Tema 2. Criptografía

Seguridad en Sistemas Informáticos

Septiembre-2018

Contenido

Criptosistemas: componentes y definiciones

Formalmente un criptosistema es una tupla de 5 componentes (M, C, K, E, D)

- M: espacio de mensajes (conjunto de todos los posibles *mensajes en claro*)
- C: espacio de cifrado (conjunto de todos los posibles *mensajes cifrados* o *criptogramas*)
- K: espacio de claves (conjunto de todos los posibles valores aptos como clave/s)
- $E: M \rightarrow C$: función de cifrado
  - algoritmo que transforma mensajes en claro en mensajes cifrados
  - tranformación gobernada por la correspondiente clave de cifrado $k \in K$
- $D: C \rightarrow M$: función de descifrado
  - algoritmo que transforma mensajes cifrados en mensajes en claro
  - dicha tranformación es gobernada por la correspondiente clave de descifrado $k \in K$
  - dependiendo del esquema de cifrado (simétrico o asimétrico) las claves de cifrado y descifrado pueden ser la misma o no

Normalmente interesa que los criptosistemas sean reversibles

$$D_k(E_k(m)) = m \quad \forall m \in M$$

Principio de Kerckhoff (1883)

En criptosistemas modernos se asume que las funciones de cifrado y descifrado son conocidas (y analizables). El único secreto es la clave/s utilizada/s.

Criptosistemas: criptoanálisis

Criptoanálisis: conjunto de técnicas que intentan desvelar el secreto de un criptosistema

- **Objetivo:** descrifrar un mensaje cifrado y/o identificar la clave utilizada
- Normalmente el criptoanálisis se basa en encontrar *trazas* del mensaje original y/o de la clave utilizada en los textos cifrados.

Tipos:

- **Sólo texto cifrado**: criptoanalista conoce algoritmo de cifrado + texto cifrado
- **Texto claro conocido**: criptoanalista dispone además de pares de texto claro + texto cifrado
- **Texto claro seleccionado**: criptoanalista puede seleccionar el texto claro que será cifrado con la respectiva clave
- **Texto cifrado seleccionado**: criptoanalista puede seleccionar el texto cifrado que será descifrado con la respectiva clave
- **Texto seleccionado**: criptoanalista puede seleccionar cualqueir combinación de texto cifrado y texto en claro que precise

Siempre son posible ataques por fuerza bruta sobre el espacio de claves

Criptosistemas: propiedades

Criptosistemas incondicionalmente seguros

- Aquellos que garantizan que en el texto cifrado no existe información suficiente para determinar el texto en claro original
  - Son inmunes a análisis estadísisticos (no hay relación estadística entre texto en claro y texto cifrado)
  - Solo el cifrado **one time pad** verifica esta propiedad

Criptosistemas computacionalmente seguros

- Aquellos que garantizan uno o los dos siguientes criterios
  1. El coste de romper el texto cifrado excede el valor de la información cifrada
  2. El tiempo para romper el texto texto cifrado excede el tiempo de vida útil de la información cifrada

Servicios de seguridad

Servicios de seguridad soportados por técnicas criptográficas (No todas las técnicas criptográficas ofrecen todos los servicios de seguridad)

- **Confidencialidad:** convierte la información en ilegible excepto para las entidades autorizadas (disponen de la/s clave/s adecuadas)
- **Integridad:** los datos no han sido alterados de forma no autorizada desde que fueron creados/enviados/almacenados
- **Autenticación:** garantiza la identidad de la entidad que creo la información
- **Autorización:** verificada la identidad, se proporciona a la entidad una clave/password/token criptográfico que le permita su acceso a determinado recurso/activo
- **No repudio:** garantiza que el emisor/creador de una información no pueda negar haber realizado su envío/creación

Clasificación de los esquemas de cifrado (I)

Por el número de claves

- **Cifrado simétrico.** Algoritmos de cifrado y descrifrado usan una única clave secreta compartida por emisor y receptor
  - La clave debe mantenerse en secreto
  - **Punto débil:** intercambio/acuerdo seguro de clave entre emisor y receptor
    - También: limitaciones de escalabilidad
    - Con n participantes se deben acordar $\frac{n(n-1)}{2}$ claves secretas diferentes
- **Cifrado asimétrico.** Se usa una clave para el cifrado y otra distinta para el descrifrado
  - Ambas claves estás relacionadas y se generan de forma conjunta
  - Una de ellas (clave de cifrado) debe ser conocida por todos los posibles emisores (clave pública)
  - La otra (clave de descrifrado) sólo debe ser accesible a su propietario (clave privada)
  - **Punto débil:** difusión fiable de las claves públicas de los participantes
    - Vulnerable ante la falsifiacación de la clave pública (junto con su clave privada ”gemela” identifica a su propietario)

Clasificación de los esquemas de cifrado (II)

Por el tipo de operaciones realizadas

Aplicable sólo al cifrado simétrico (el cifrado asimétrico se basa en operaciones matemáticas sobre los datos)

- **Cifrado de sustitución.** Cada elemento del texto en claro (letra, byte, bloque, ...) es *mapeado* (transformado) en otro elemento (normalmente del mismo alfabeto) en el texto cifrado
  - Tranformación gobernada por la clave de cifrado
- **Cifrado de trasposición.** Los elementos del texto en claro (letra, byte, bloque, ...) se reorganizan cambiando su ubicación en el texto cifrado resultante
  - Permutaciones a realizar gobernadas por la clave de cifrado

Los algoritmos simétricos modernos combinan varias etapas de operaciones de sustitución y transposición/permutación aplicadas de forma secuencial (cifrado producto)

Clasificación de los esquemas de cifrado (III)

Por el modo en que se procesan los datos

- **Cifrador de bloque.** (*block cypher*) Realiza las transformaciones de cifrado/descrifrado sobre un bloque de elementos de tamaño fijo cada vez.
  - Produce un bloque completo de cada vez
  - Puede ser necesario incluir relleno (*padding*) en el bloque final para completar su tamaño
  - Los cifradores de bloque modernos usan tamaños de bloque de 64, 128 o 256 bits.
- **Cifrador de fujo.** (*stream cypher*) Procesa los elementos de entrada continuamente
  - No se espera a completar un bloque, se procesa bit a bit (cifradores modernos) o caracter a caracter (cifradores clásicos)
  - La mayor parte de cifradores de flujo modernos se basan en generadores de números aletorios y operaciones XOR a nivel de bits (tratan de emular el funcionamiento del cifrado *one time pad*)
  - Existen configuraciones/modos de cifrado que permiten a emular un cifrado de flujo usando cifradores de bloque

Criptografía clásica (I)

Anterior al desarrollo de la criptografía "'por computador"

- Todos los métodos criptográficos clásicos son simétricos (cifrado asimétrico surge a mediados de los 70)

Cifrado monoalfabético (Cifrado de sustitución)

- Cada símbolo del texto en claro se reemplaza siempre por el mismo símbolo en el texto cifrado
  - Reemplazo de símbolos gobernado por la clave
  - La "tabla de reemplazo" es siempre la misma para todos las posiciones del texto en claro
- Ejemplo: Cifrado del César
  - Reemplaza cada letra en claro por la que está 3 posiciones más adelante
  - `http://es.wikipedia.org/wiki/Cifrado_César`
  - `http://www.cryptool-online.org`
- Ataques
  - Algoritmo del César: sólo 26 claves posibles $\Rightarrow$ ataque por fuerza bruta sobre el espacio de claves
  - Cifrado monoalfabético general: (26! claves $\approx$ tablas de sustitución $\Rightarrow$ no posible fuerza bruta manual)

    Es posible estudio estadístico de frecuencias relativas de símbolos
    - Reemplazo de cada símbolo es siempre el mismo $\Rightarrow$ símbolos cifrados mantienen las frecuencias relativas de sus respectivos símbolos en claro

Criptografía clásica (II)

Cifrado polialfabético (Cifrado de sustitución)

- La sustitución a realizar sobre cada símbolo del texto en claro varía en función de su posición dentro del mensaje
  - La ”tabla de reemplazo” varía
  - Se pretende que un mismo símbolo en claro sea cifrado de formas disitintas
- Ejemplo 1: Cifrado de Vigenere
  - Utiliza 26 cifrados del César que se van alternando según indiquen las letras de la palabra clave
    - La palabra clave se repite tantas veces como sea necesario para ”cubrir” la totalidad del texto en claro
  - `http://es.wikipedia.org/wiki/Cifrado_de_Vigènere`
  - `http://www.cryptool-online.org`
  - Ataque: fuerza bruta sobre el tamaño de clave + análisis de frecuencias
    - Se repite la palabra clave $\Rightarrow$ si se sabe su longitud (k) se sabe cada cuantas posiciones se repiten los cifrados
    - Cada k posiciones se usa el mismo cifrado César $\Rightarrow$ posible realizar estudio estadístico

Criptografía clásica (III)

Cifrado polialfabético (2)

- Ejemplo 2: Cifrado *one time pad* (tb. cifrado de Vernam ó libreta de un sólo uso)
  - Evita la repetición de la clave usando una clave binaria aleatoria tan grande como el texto en claro
  - `http://es.wikipedia.org/wiki/One_Time_Pad`
  - Funcionamiento (cifrado)
    1. Convertir las letras del texto en claro a codificación binaria
    2. Generar una secuencia binaria aletoria del mismo tamaño que el texto en claro codificado en binario
    3. Calcular el XOR entre texto en claro binario y la clave aletoria
    4. Hacer llegar al destinatario de forma segura texto en claro + clave aletoria
    5. **Descrifrado:** XOR entre texto cifrado y clave aleatoria
  - Ventajas
    - Es incondicionalmente seguro si se verificada que la clave es
      - totalmente aleatoria
      - tan larga como el mensaje
      - de un sólo uso (una clave distinta para cada mensaje)

