Tema 4.3: Seguridad

Sistemas Distribuidos

2022-2023

Introducción

Los sistemas distribuidos son más vulnerables a ataques:

- La red es accesible por terceros
- Múltiples nodos ⇒ Más puertas que guardar

La seguridad del sistema es igual a la del eslabón más débil

Amenazas

- **Interceptación** (acceso para lectura). Rompe la *confidencialidad*.
- **Interrupción** (eliminación). Rompe la *disponibilidad*.
- **Modificación** (acceso para escritura). Rompe la *integridad* y la *autenticidad*.
- **Fabricación** (creación). Puede romper diferentes aspectos.

Políticas de seguridad

Antes de abordar los aspectos técnicos, es necesario definir una *política* que determine cosas como:

- ¿Qué servicios, datos o recursos estarán protegidos?
- ¿Qué aplicaciones tendrán permitido su acceso?
- ¿Qué tipo de acceso estará permitido? (lectura, escritura, creación)
- ¿Qué usuarios tendrán permitida cada tipo de acceso?
- etc.

Una vez definida la *política* se puede pensar qué *mecanismos* usar para hacerla valer.

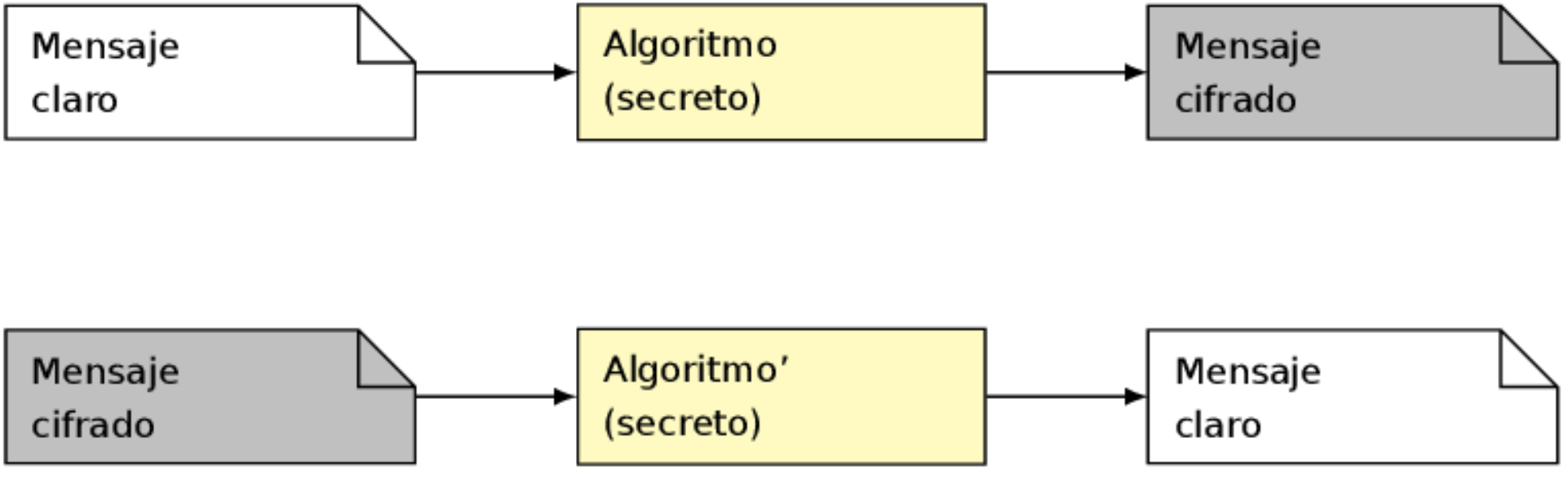
Mecanismos de seguridad

- **Criptografía**. Permite cifrar la información para hacerla incomprensible al usuario no autorizado.
- **Control de acceso**. Permite limitar qué usuarios o entidades tienen acceso a qué recursos.
- **Autenticación**. Permite verificar la identidad de usuarios o entidades.
- **Auditoría**. Permite registrar y almacenar todas las incidencias de seguridad.

