$
- 2)  $r = (\alpha^k \bmod p) \bmod q = (17^{12} \bmod 223) \bmod 37 = 171 \bmod 37 = 23$
- 3)  $s = [h(M) + x * r] * [\text{inv}(k, q)] \bmod q = [30 + 25 * 23] * 34 \bmod 37 = 35$
- 4) La firma digital de  $h(M) = 30$  será:  $(r, s) = (23, 35)$
- 5) **Benito** transmite a **Adela** el bloque:  $(M, r, s) = (M, 23, 35)$

Hola Adela, soy Benito y firmo con DSS.

Sea  $h(M) = 30$  (un elemento de  $q_B$ )

## Claves Benito

$$p_B = 223; q_B = 37; \alpha = 17$$

$$y = \alpha^x \bmod p = 30$$

$$x = 25; k = 12$$

Como se firmará sobre valores de  $q_B$ , los mensajes a firmar serán  $0 \leq h(M) \leq 36$ .



**Adela**

# Comprobación de la firma DSS por A



Benito

## Claves Benito

$$p_B = 223; q_B = 37; \alpha = 17$$

$$y = \alpha^x \text{ mod } p = 30$$

$$x = 25; k = 12$$

Adela recibe el bloque:

$$(M, r, s) = (M, 23, 35)$$



Adela

## Comprobación de la firma

- 1)  $w = \text{inv}(s, q) = \text{inv}(35, 37) = 18$
- 2)  $u = h(M) * w \text{ mod } q = 30 * 18 \text{ mod } 37 = 22$
- 3)  $v = r * w \text{ mod } q = 23 * 18 \text{ mod } 37 = 7$
- 4)  $¿(\alpha^u y^v \text{ mod } p) \text{ mod } q = r ?$
- 5)  $[(17^{22} 30^7) \text{ mod } 223] \text{ mod } 37 = 23$

¿Igualdad?

En caso afirmativo,  
se acepta la firma

Y el tamaño será menor que  $q_B = 37$  es decir  $\ll p_B = 223$  que era precisamente el punto débil del sistema de ElGamal.

## Seguridad de los 160 bits de $q$

- Podríamos pensar que al bajar el número de bits de 1.024 en la firma de ElGamal a sólo 160 (el valor de  $q$ ) en la firma DSS la seguridad de dicha firma está comprometida.
- No obstante, la firma DSS tiene la misma fortaleza que la de ElGamal ya que  $q$  es un subgrupo de  $p$ . Eso quiere decir que para resolver el problema del logaritmo discreto en  $q$ , habrá que hacerlo obligatoriamente en  $p$ .
- Para evitar diversos ataques tanto en la firma ElGamal como en DSS, deberá firmarse siempre una función hash.
- DSS requiere el uso del hash  $h(M)$  sobre  $M$  con SHA-1.
- **Importante:** recuerde que se firma un hash SHA-1 de 160 bits en un cuerpo  $q$  también de 160 bits, aunque  $p$  sea de 1.024 bits.

<http://lists.gnupg.org/pipermail/gnupg-users/2000-August/006286.html>



## Mensajes que no tienen firma DSS

- No todos los valores  $h(M)$  podrán firmarse con DSS.
- Para comprobar la firma en recepción se calcula el valor  $w = \text{inv}(s, q)$ , donde  $s = [h(M) + b * r] * [\text{inv}(k, q)] \text{ mod } q$ . Luego, debe existir dicho inverso.
- Si  $s = 0$  no existe el inverso. Luego esta condición deberá comprobarse en emisión antes de proceder a la firma.
- No obstante, la probabilidad de que se dé esta situación en el cuerpo  $q$  de 160 bits será de uno en  $2^{160}$ .
- Así mismo, en emisión deberá verificarse que  $r \neq 0$ . En ambos casos se elegirá un nuevo valor de  $h$ .



# Ejemplo de mensaje sin firma por $s = 0$



Benito

Hola Adela, vamos a ver qué pasa ahora.