      No habrá traza estadística del texto en claro en el texto cifrado
  - Limitaciones
    - Díficil el intercambio de claves tan grandes y de un sólo uso
    - No trivial garantizar la perfecta aleatoriedad de la clave generada

Criptografía clásica (IV)

Cifrado Playfair (Cifrado de sustiución monoalfabética)

- Uso de bigramas (pares de letras) como unidades de cifrado
  - Dificulta criptanálisis estadístico
  - Generalizable a *n*-gramas
- `http://es.wikipedia.org/wiki/Cifrado_de_Playfair`
- Funcionamiento
  - Con la clave $\rightarrow$ construir matriz de cifrado 5x5
    - Rellenar matriz 5x5 con letras de la clave (sin repetidos)
    - Completar matriz en orden alfabético (sin repetidos) [*I* y *J* en misma casilla]
  - Insertar *X* entre caracteres consecutivos iguales
  - Insertar *X* al final para garantizar longitud par
  - Aplicar matriz de cifrado sobre pares de letras
    - Ambas letras en misma fila $\rightarrow$ reemplazarlas por las que tengan a su derecha
    - Ambas letras en misma columna $\rightarrow$ reemplazarlas por las que tengan debajo
    - En otro caso $\rightarrow$ reemplazarlas por las que estén en la ”esquina opuesta” de la matriz de cifrado

Criptografía clásica (V)

Máquinas de rotor (Cifrado polialfabético electromecánico)

1ª mitad S. XX (2da guerra mundial): Enigma (Alemania) y Purple (Japón)

- **Idea básica:** combinación ”secuencial” de múltiples cifrados monoalfabéticos
  - Combina cifradores sencillos para dar lugar a uno más poderoso
  - Idea de cifrado producto usada en la actualidad
- Estructura general
  - Conjunto de n cilindros (rotores) que rotan independientemente
  - Cada rotor tiene 26 puntos de entrada y 26 de salida asociados a las letras del alfabeto
  - Las conexiones entre esos puntos de entrada/salida definen una sustitución monoalfabética simple en cada rotor
  - Cada caracter de entrada supone un ”avance” del primer rotor
  - Un giro completo del rotor k supone un avance de una posición en el rotor $k + 1$
- **Objetivo:** cada caracter del mensaje en claro será cifrado con una sustitución distinta
  - Una misma letra será cifrada con distintas sustituciones en función de su posición en el texto de entrada
  - Engranajes de rotores + cableado de rotor + posición inicial $\Rightarrow$ deciden secuencia de sustituciones
  - Clave = posición inicial de los cilindros

Criptografía clásica (VI)

Cifrado de trasposición

- Reordena las letras del texto en claro según indique la clave
- Ejemplo 1: método de la escítala
  - `http://es.wikipedia.org/wiki/Escitala`
  - Uso de un bastón de grosor variable (= clave) con una cinta de escritura enrollada
  - Escritura a lo largo del bastón
  - El texto cifrado (permutado) queda en la cinta
  - Ataque: fuerza bruta sobre el ”ancho” de la clave (nº de saltos)
- Ejemplo 2: *Rail fence*
  - `http://es.wikipedia.org/wiki/Cifrado_Rail_Fence`
  - Disposición del texto en claro en ”zig-zag” de profundidad fija (=clave)
  - El texto cifrado (permutado) se lle en horizontal
  - Ataque: fuerza bruta sobre la ”produndiad” de la clave (nº de railes)

1. Fundamentos y evolución
   - Criptosistemas
   - Servicios de seguridad
   - Clasificación de los esquemas de cifrado
   - Criptografía clásica
2. Cifrado simétrico
   - Confusión y difusión: cifrado producto y redes Feistel
   - Data Encryption Standard (DES)
   - Otros algoritmos simétricos
3. Cifrado asimétrico
   - Principios de funcionamiento
   - Usos del cifrado asimétrico
   - Algoritmo RSA
   - Otros algoritmos asimétricos
4. Infraestructuras criptográficas: certificados digitales, PKI
   - Distribución fiable de claves públicas
   - Certificados digitales
   - Autoridades de certificación y PKI

Confusión y difusión

Propuesta teórica de Claude Shannon (1948) que caracteriza un modelo ideal de cifrador simétrico que minimiza las posibilidades de un análisis estadístico.

- **Objetivo:** evitar el criptoanálsis estadístico
- *Cifrado ideal:* cualquier tipo de estadística sobre el texto cifrado es independiente de la clave utilizada y del texto en claro involucrado

Principios de difusión y confusión: bloques básicos de construcción de cualquier sistema criptográfico

- **Confusión.** Pretende hacer que la relación entre la clave y el texto cifrado resultante lo más compleja posible
  - **Objetivo:** dificultar el descubrimiento de la clave con análisis estadístico
  - Cada porción del texto cifrado debe depender de forma compleja de distintas partes de la clave
    El texto cifrado debe dar la apariencia de ser totalmente aletorio
  - Se consigue mediante el uso de algoritmos de sustitución complejos gobernados por la clave
- **Difusión.** Pretende disipar la estructura estadística del texto en claro en el texto cifrado resultante
  - **Objetivo:** ocultar cualquier relación estadística entre texto en caro y cifrado
  - Cada bit del texto cifrado debe de verse afectado por mucho dígitos del texto en claro
    Es decir, cambios en un bit en claro afectan a muchos ($> 50\,\%$) bits cifrados
  - Se consigue mediante el uso de permutaciones complejas a nivel de bits

Cifradores actuales utilizan varias etapas que combinan ambos componentes (**cifrado producto**)

Redes Feistel (I)

Esquema de cifrado por bloques basado en un cifrado producto que combina operaciones de confusión y difusión.

- Simula cifrado de sustitución complejo mediante 2 o más operaciones de cifrado sencillas
- Cada etapa de cifrado elemental combina sustituciones y permutaciones
- Resultado final criptográficamente más fuerte que cifrados elementales aislados

`http://es.wikipedia.org/wiki/Cifrado_de_Feistel`

Es un esquema general, cada implementación concreta parametriza

- Tamaño de bloque y de claves
- Número de etapas
- Función de redondeo $F()$
- Generador de subclaves de etapa

Nota: Las redes de sustitución-permutación son otra arquitectura de cifrador similar. Aplican los mismos principios de confusión y difusión aunque con una estructura más compleja (`http://en.wikipedia.org/wiki/Substitution-permutation_network`)

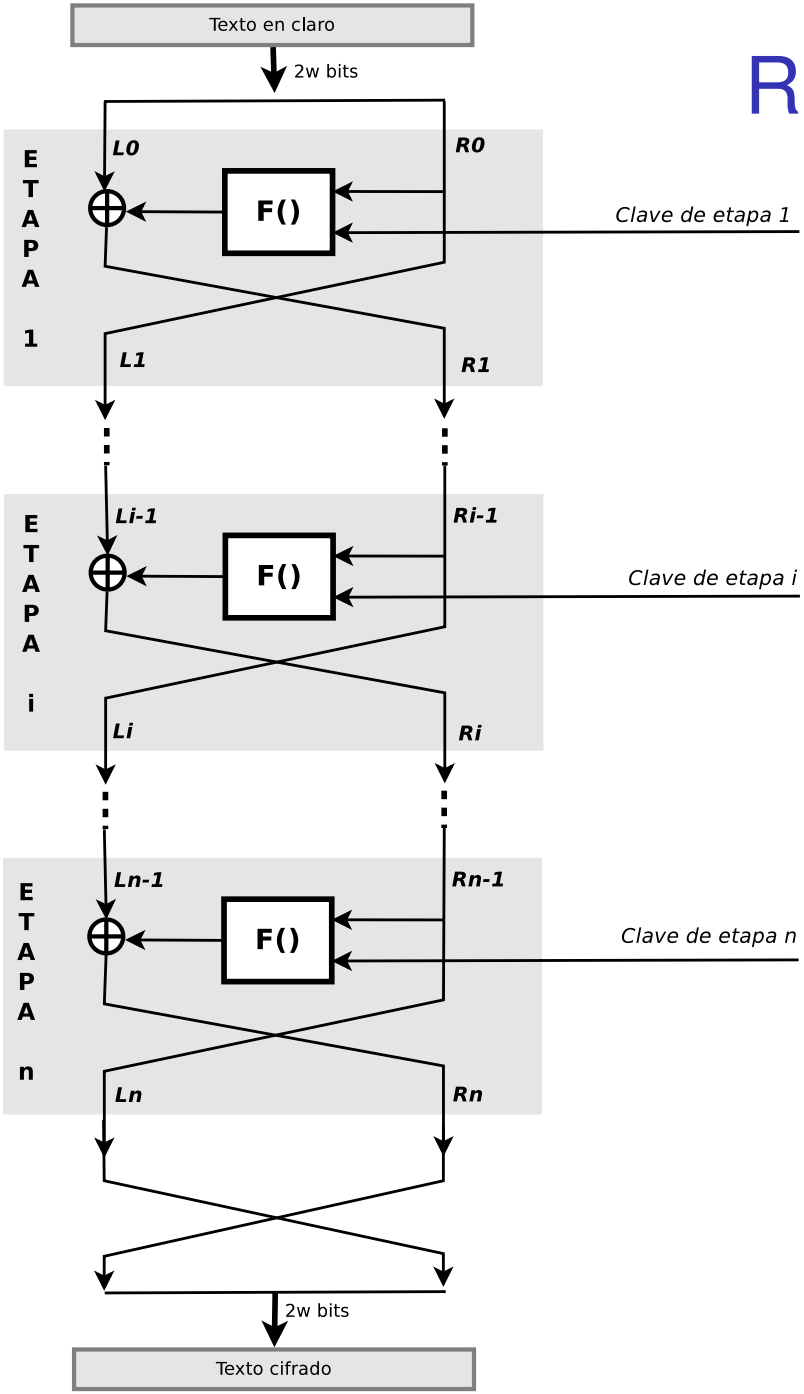
Redes Feistel (II)