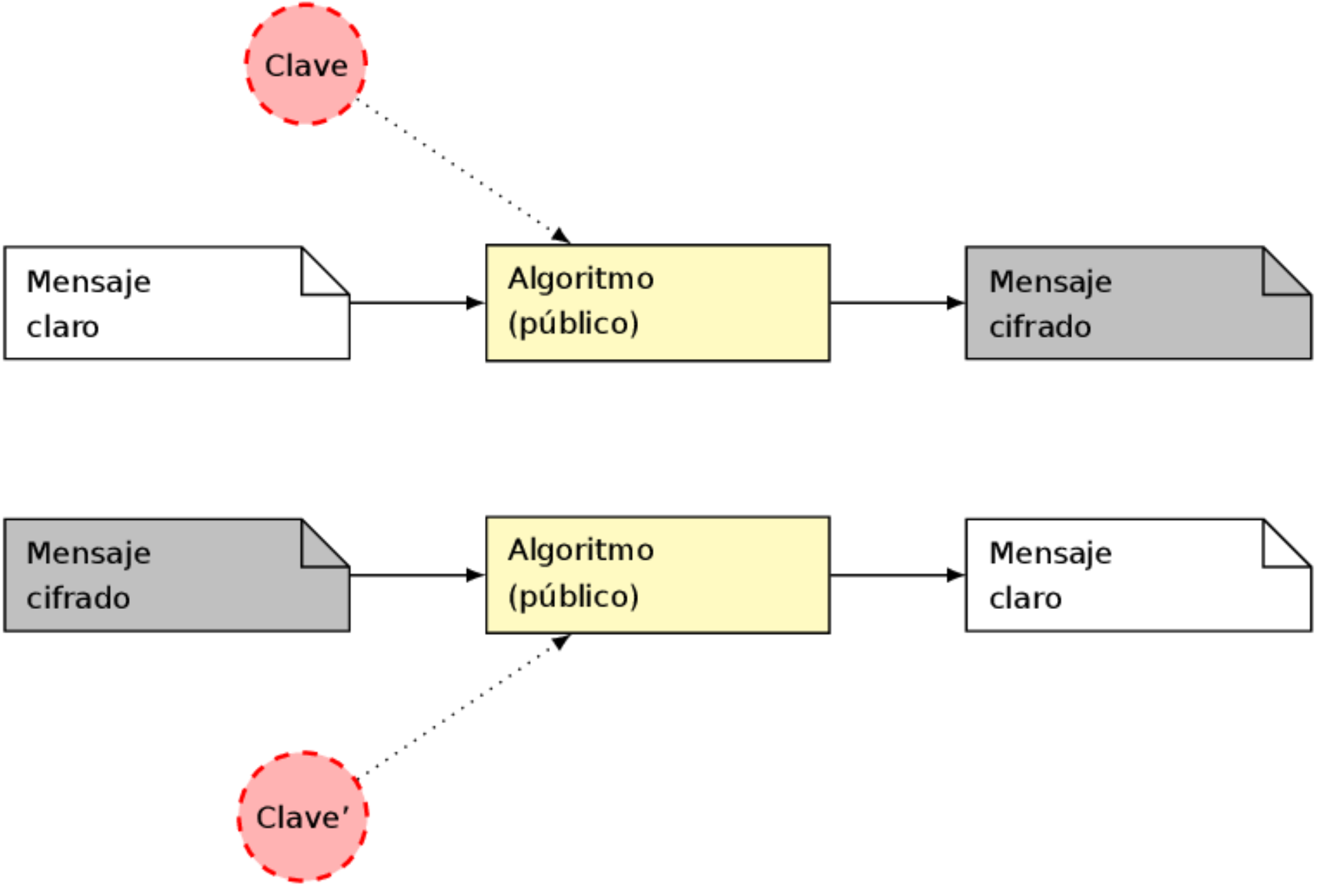
Nos centraremos seguidamente en la criptografía.