Sea ahora  $h(M) = 17$  (un elemento de  $q_B$ )



Adela

## Claves Benito

$$p_B = 223; q_B = 37; \alpha = 17$$

$$y = \alpha^x \text{ mod } p = 30$$

$$x = 25; k = 12$$

## Firma

- 1)  $\text{inv}(k, q) = \text{inv}(12, 37) = 34$
- 2)  $r = (\alpha^x \text{ mod } p) \text{ mod } q = (17^{12} \text{ mod } 223) \text{ mod } 37 = 171 \text{ mod } 37 = 23$
- 3)  $s = [h(M) + b \cdot r] \cdot [\text{inv}(k, q)] \text{ mod } q = [17 + 25 \cdot 23] \cdot 34 \text{ mod } 37 = 0$
- 4) La firma digital de  $h(M) = 17$  sería:  $(r, s) = (23, 0) \dots \text{☹}$

Como es obvio si  $h(M) = 17 + K \cdot q_B$ , es decir 17, 54, 91, 128, 165, 202 ... caemos en un cuerpo de equivalencia en mod  $q_B$  y se repite esta situación.

Fin del capítulo

# Firmas simétricas versus asimétricas

## SIMÉTRICAS

- **Claves:** una clave secreta para ambos comunicantes.
- **Cifrado:** en bloque y en flujo.
- **Tasas de cifra:** MB/seg.
- **Fiabilidad:** baja; necesario un intercambio previo de clave.
- **Vida de la clave:** efímera, one-time pad, segundos, minutos.
- **Algoritmos:** Rabin, Desmedt, Matyas-Meyer, Lamport-Diffie.

## ASIMÉTRICAS

- **Claves:** comunicantes con claves secreta y pública.
- **Cifrado:** sólo bloque.
- **Tasas de cifra:** KB/seg.
- **Fiabilidad:** alta; no necesita intercambio previo de clave.
- **Vida de la clave:** duradera, se repite, meses, un año.
- **Algoritmos:** RSA, ElGamal, DSS, Rabin, curvas elípticas.

Los sistemas de firma digital actuales son con clave pública.

## Cuestiones y ejercicios (1 de 2)

1. ¿Por qué cree que un escenario de integridad en la que hay siempre una tercera parte de confianza activa no sería adecuado en Internet?
2. Si una Autoridad de Certificación es la tercera parte de confianza, ¿actúa de forma activa o pasiva en la comunicación? Explíquelo.
3. ¿Qué importancia tiene la redundancia del lenguaje en un sistema de autenticación? ¿Qué sucedería si no hubiese esa redundancia?
4. Hacemos una autenticación de mensaje mediante un MAC en el que el algoritmo es DES. ¿Sería acertado usar modo ECB? ¿Por qué?
5. En un sistema de autenticación mediante Kerberos, ¿cómo sabe el que comienza el protocolo (A) que no hay un impostor que intenta suplantarle? ¿Cómo sabe el que comienza el protocolo (A) que el segundo usuario (B) o elemento del sistema ha recibido bien la clave de sesión entregada por el centro de distribución de claves KDC?

## Cuestiones y ejercicios (2 de 2)

6. ¿Cómo podríamos usar una función hash para comprobar que un archivo no se ha modificado o que está libre de virus?
7. Nos afirman que podemos autenticar un mensaje usando para ello sólo una función hash. ¿Es esto verdadero? ¿Por qué?
8. ¿Por qué se dice que una firma digital tiene condiciones más fuertes que una firma manuscrita?
9. ¿Por qué es importante que la firma digital dependa del mensaje?
10. Firme digitalmente con RSA el mensaje  $M = 121$  si los datos del firmante son  $p = 19$ ;  $q = 31$ ,  $d = 149$ . Compruebe la firma.
11. Con  $p = 331$  y clave privada 15, vamos a firmar digitalmente con ElGamal el mensaje  $M = 250$ . Para ello, elija los valores más pequeños posibles de los parámetros  $\alpha$  y  $h$ . Compruebe la firma.
12. Repita el ejercicio 10 con DSS y valores propuestos por Ud.