Estructura genérica

1. Entrada: bloque en claro de longitud $2w$ + clave secreta k
2. Bloque de entrada se divide en 2 mitades: L_0 y R_0
3. L_0 y R_0 se pasan por N redondeos/etapas
4. En cada **etapa/redondeo** i:
   - Se recibe
     L_{i-1}: mitad izq. de la etapa previa
     R_{i-1}: mitad der. de la etapa previa
     k_i: subclave de etapa
   - Se calcula
     $L_i = R_{i-1}$
     $R_i = L_{i-1} \oplus F(R_{i-1}, k_i)$
     - La función de redondeo $F()$ realiza una sustitución gobernada por la clave de etapa sobre la mitad R_{i-1} que se combina con un XOR con L_{i-1}.
     - Para cada etapa se genera una subclave de etapa distinta a partir de la anterior.
     - Las dos mitades resultantes se intercambian como último paso de la etapa
5. Se termina con una permutación entre las mitades L_n y R_n finales

En descrifrado se realizan las mismas operaciones empleando las subclaves de etapa en orden inverso

Redes Feistel (III)



Data Encryption Standard (DES)

Estándar de cifrado adoptado por el NIST (*National Institute of Standards and Technology*) en los años 70.

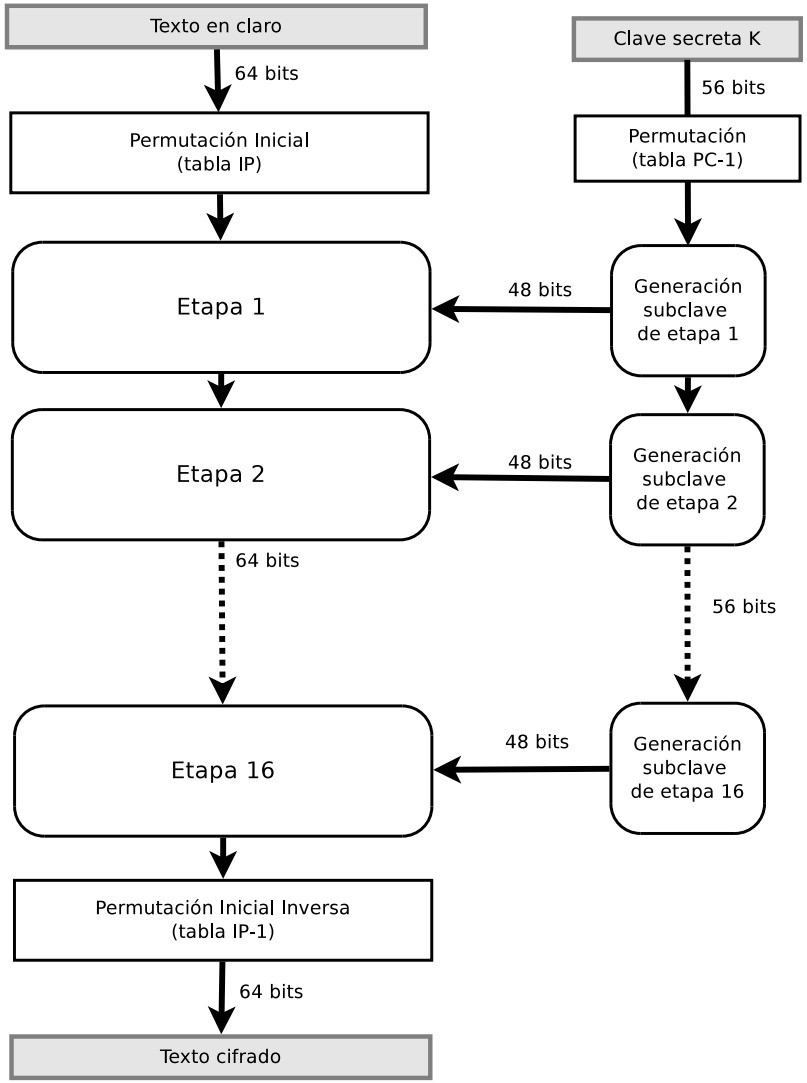
Estándar publicado: `http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf`

- Basado en el desarrollo Lucifer de IBM con aportaciones de la NSA (*National Security Agency*)
- Implementa una red Feistel de 16 etapas, con clave de 56 bits y tamaño de bloque de 64 bits
- Incluye una permutación inicial y su permutación inversa después de la última etapa

Reemplazado como estándar en 2001 por el cifrador AES (*Advanced Encryption Standard*) pero aún sigue en uso

- Algoritmo muy estudiado y criptoalizado en detalle
- Principal limitación: tamaño de clave demasiado pequeño
  - Se ha demostrado que se pueden romper claves DES de 56 bits por fuerza bruta con equipos distribuidos en días
  - En la práctica se suele emplear la variante 3DES (Tiple DES) con claves de 112 o 168 bits

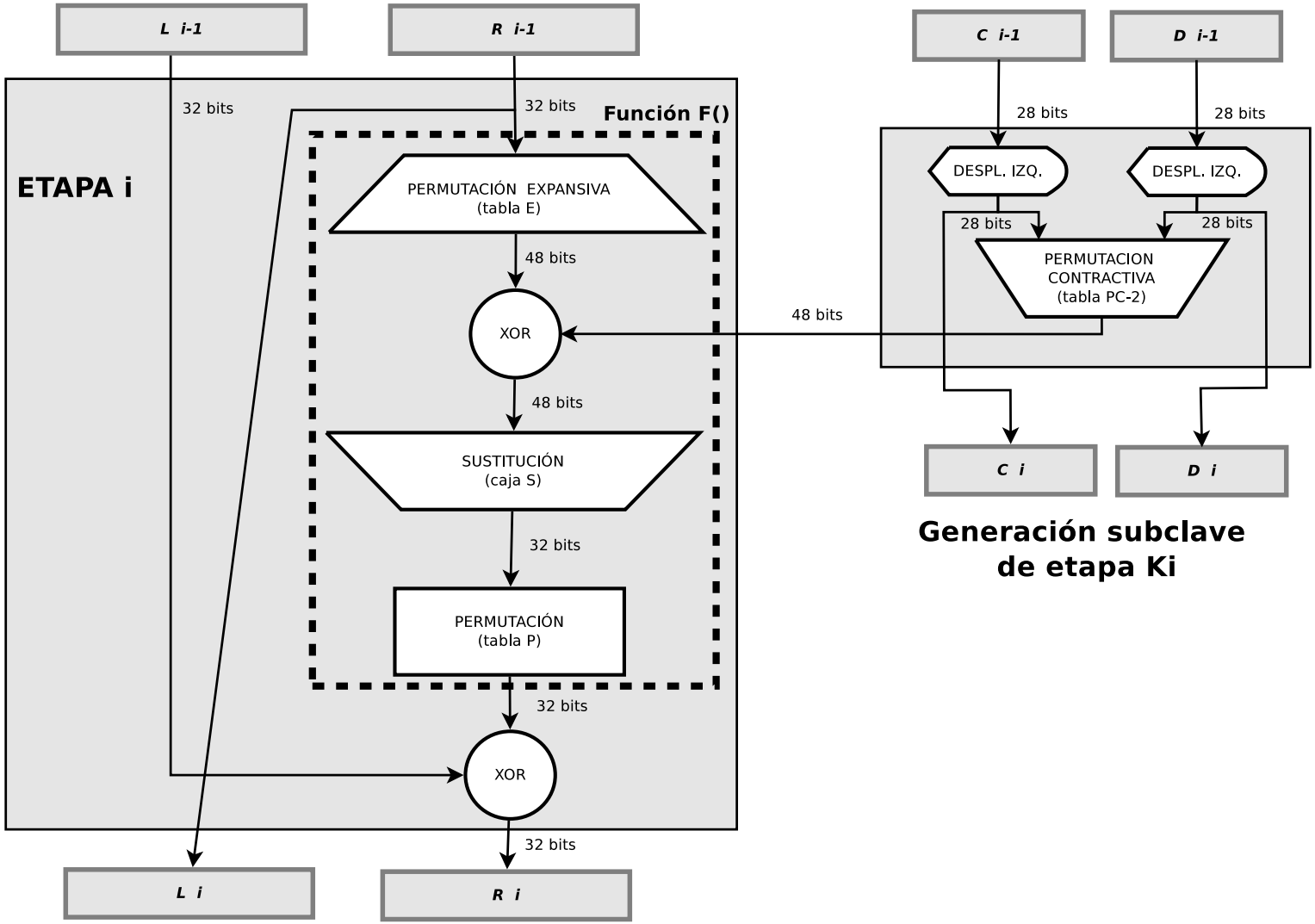
Estructura DES



Especificación de las ”cajas” en

`http://en.wikipedia.org/wiki/DES_supplementary_material`

Etapas DES y generación subclaves



Ampliaciones y mejoras: 3DES

Principal limitación DES: tamaño de clave escaso $\Rightarrow$ posible ataque por fuerza bruta en tiempo razonable

Solución: aplicar DES repetidas veces (3DES)

- Alterna 3 cifrados/descrifrados con DES en secuencia (permite "simular" DES simple de 1 sóla clave)
- Duplica o triplica tamaño de clave efectivo

Triple DES con 2 claves (clave de 112 bits)

CIFRADO

$$C = \mathbf{E}^{DES}_{k_1}[\mathbf{D}^{DES}_{k_2}[\mathbf{E}^{DES}_{k_1}[M]]]$$