Criptografía

En los orígenes



En la actualidad



La clave

- No es lo mismo que una *contraseña* (key $\neq$ password)
- Los algoritmos criptográficos garantizan que **la clave no puede obtenerse** a partir del mensaje cifrado, ni aún teniendo la versión sin cifrar.
- Si el algoritmo criptográfico es seguro, la única forma de encontrar la clave es la ***fuerza bruta***.
  - Es decir: probar una a una todas las claves posibles

Tipos de criptografía

- **Simétrica** (*clave compartida* o *clave única*):
  - Se usa la misma clave para cifrar y descifrar
  - (`Clave` y `Clave'` son la misma).
  - Ejemplos: DES, AES, Blowfish, RC4
- **Asimétrica** (*pareja de claves privada/pública*):
  - Hay dos claves, emparentadas matemáticamente. Lo que una cifra, la otra lo descifra
  - (`Clave` y `Clave'` se generan a la vez con un algoritmo específico)
  - Ejemplos: RSA, ElGamal

Hash criptográfico

- Es un tipo de *función hash*. Estas funciones convierten una secuencia de bytes de longitud arbitraria en otra de tamaño fijo.
- **No es un sistema de cifrado** pues no permite recuperar la información (*picadora de carne*).
- Tiene muchos usos:
  - Verificar la *integridad*
  - Simplificar otras operaciones (al reducir el volumen de datos)
  - Evitar guardar contraseñas
  - etc.

Propiedades de un hash criptográfico

Es una función *hash* $H()$ que cumple:

- Dado un valor h no es posible encontrar un mensaje m' tal que $h = H(m')$
- **Resistencia débil a la colisión**,
  conocidos m y $h = H(m)$ no es posible encontrar otro m' tal que $h = H(m')$
- **Resistencia fuerte a la colisión** no es posible generar dos valores m y m' tales que $H(m) = H(m')$

Ejemplos: MD5, SHA

Criptografía de clave simétrica

- **Orientada a bloques**.
  - El mensaje M se divide en bloques de n bits.
  - Cada bloque se cifra usando el mismo algoritmo, que involucra a la clave.
  - El método suele ser "barajar los bytes" en una serie de pasos.
- **Orientada a flujo de bytes**
  - Se genera una secuencia pseudo-aleatoria de bytes.
  - Para ello se usa la clave.
  - Cada byte del mensaje se "mezcla" con un byte de esa secuencia (p.ej: usando `XOR`)

Ventajas e inconvenientes

En la criptografía de clave simétrica:

- El tipo de operaciones que usa son muy rápidas de ejecutar
- Se pueden (y suelen) implementar en hardware $\Rightarrow$ más velocidad
- Es el mecanismo de cifrado más eficiente

Pero...¿cómo comunicar la clave compartida?

Se requiere un canal seguro.