Use el portapapeles

## Prácticas del tema 16 (1/3)

Software ExpoCrip:

[http://www.criptored.upm.es/software/sw\\_m0011.htm](http://www.criptored.upm.es/software/sw_m0011.htm)



1. **FIRMA RSA.** Con la clave  $p = 3$ ,  $q = 11$ ,  $e = 3$ , compruebe que la firma de mensajes numéricos dentro del cuerpo  $\{0, 1, \dots, 32\}$  son todos valores distintos y que no puede haber firmas iguales para mensajes distintos.
2. Observe lo que pasa al firmar los mensajes 0, 1, 10, 11, 12, 21, 22, 23, 32.
3. ¿Qué sucede con estos mensajes si ahora se elige la clave  $e = 13$ ?
4. Con esta última clave firme el mensaje numérico  $N = 13$ .
5. Compruebe que la firma es correcta. ¿Cómo comprueba esto el programa?
6. Cambie el mensaje numérico a  $N = 35$ , vuelva a firmar y a comprobar la firma. ¿Por qué sucede esto ahora? ¿Qué pasa ahora si  $N = 2$ ?
7. Con esta última clave firme el mensaje hexadecimal  $N = 1F$ .
8. Con esta clave firme el mensaje  $M = \text{Hola}$ ; compruebe la firma. Con la ayuda del programa, vea cómo se generan los bloques para la firma.

Use el portapapeles

## Prácticas del tema 16 (2/3)

9. Para la clave  $p = 37$ ,  $q = 53$ ;  $e = 25$ , firme el mensaje  $M =$  **Por este trabajo le pagaremos 1.000,00 Euros**. Compruebe la firma.
10. Modifique el texto en claro y cambie **1.000,00** por **9.000,00**. Vuelva ahora a comprobar la firma y comente qué ha sucedido y porqué.
11. **FIRMA ELGAMAL**. Con la clave  $p = 19$ ,  $\alpha = 10$ ,  $x = 7$ ,  $k = 5$ , compruebe que las firmas de todos los valores de cuerpo  $\{0, 1, \dots, 18\}$  son valores distintos y no puede haber firmas iguales para mensajes distintos.
12. ¿Qué sucede si supone un generador  $\alpha = 4$ ? ¿Qué hace el programa?
13. ¿Se observa el mismo problema de mensajes no firmables que en RSA? ¿Existe algo que le llame la atención? Si no, vea la siguiente pregunta.
14. Genere claves de firma para  $p = 11$ ,  $p = 13$ ,  $p = 17$ ,  $p = 23$ ,  $p = 29$ ,  $p = 37$ ; y compruebe que la firma del mensaje 0 y  $p-1$  siempre es el mismo valor y que precisamente no aparece entre los restos de firma el valor  $p-1$ .
15. Con una de esas claves firme  $M =$  **Esta es otra firma**. Compruebe la firma.

Use el portapapeles

## Prácticas del tema 16 (3/3)

16. Para  $p = 472287102421$ ,  $\alpha = 33$ ,  $x = 31023$ ,  $k = 40981$ , firme y compruebe estos mensajes:  $N_{10} = 2001$ ,  $N_{16} = FD9C5$ ,  $M_{ASCII} = \text{Hola}$ , ¿qué tal?
17. **FIRMA DSS**. Para la clave DSS  $p = 19$ ,  $q = 3$ ,  $g = 13$ ,  $x = 2$ ,  $k = 2$ , firme todos los valores de cuerpo  $q \{0, 1, 2\}$  y compruebe que los valores de las firmas están siempre en el cuerpo de  $q$ . ¿Son todas distintas?
18. ¿Qué tipo de generador usa el programa? ¿Es un generador de  $p$  o de  $q$ ?
19. Repita el ejercicio anterior para  $p = 23$ ,  $q = 11$ ,  $g = 17$ ,  $x = 5$ ,  $k = 9$ .
20. Para la clave DSS  $p = 53$ ,  $q = 13$ ,  $g = 31$ ,  $x = 8$ ,  $k = 6$ , firme los primeros 7 mensajes numéricos  $N = \{0, 1, \dots, 6\}$ . Para cada una de ellos observe el seguimiento de la comprobación y compruebe manualmente al menos una.
21. Con esta clave firme el mensaje  $M = A$ . ¿Puede comprobar la firma? Repita para  $M = AB$ ,  $M = ABC$ . Repita para esos valores en hexadecimal.
22. Si  $p = 124540019$ ,  $q = 17389$ ,  $g = 110217528$ , ( $\alpha = 10083255$ ),  $k = 9557$ , y  $x = 12496$ , compruebe que la firma de  $M = 5246$ :  $(r, s) = (34, 13049)$ .  
Valores A. Menezes: <http://www.cacr.math.uwaterloo.ca/hac/about/chap11.pdf>.