DESCIFRADO

$$M = \mathbf{D}^{DES}_{k_1}[\mathbf{E}^{DES}_{k_2}[\mathbf{D}^{DES}_{k_1}[C]]]$$

Triple DES con 3 claves (clave de 168 bits)

CIFRADO

$$C = \mathbf{E}^{DES}_{k_3}[\mathbf{D}^{DES}_{k_2}[\mathbf{E}^{DES}_{k_1}[M]]]$$

DESCIFRADO

$$M = \mathbf{D}^{DES}_{k_1}[\mathbf{E}^{DES}_{k_2}[\mathbf{D}^{DES}_{k_3}[C]]]$$

Nota: Un "doble" DES con 2 claves que aplicara cifrado DES dos veces es vulnerable a un ataque *meet-in-the-middle* (`http://en.wikipedia.org/wiki/Meet-in-the-middle_attack`) que resulta no mucho más costoso que un ataque por fuerza bruta a una clave DES simple de 56 bits

Otros algoritmos simétricos

Cifradores de bloque

Algoritmo	Arquitectura	Tam. clave	Tam. bloque
DES (y 3DES)	red feistel (16 etapas)	56, 112, 168	64
AES/Rijndael	red sustitución-permutación (10, 12, 14 etapas)	128, 192, 256	128
IDEA	basado en red feistel (8 etapas)	128	64
Blowfish	red feistel (16 etapas)	hasta 448	64
RC5	red feistel (hasta 255 etapas)	hasta 2048	64

Cifradores de flujo

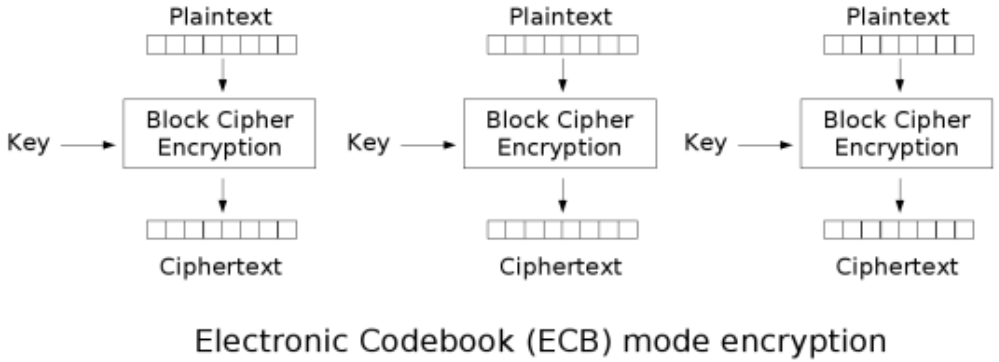
Algoritmo	Arquitectura	Tam. clave
RC4	gen. *keystream* pseudoaleatoria	40-2048
Trivium	gen. *keystream* pseudoaleatoria	80 (+ VI)

Modos de operación en cifradores en bloque

Modo ECB (*electronic code book*)

Cifra cada bloque de texto en claro por separado

- **Problema:** bloques en claro idénticos resultan en bloques cifrados idénticos

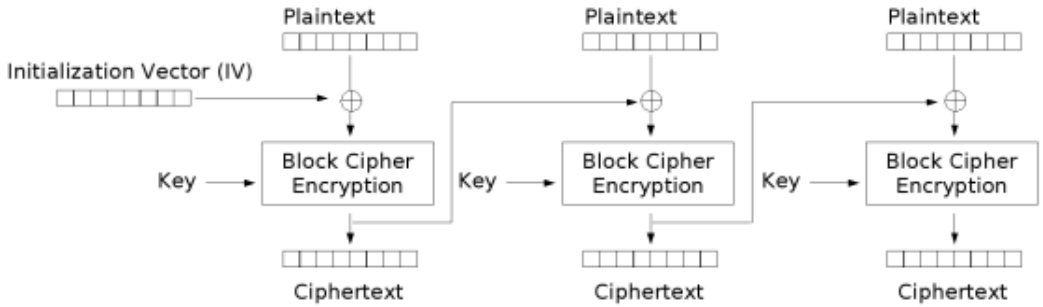


Electronic Codebook (ECB) mode encryption

Modo CBC (*cypher block chaining*)

"Encadena' los bloques cifrados: resultado de cifrar un bloque se combina (XOR) con siguiente bloque en claro

- Cada bloque cifrado depende de todos los bloques en claro previos
- Requiere vector de inicialización (acordado por emisor y receptor)



Cipher Block Chaining (CBC) mode encryption

Otros: modo CFB (*Cypher Feedback*)

$\rightarrow$ Emula cifrado en flujo con cifradores de bloque

Fuente: `http://en.wikipedia.org/wiki/Block_cipher_modes_of_operation`

1 Fundamentos y evolución
- Criptosistemas
- Servicios de seguridad
- Clasificación de los esquemas de cifrado
- Criptografía clásica

2 Cifrado simétrico
- Confusión y difusión: cifrado producto y redes Feistel
- Data Encryption Standard (DES)
- Otros algoritmos simétricos

3 Cifrado asimétrico
- Principios de funcionamiento
- Usos del cifrado asimétrico
- Algoritmo RSA
- Otros algoritmos asimétricos

4 Infraestructuras criptográficas: certificados digitales, PKI
- Distribución fiable de claves públicas
- Certificados digitales
- Autoridades de certificación y PKI

Cifrado asimétrico: principios de funcionamiento (I)

Propuesto por Diffie y Hellman en 1976.

- Basada en el uso de funciones matemáticas
- Utiliza dos claves: una para cifrar + otra diferente para descrifrar
- Originalmente mecanismo para intercambio de claves secretas
- Extensible a otros servicios: confidencialidad, distribución de claves, autentificación, firma digital

Requisitos generales (Diffie y Hellman)

1. Computacionalmente fácil para cada parte generar un par de claves(pública-privada)
2. Computacionalmente fácil para emisor cifrar un mensaje conociendo la clave pública del destino

$$Y = E_{KU_B}[X]$$

3. Computacionalmente fácil para destino descifrar un texto cifrado conociendo la clave privada

$$X = D_{KR_B}[Y] = D_{KR_B}[E_{KU_B}[X]]$$

4. Computacionalmente impracticable para un oponente determinar la clave privada conociendo la clave publica
5. Computacionalmente impracticable para un oponente determinar el mensaje original a partir de la clave pública y del texto cifrado
6. **OPCIONAL:** Cualquiera de las 2 claves puede usarse para cifrar, y la otra para descifrar

$$X = D_{KR_B}[E_{KU_B}(X)] = D_{KU_B}[E_{KR_B}(X)]$$

Cifrado asimétrico: principios de funcionamiento (II)

Uso de **funciones *"one-way"*** (funciones un único sentido)

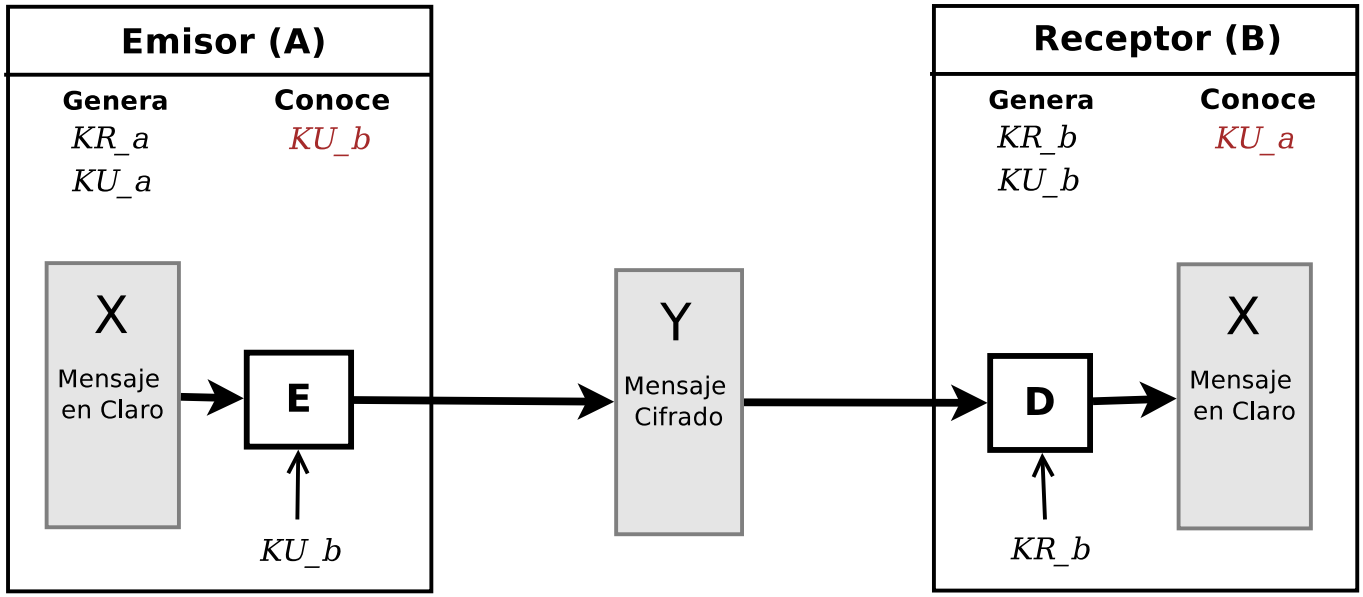
- Función que tiene una única inversa
- El cálculo de la función ($Y = f(X)$) debe ser "fácil", mientras que el cálculo de la inversa ($X = f^{-1}(Y)$) debe ser impracticable
- Debe ser fácil de calcular en el otro sentido (función inversa) si se conoce información adicional [**función *"trap-door one-way"*** (función de único sentido con puerta trampa)]
  - $Y = f_K(X)$ debe ser fácil si X es conocido
  - $X = f_K^{-1}(Y)$ debe ser fácil si K e Y son conocidos
  - $X = f_K^{-1}(Y)$ debe ser impracticable si Y es conocido y K desconocido