Criptografía de clave asimétrica

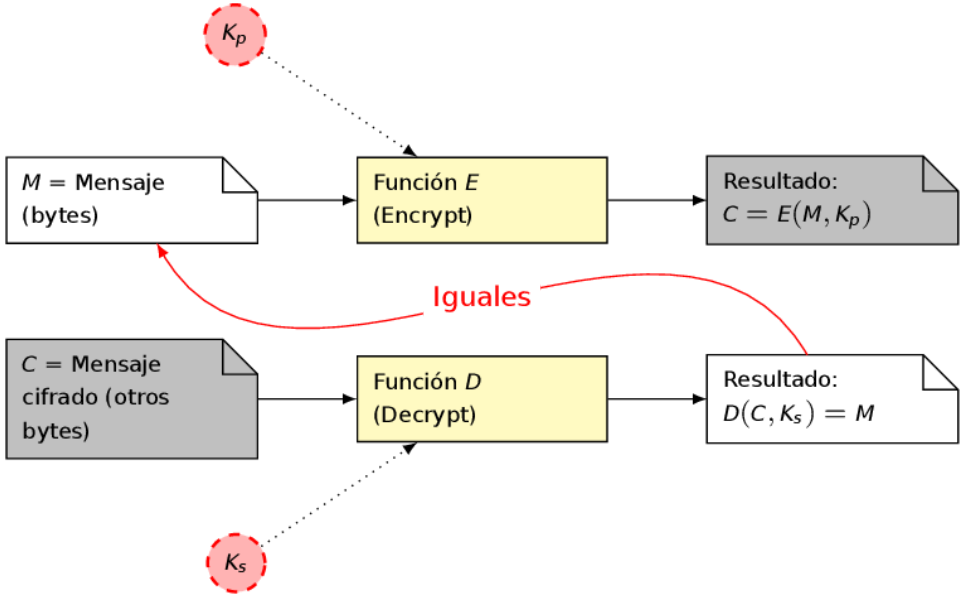
- Usa dos claves K_s y K_p, generadas a la vez mediante un cierto algoritmo
- El mecanismo de generación garantiza que:
- Lo que una cifra, la otra lo descifra.

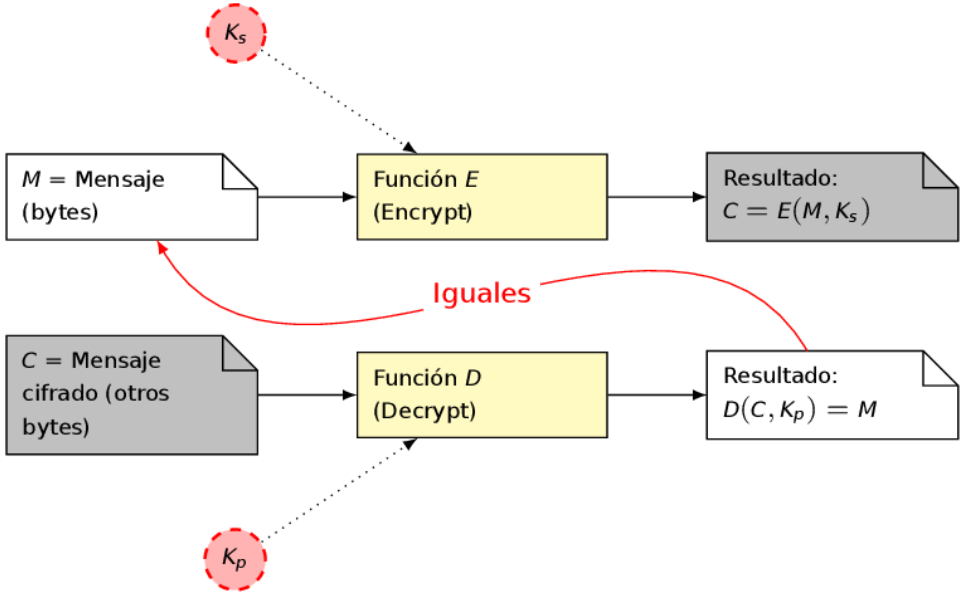
Es decir, si:

- $E(m, k)$ es la función que cifra el mensaje m con la clave k
- $D(m', k')$ la que descifra el mensaje m' con la clave k'

Entonces se cumple:

- $D(E(M, K_p), K_s) == M$
- $D(E(M, K_s), K_p) == M$


K_p
M = Mensaje (bytes)
Función E (Encrypt)
Resultado: $C = E(M, K_p)$
Iguales
C = Mensaje cifrado (otros bytes)
Función D (Decrypt)
Resultado: $D(C, K_s) = M$
K_s


K_s
M = Mensaje (bytes)
Función E (Encrypt)
Resultado: $C = E(M, K_s)$
Iguales
C = Mensaje cifrado (otros bytes)
Función D (Decrypt)
Resultado: $D(C, K_p) = M$
K_p

Criptografía de clave asimétrica

Uso:

- A las claves K_s, K_p se les asigna arbitrariamente un rol:
  - Una de ellas será **privada** (digamos K_s) y jamás se comunica a nadie
  - La otra será **pública** (digamos K_p) y se comunica a todo el mundo

K_p puede comunicarse por cualquier canal, seguro o inseguro (email, página web...)