Todos los algoritmos/esquemas de clave pública dependen del uso explícito o implícito de una determinada función de único sentido con puerta trampa

- Dicha "función" condiciona su funcionamiento y sus características.
- Ejemplos:
  - factorización en n^o primos en RSA
  - problema del logaritmo discreto en Diffie-Helman y ElGamal
  - problema de caracterización de curvas elípticas en algoritmos CCE (*cifrado con curvas elípticas*)

Usos del cifrado asimétrico: confidencialidad

- Emisor cifra mensaje con clave pública del destinatario
- Receptor descifra mensaje con su propia clave privada



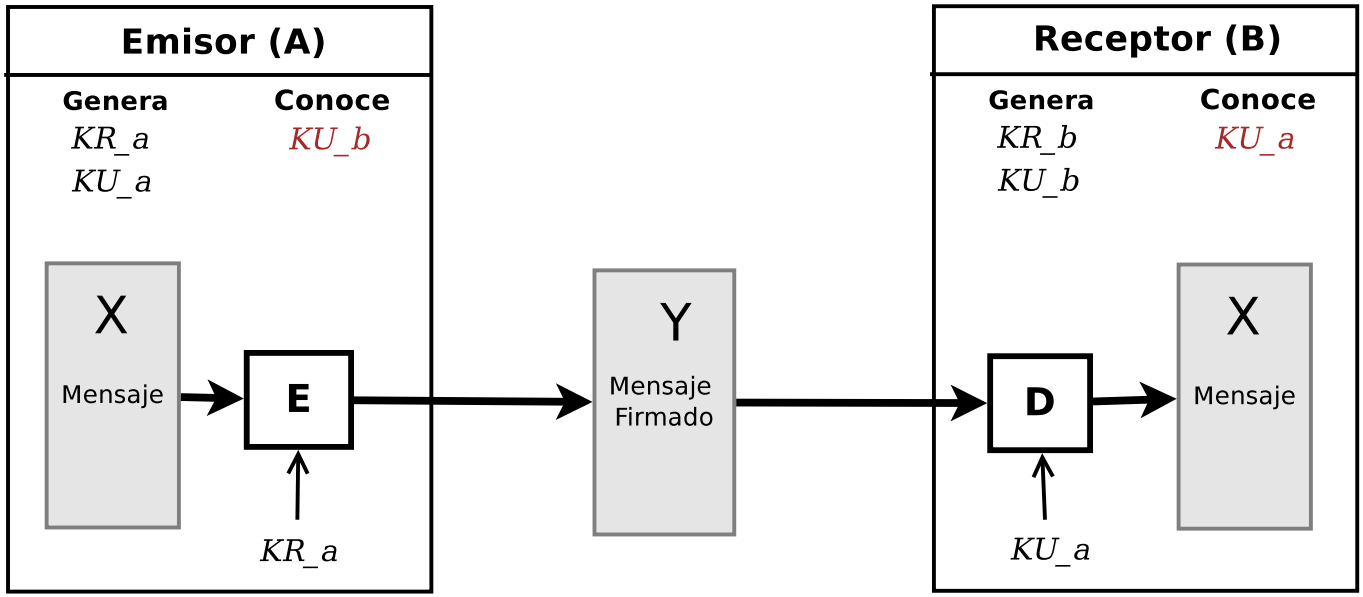
KU_x: clave pública de X
KR_x: clave privada de X

Sólo el propietario de $\mathbf{KR_b}$ puede descrifrar un mensaje cifrado con $\mathbf{KU_b}$

Conclusión: sólo el receptor podrá ver el mensaje (garantiza **confidencialidad**)

Usos del cifrado asimétrico: firma digital

- Emisor cifra (firma) mensaje con su propia clave privada
- Receptor descifra (verifica) mensaje con la clave pública del emisor



Sólo el propietario de $\mathbf{KR_a}$ pudo cifrar un mensaje descifrado con $\mathbf{KU_a}$

Conclusión: sólo el emisor pudo haber firmado el mensaje

Se garantiza
- **autenticidad:** sólo A pudo firmar el mensaje
- **integridad:** nadie, excepto A, puede modificar el mensaje
- **no repudio:** A no puede negar que él creo/firmó el mensaje

Funciones HASH (I)

Limitación cifrado asimétrico: alto coste computacional en operaciones de cifrado/descifrado

- Operaciones aritméticas con ”números muy grandes”
- No aplicable en cifrado/descrifrado de grandes volúmenes de datos

Solución { uso funciones HASH en firma digital
esquemas de cifrado ”híbrido”

Funciones HASH criptográficas

Algoritmos unidireccionales que toman una cantidad arbitraría de datos y generan un valor de tamaño fijo (resumen) específico para dichos datos

- También: funciones resumen, huellas digitales, checksums, etc
- Usados en firma digital, códigos de autenticación de mensajes (HMAC), contraseñas, ...

Ejemplos:

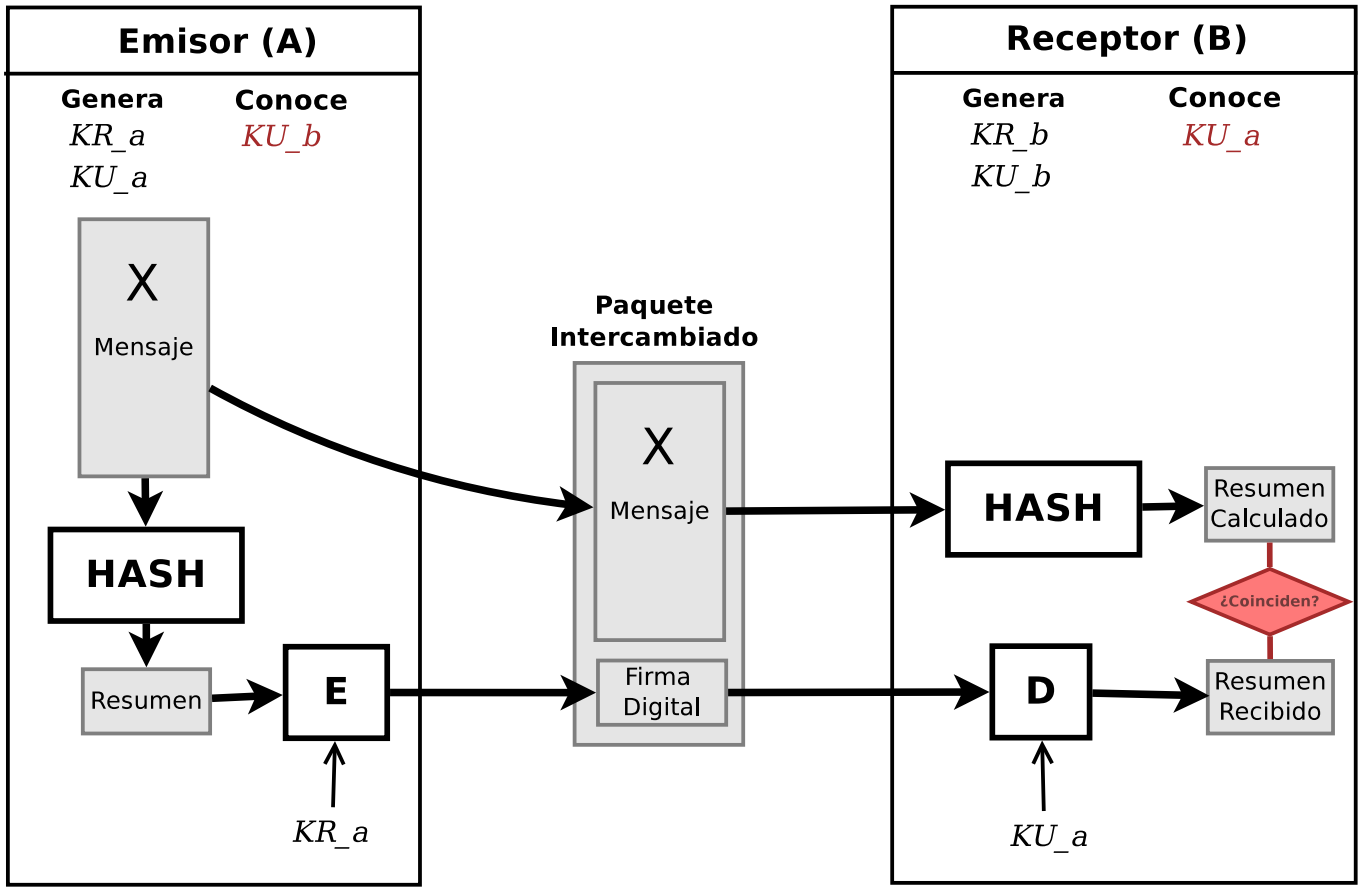
- MD5, resúmenes de 128 bits (RFC 1221 `http://tools.ietf.org/html/rfc1321`)
  Considerado poco resistente ante colisiones, no recomendado en firma digital
- SHA-1, resúmenes de 160 bits (variantes: SHA-224, SHA-256, SHA-512)
- RIPEMD, resúmenes de 160 bits

Funciones HASH (II)

Propiedades

1. $HASH()$ puede aplicarse a bloques de cualquer tamaño y produce una salida de tamaño fijo
2. $HASH(X)$ es computacionalmente fácil de calcular.
3. Para un valor $HASH(X)$ dado es computacionalmente impractible encontrar/construir el bloque X que lo origina [unidireccionalidad]
   - **Nota:** uso de *rainbow tables* (tablas de hashes precaculadas + ”funciones inversas”) acelera ”adivinación” de resúmenes por fuerza bruta (especialmente passwords)
4. Para un bloque X dado es computacionalmente impractible encontrar/construir otro bloque Y ($\neq X$) con $HASH(Y) = HASH(X)$ [resistencia débil a colisiones]
   - Protege frente a falsificaciones (ataques de preimagen)
5. Es computacionalmente impractible encontrar/construir un par (X, Y) tales que $HASH(X) = HASH(Y)$ [resistencia fuerte a colisiones]
   - Protege frente a ”ataques del cumpleaños” (ataques de colisión)

Firma digital con func. HASH



1. Función hash garantiza **integridad** (si se modifica mensaje funciones hash no coincidirán)
2. Cifrar con **KR_a** garantiza
   - **autenticidad:** sólo A pudo firmar el mensaje
   - **no repudio:** A no puede negar que firmó el mensaje

Usos del cifrado asimétrico: intercambio de claves

Distintas formas de usar algoritmos asimétricos para intercambio de claves secretas

Opción 1: envío de claves secretas cifradas con clave pública del destinatario (cifrado con RSA)

- Formato de mensaje de PGP (*Pretty Good Privacy*)
  - Cuerpo del mensaje cifrado con un algoritmo simétrico usando una clave secreta de un sólo uso
  - Clave secreta se cifra con clave pública del destinatario y se envía incluida en el mensaje PGP
  - Adicionalmente, se puede firmar el mensaje con clave privada de firma del emisor
- Negociación de *"clave maestra"* en SSL/TLS (*Secure Socket Layer / Transport Layer Security*)
  - En establecimiento de conexión, Servidor se autentica ante Cliente con certificado digital aceptado (opcionalmente, también el Cliente) que incluye su clave pública
  - Con esa clave, Cliente envía Servidor *"clave maestra"* cifrada con RSA
  - A partir *"clave maestra"* Cliente y Servidor generan claves secretas de cifrado y atenticación (usadas en mensajes intercambiados una vez establecida la conexión)

Opción 2: algoritmos asimétricos específicos para acuerdo de claves

- Ejemplo: algoritmo de acuerdo de claves de Diffie-Hellman

Algoritmo Diffie-Hellman de acuerdo de claves (I)

Ejemplos de uso

- Negociación claves IPSec (protocolo IKE (*Internet Key Exchange*))
- Negociación de la *"clave maestra"* en SSL/TLS

Logaritmos discretos

- Función *one-way* sobre la que se fundamenta DH (también ElGamal)
- Generalización del logaritmo en aritmética modular.
- **Definición**: $\boxed{c \equiv \log_b(a)\ (mod\ N)\ \text{ si y sólo si }\ a \equiv b^c\ (mod\ N)}$

  Se trata de encontrar el exponente c al que habría que elevar la base b para obtener el valor a

  - Factible calcular $(b^c\ mod\ N)$
  - Computacionalmente impracticable $(\log_b(a)\ mod\ N))$ (para valores grandes de p y g)

Algoritmo Diffie-Hellman (II)

Funcionamiento intercambio clave Diffie-Hellman

A y *B* quieren acordar una clave secreta

1. *A* y *B* acuerdan $\begin{cases} \text{un nº primo "grande" } \mathbf{p} \\ \text{nº entero } \mathbf{g} \text{ primo relativo con } p \end{cases}$ (parte pública)
2. *A* selecciona un nº entero aleatorio "grande" **x** (parte privada)
   Calcula $\mathbf{X} = (\mathbf{g^x \bmod p})$ y se lo envía a *B*.
3. *B* selecciona un nº entero aleatorio "grande" **y** (parte privada)
   Calcula $\mathbf{Y} = (\mathbf{g^y \bmod p})$ y se lo envía a *A*.
4. *A* calcula $\mathbf{K} = (\mathbf{Y}^x \ mod \ p)$
5. *B* calcula $\mathbf{K'} = (\mathbf{X}^y \ mod \ p)$

Se verifica que $\boxed{K \equiv K' \ (mod \ p)}$

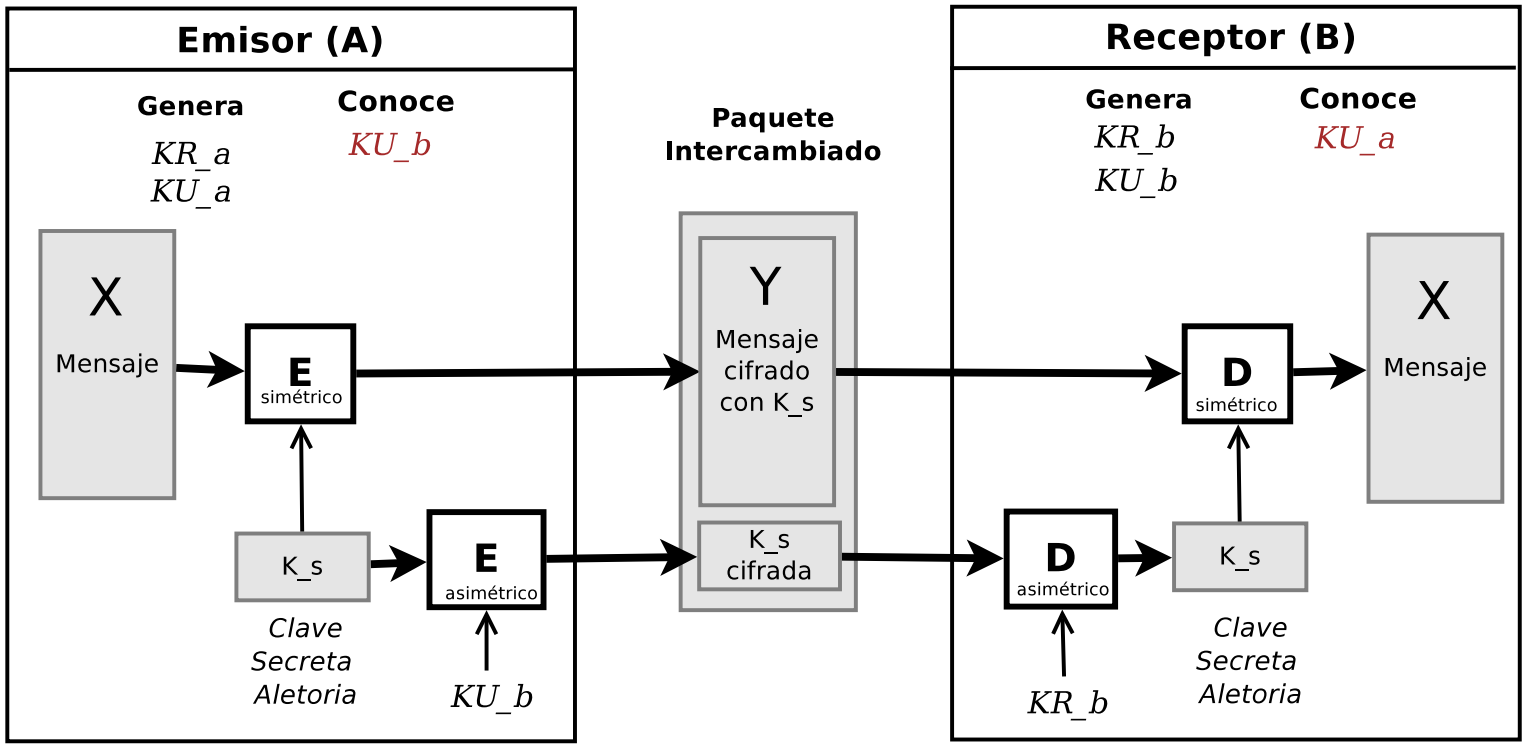
$$\begin{aligned} \mathbf{K} &= (\mathbf{Y}^x \ mod \ p) = ((g^y \ mod \ p)^x \ mod \ p) = \\ &= (g^{y \cdot x} \ mod \ p) = (g^{x \cdot y} \ mod \ p) = \\ &= ((g^x \ mod \ p)^y \ mod \ p) = (\mathbf{X}^y \ mod \ p) = \mathbf{K'} \end{aligned}$$

K será la clave simétrica acordada (secreto compartido)

- *K* nunca viaja por la red
- Atacante sólo tiene acceso a **X**, **Y**, *p* y *g*.
- Para conocer *K* debe encontrar *x* ó *y* que verifiquen $\begin{cases} x = \log_g(\mathbf{X}) \ mod \ p \\ y = \log_g(\mathbf{Y}) \ mod \ p \end{cases}$
- Computacionalmente impractiable para valores grandes de *p* y *g*.

Cifrado híbrido

- Cifrado simétrico de datos con clave secreta aleatoria (de un sólo uso) generada por emisor
- Intercambio de clave secreta aletoria con cifrado asimétrico



Sólo el receptor podrá recuperar la clave secreta y con ella descrifrar el mensaje

(garantiza **confidencialidad**)

RSA: Fundamentos matemáticos

- Primera propuesta que sigue el esquema de criptografía asimétrica definido por Diffie y Hellman (publicado por Ron Rivest, Adi Shamir, y Len Adleman en 1978)
- Cifrado por bloques
  - texto plano y texto cifrado son nº enteros entre 0 y $n-1$ para algun n
  - k = tamaño de clave = tamaño de bloque, con $2^{k-1} < n \leq 2^k$
- Trabaja con enteros y operaciones aritméticas *mod N*
  - Aprovecha propiedades matemáticas específicas de la aritmética molular (teorema Euler-Fermat)

RSA: Funcionamiento (I)

Funciones de cifrado y descifrado

Siendo M el bloque del texto plano y C el bloque de texto cifrado

CIFRADO: $C = M^e \bmod n$

DESCIFRADO: $M = C^d \bmod n = (M^e)^d \bmod n = M^{ed} \bmod n$

Claves pública y privada

CLAVE PÚBLICA: $\mathbf{KU} = [\mathbf{e}, \mathbf{n}]$

CLAVE PRIVADA: $\mathbf{KR} = [\mathbf{d}, \mathbf{n}]$

- Emisor y receptor conocen el valor de **n** (módulo)
- Emisor conoce el valor de **e** (exponente de cifrado)
- Receptor conoce el valor de **d** (exponente de descifrado)

Estos valores de las claves deben verificar:

1. Es posible encontrar valores de **e**, **d**, y **n** tales que:

$$M^{ed} \bmod n = M \text{ para todo } M < n$$

2. Debe ser relativamente fácil calcular $M^e \bmod n$ y $C^d \bmod n$ para $M < n$.
3. Debe ser impracticable determinar d dado e y n.

Existen teoremas y algoritmos que aseguran que los 2 primeros requisitos se pueden cumplir fácilmente

El tercer requisito se puede cumplir para valores grandes de n y d

RSA: Funcionamiento (II)

Generación de claves

1. Seleccionar 2 números primos, **p** y **q** ($p \neq q$)
2. Calcular $\mathbf{n} = \mathbf{p} \cdot \mathbf{q}$ (módulo usado en cifrado y descrifrado)
3. Calcular $\Phi(n) = (p-1) \cdot (q-1)$
4. Seleccionar aleatoriamente un entero **e** tal que:
   $1 < e < \Phi(n)$ y $\boxed{mcd(\Phi(n), e) = 1}$ [e primo relativo de $\Phi(n)$]
5. Seleccionar/calcular **d** que verifique:
   $\boxed{(d \cdot e)\ mod\ \Phi(n) = 1} \Rightarrow d = e^{-1}\ mod\ \Phi(n)$ [d es el inverso módulo $\Phi(n)$ de e]
6. Claves $\begin{cases} \text{pública}: & KU = \{e, n\} \\ \text{privada}: & KU = \{d, n\} \end{cases}$

Se puede demostrar que los valores e y d cumplen las 3 propiedades descritas anteriormente

NOTA: $\Phi(n)$ se denomina *función Totient de Euler*

- Es el nº de enteros menores o iguales a n que son primos relativos con n
- Fácil de calcular si se conocen los factores primos de n y muy difícil en caso contrario

Para el caso $n = a \cdot b$, con a y b primos, tenemos $\Phi(n) = (a-1) \cdot (b-1)$

RSA: Ejemplo

(a) Generación de claves

1. $p = 7$ y $q = 17$
2. $n = 7 \cdot 17 = 119$
3. $\Phi(119) = (p-1)(q-1) = 96$
4. $e = 5$ (sirve cualquier primo relativo con $\Phi(n)$)
5. $d \cdot e \; mod \; \Phi(n) \; = \; 1$
   Encontrar d que verifique $(d \cdot 5) \; mod \; 96 \; = \; 1$ (equiv. $\exists \; k \quad$ tal que $(d \cdot 5) = k \cdot 96 + 1$)
   Seleccionamos $d = 77$, ya que $(77 \cdot 5) \; = \; 385 = 4 \cdot 96 + 1$
6. Clave pública: $KU = \{5, 119\}$
7. Clave privada: $KR = \{77, 119\}$

(b) Cifrado del mensaje $M = 43$:

$$C = M^e mod \; n = \quad 43^5 mod \; 119 = \quad 147008443 \; mod \; 119 = 8$$

(c) Descifrado de $C = 8$:

$$\begin{aligned} M \quad &= C^d mod \; n = 8^{77} \; mod \; 119 = \\ &= 3,450873173 \cdot 10^{69} \; mod \; 119 = 34508731733952 \ldots 899648 \; mod \; 119 = \\ &= 43 \end{aligned}$$

RSA: ataques y debilidades

Siempre posible ataque fuerza bruta $\Rightarrow$ uso claves grandes

Atacante conoce $KU = \{e, n\} \Rightarrow$ 3 aproximaciones para atacar RSA matemáticamente

- Factorizar n en sus dos factores primos (permite conocer $\Phi(n)$ y el cálculo de d)
- Determinar $\Phi(n)$ directamente
- Determinar d directamente (fuerza bruta)

Mayoría de aproximaciones al criptoanálisis de RSA enfocadas en factorizar n

- Otras alternativas consumen al menos el mismo tiempo y complejidad
- Para n grande con factores primos grandes, factorización es problema *NP*

Otros:

- Ataque de mensaje probable (exclusivo de métodos asimétricos)
  - Ante mensajes cifrados de "pocos" bits (ej. clave DES de 56 bits) oponente puede cifrar todos los posibles mensajes utilizando KU de destino
  - Podría "descifrar" el mensaje original encontrando aquel que coincida con el texto cifrado transmitido
  - No importa la longitud de la clave $\rightarrow$ sería un ataque de fuerza bruta de 56 bits
  - Se puede evitar añadiendo bits aleatorios a mensajes simples
- Ataques de tiempo *(timing attacks)*
  - Tipo de *side channel attack*, explota características específicas de algunas implementaciones de RSA
  - **Objetivo:** reducir espacio de claves analizando tiempo empleado en operaciones de cifrado

Implementaciones de RSA que usan "exponenciacion binaria" para acelerar el cálculo de potencias $mod\ n$ tienen un tiempo de cifrado proporcional al n^o de 1's en el exponente

Otros algoritmos asimétricos

Algoritmo	**Cifrado**	**Firma digital**	**Intercambio claves**
RSA	SI	SI	SI
Diffie-Hellman			SI
El Gamal	SI	SI	SI
DSA		SI	
ECC (cifrado de curva elíptica)	SI	SI	SI
ECDSA (DSA con curva elíptica)		SI	

1 Fundamentos y evolución
- Criptosistemas
- Servicios de seguridad
- Clasificación de los esquemas de cifrado
- Criptografía clásica

2 Cifrado simétrico
- Confusión y difusión: cifrado producto y redes Feistel
- Data Encryption Standard (DES)
- Otros algoritmos simétricos

3 Cifrado asimétrico
- Principios de funcionamiento
- Usos del cifrado asimétrico
- Algoritmo RSA
- Otros algoritmos asimétricos

4 Infraestructuras criptográficas: certificados digitales, PKI
- Distribución fiable de claves públicas
- Certificados digitales
- Autoridades de certificación y PKI

Distribución fiable claves públicas (I)

Punto débil del cifrado asimétrico: distribuir claves públicas garantizando la identidad de su propietario (persona, servidor, etc)

- Cifrado asimétrico se basa en la confianza de que una clave pública realmente pertenece a su propietario
- Una clave pública ”no legítima” falsamente vinculada a un usuario
  - compromete el tráfico cifrado enviado a ese usuario
  - hace aceptar firmas digitales como suyas cuando realmente no los son
- Necesidad de mecanismos fiables para publicar/distribuir claves públicas garantizando la identidad de su propietario

Opción 1: anuncio público

- Se confía en las KU que el propio usario difunde
  - Directamente o publicándolas en un repositorio como los *key servers* de PGP
- Ninguna garantía de autenticación frente a KU falsificadas

Distribución fiable claves públicas (II)

Opción 2: acreditación mútua entre usuarios

- Se confía en KUs firmadas por usuarios de ”confianza”
  - Usuarios en los cuales confiamos y hemos obtenido su KU de forma fiable firman con su KR las KU de otros usuarios
  - KUs firmadas por usuarios de nuestra confianza (tenemos su KU para comprobarlo) se admiten como confiables
  - Se asume que un usuario que firma la KU de otro ha comprobado su autenticidad
- Confianza ”horizontal” (basada en reputación de los usuarios, ”transitiva”)
  - Esquema utilizado en los *anillos de confianza* (*web of trust*) de PGP

Opción 3: uso de certificados digitales y autoridades de certificación

- Se confía en un ”tercero” que comprueba y certifica la identidad del propietario de la KU
- La autoridad de certificación (CA) firma digitalmente un certificado digital que vinculada la identidad del usuario y su KU
- Confianza ”vertical” (jerárquica y centralizada)
  - Si se reconoce (”confía”) la CA que emite el certificado se da por buena la autenticidad de la KU incluida en él
  - Requiere una infraestructura y unas reglas (PKI, *public key infraestructure*)

Certificados digitales

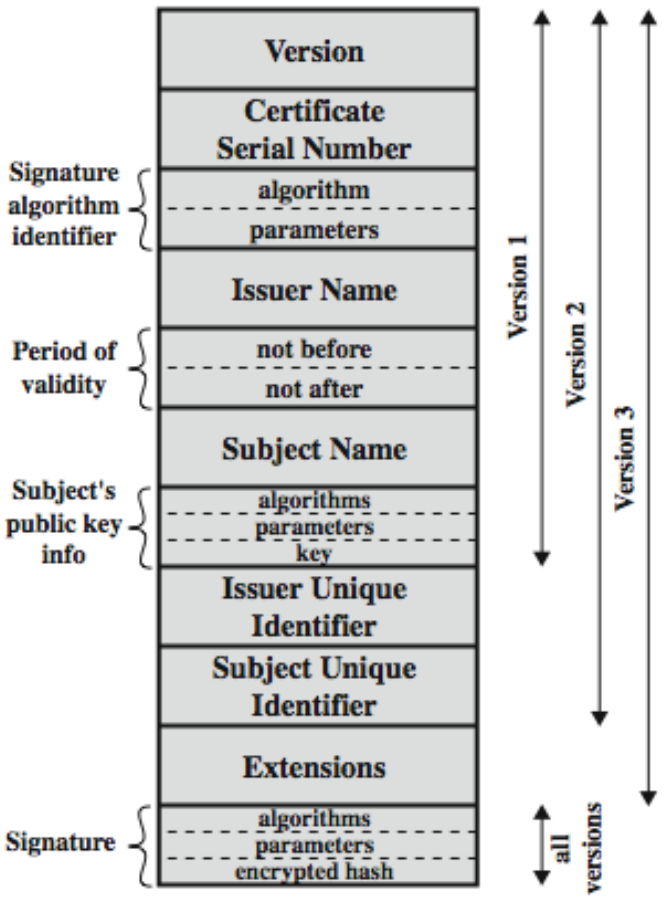
Funcionamiento básico

- Autoridad certificadora (CA) emite un certificado de clave pública para cada participante que se lo requiera
  - Certificado digital contiene { IDENTIFICACIÓN DEL PROPIETARIO; CLAVE PÚBLICA DEL PROPIETARIO; INFO. ADICIONAL [validez, uso previsto] }
  - Todas las partes firmadas digitalmente por CA (con su clave privada KR_{CA})
    $\Rightarrow$ Sólo la CA puede crear y/o modificar sus certificados
  - CA garantiza que la clave publica pertenece al usuario identificado en el certificado
    - CA se responsabiliza de comprobar la autenticiadad de esas *KU*
    - Solicitud de certificado en persona o por una comunicación autentificada segura
- Participantes distribuyen su clave pública enviando su propio certificado
- El participante que lo reciba puede verificar que el certificado es auténtico
  - Fué creado y firmado por la CA correspondiente (reconocida por ambos)
  - Puede recuperar la clave pública del otro participante y verificar su validez/autenticidad (comprobando la firma digital con la clave pública de la CA)

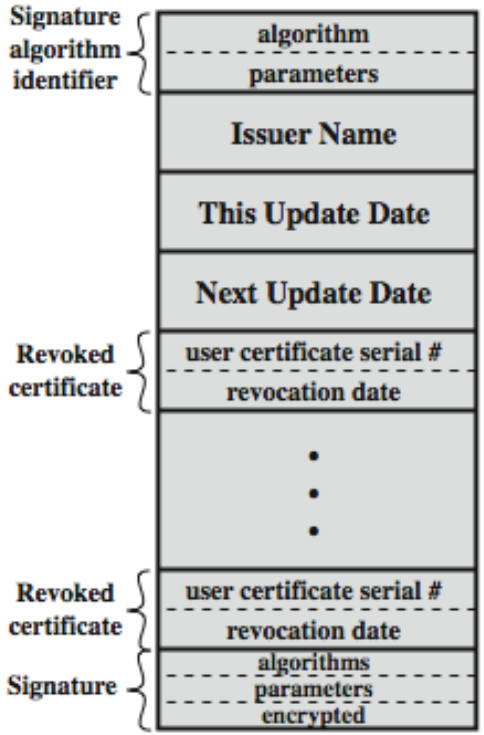
Certificados digitales (II)

Certificados X.509 : formato para almacenar e intercambiar certificados digitales

- Especificado en la definicion del servicio de directorio X.500
- Incluye la definición del procedimiento para validar ”rutas de certificación” (cadenas de certificados)
- Define ”contenedor” para info. de certificados independiente de algoritmos de cifrado concretos



(a) X.509 Certificate



(b) Certificate Revocation List

Fuente: `http://williamstallings.com/Cryptography/Crypto5e-Instructor/`

Certificados digitales (IV)

Certificado de servidor (emitido y firmado por una Autoridad de Certificacion [CA])

```
openssl x509 -text -in mancomun.org.pem

Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 1018832349 (0x3cba25dd)
        Signature Algorithm: sha1WithRSAEncryption
        Issuer: C=ES, O=FNMT, OU=FNMT Clase 2 CA
        Validity
            Not Before: Mar 15 16:16:42 2010 GMT
            Not After : Mar 15 16:16:42 2014 GMT
        Subject: C=ES, O=FNMT, OU=FNMT Clase 2 CA, OU=publicos, OU=500070015, CN=*.mancomun.org
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (1024 bit)
                Modulus (1024 bit):
                    00:df:7a:37:12:8b:ce:7a:d4:ca:f9:bc:a8:26:bf:
                    27:66:df:ce:cf:8e:df:94:0d:05:52:5d:a4:aa:7e:
                    . . .
                    35:e5:6b:9c:cb:c7:8f:ac:11
                Exponent: 65537 (0x10001)
        X509v3 extensions:
             X509v3 Subject Alternative Name:
                DirName:1.3.6.1.4.1.5734.1.14=Xunta de Galicia/1.3.6.1.4.1.5734.1.8=wildcard *.mancomun.org
            X509v3 Basic Constraints:
                CA:FALSE
            X509v3 CRL Distribution Points:
                DirName:/C=ES/O=FNMT/OU=FNMT Clase 2 CA/CN=CRL6885
          . . .
    Signature Algorithm: sha1WithRSAEncryption
        6d:7d:c2:67:cd:5c:36:28:ab:6f:71:14:03:66:79:25:d8:55:
        31:62:29:7f:6e:f4:cd:3a:30:a9:2f:e7:e4:08:8e:e8:b3:ad:
        . . .
        cb:af
```

Certificados digitales (V)

Certificado autofirmado (Autoridad de Certificacion RAIZ)

```
openssl x509 -text -in FNMTClase2CA-FNMT.pem

Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 921770777 (0x36f11b19)
        Signature Algorithm: sha1WithRSAEncryption
        Issuer: C=ES, O=FNMT, OU=FNMT Clase 2 CA          <<-- Emisor (C.A.)
        Validity
            Not Before: Mar 18 14:56:19 1999 GMT
            Not After : Mar 18 15:26:19 2019 GMT
        Subject: C=ES, O=FNMT, OU=FNMT Clase 2 CA         <<-- Id. dueño de KU (C.A.)
        Subject Public Key Info:                          <<-- Valor de la KU
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (1024 bit)
                Modulus (1024 bit):
                    00:98:3f:ad:19:36:93:3d:3e:fe:76:42:14:fd:35:
                    6f:f1:fa:ad:22:7a:58:e3:46:d0:5d:c6:5a:f9:62:
                    . . .
                    e2:ff:19:f4:94:09:d5:96:61
                Exponent: 3 (0x3)
        X509v3 extensions:                                <<-- Opcional
            Netscape Cert Type:
                SSL CA, S/MIME CA, Object Signing CA
            X509v3 CRL Distribution Points:
                DirName:/C=ES/O=FNMT/OU=FNMT Clase 2 CA/CN=CRL1
            X509v3 Basic Constraints:
                CA:TRUE
            ...
    Signature Algorithm: sha1WithRSAEncryption            <<-- Firma del certificado
        61:4c:a0:7c:59:63:5b:66:f8:ee:65:13:ce:43:80:47:b9:b2:
        35:c9:c8:84:c7:6b:73:60:45:e4:9d:37:9d:f5:8e:25:b9:f9:
        . . .
        08:5d
```

Public Key Infraestructure (I)

Uso de cifrado asimétrico a gran escala $\Rightarrow$ necesidad de organización, infraestructura y procedimientos

Componentes

- *Autoridad de certificación (CA):* emite y revoca certificados
- *Autoridad/es de registro (RA):* verifica identidad de titulares de certificados
- *Repositorios:* repositorio de certificados + listas de revocación de certificados (CRL)
- *Autoridad de validación (VA):* verifica validez de certificados
- *Autoridad de sellado de tiempo (TSA):* firma documentos acreditando su ”existencia’ en un momento dado
- *Entidades finales:* propietarios/solicitantes de certificados (servidores, personas, software, etc)

Public Key Infraestructure (II)

Cadenas de certificados y jerarquías de C.A.

- **Problema:** comunicación con un usuario que tiene un certificado emitido por una CA no reconocida
  - No es posible comprobar su validez (se desconoce *KU* de la nueva CA)
  - No hay ‘’confianza” en nueva CA (nadie lo respalda)
- **Solucion:**
  - Exigir que la nueva CA tenga un certificado que garantice la autenticidad de su clave pública
  - Ese certificado estará firmado por otra CA en la que confiemos
- Puede establecerse una jerarquía de CAs
  - Las CA de más bajo nivel certifican participantes/usuarios
  - Las CA de cada nivel son certificadas por una de nivel superior

Cadena de certificados: para verificar una KU puede ser necesario obtener y validar la secuencia de certificados de CA’s intermedias hasta llegar a una CA raíz reconocida

- Siguiendo la cadena de certificados de forma ”descendente” se puede ir validando la autenticidad las sucesivas KUs
- CAs raíz certifican a sí mismas sus propias KUs (certificados autofirmados)
- En las aplicaciones/servicios los certificados autofirmados de las CA raíz se configuran manualmente

Public Key Infraestructure (III)

Listas de revocación de certificados (CRLs)

CAs deben mantener y publicar periodicamente las CRLs

- Lista de certificados que han dejado de ser válidos antes de su caducidad
  - certificado contenía información erronea (erratas, etc)
  - fueron sustituidos por otros o sus propietarios han decidido dejar de usarlos
  - cambio/desaparición del propietario
  - la KR asociado fue comprometida
  - certificados irregulares (emitidos tras suplantar al propietario legítimo, etc)
- Firmada con la KR de la CA emisora

Antes de utilizar una KU extraida de un certificado digital, participantes deberían de comprobar que ese certificado no haya sido revocado por la CA emisora

Alternativa 1: comprobación off-line

- Manteniendo una ”caché” local con las versiones actuales de CRLs descargadas desde las CAs
- Cada certificado recibido es cotejado con esas CRLs

Requiere descargar periódicamente la CRL

Alternativa 2: comprobación on-line

- Empleando el protocolo OCSP *(Online Certificate Status Protocol)*
- Consulta a un servidor (de la propia CA o externo) sobre la revocación de un certificado dado (en base a su nº de serie)

Requiere conexión permanente con un servidor OCSP