Un atacante puede entonces tener acceso a K_p, pero nunca a K_s.

Uso para cifrado

- Para uso particular (la misma persona cifra y descifra):
  - Se cifra un archivo con K_p
  - Sólo quien tenga K_s podrá descifrarlo.
  - No importa que otras personas tengan K_p, no pueden usarla para descifrar.

Pero...¿Y si quiero enviar un archivo cifrado a alguien?

¿Con qué clave lo cifro?

Envío de mensajes cifrados

- El mensaje se cifra con la K_p **del destinatario**
- Que podrá descifrarlo usando su propia K_s

Por tanto:

- Para poder enviar mensajes a X necesito la K_p de X
- Pero puedo obtenerla fácilmente (es pública!)

¿Se te ocurre algún riesgo si X me la transmite por canal inseguro?

Volveremos sobre esto

Uso para firma digital

Si cifro un archivo con mi K_s

- No estoy protegiendo su confidencialidad
- Ya que cualquiera que tenga mi K_p puede descifrarlo
- Pero sí tiene una garantía de **autenticidad**,
- pues sólo el propietario de K_s ha podido cifrarlo

En lugar de cifrar el archivo completo, basta firmar su *hash criptográfico* que es más corto.

> Esto se denomina **firma digital**

Verificación de la firma digital

La firma digital de un archivo M es un *hash* del archivo, $H(M)$, cifrado con una clave privada K_s.

$$\text{Firma} = E(H(M), K_s)$$

Dado el archivo M y su Firma ¿cómo se verifica que la firma es correcta?

- Se requiere K_p (pareja de K_s)
- Se verifica que se cumpla:

$$D(\text{Firma}, K_p) == H(M)$$

Comunicación de la clave pública

- Para **cifrar** mensajes necesito la clave pública **del destinatario**
- Para verificar **firmas** digitales necesito la clave pública **del firmante**
- Análogamente, para que otros puedan enviarme mensajes cifrados o verificar mis firmas, necesitarán *mi clave pública*.

¿Cómo difundir K_p? ¿Existe algún riesgo si se usa un canal inseguro?

Ej: email, página web, twitter, foro público

Comunicación de la claves pública

Sí existe un riesgo si las K_p se difunden por canal inseguro

- Pero no es un riesgo de *confidencialidad*
  (no importa que otros conozcan mi K_p)
- Es un riesgo de *integridad*
- Alguien podría modificar mi K_p
- ¿Y entonces...?

El ataque del *hombre en el medio*: confidencialidad

1. A envía su K_{Ap} a B para que B pueda enviarle un mensaje secreto
2. C, en medio de la comunicación obtiene K_{Ap}
3. Y modifica el mensaje haciéndose pasar por A pero incluyendo K_{Cp}
4. B recibe lo que cree K_{Ap} y lo usa para cifrar M
5. B envía el resultado de $E(M, K_{Cp})$, C lo intercepta
6. Usando K_{Cs}, C descifra el mensaje y obtiene M
7. C vuelve a cifrar el mensaje con K_{Ap} y se lo envía a A
8. A lo descifra con K_{As} y no nota nada raro

C está leyendo los mensajes secretos para A

El ataque del *hombre en el medio*: autenticidad e integridad

Tras el ataque anterior, C puede *suplantar* a A

1. A escribe un mensaje y lo firma digitalmente con K_{As}
2. C intercepta el mensaje, lo descifra con K_{Ap}
3. C altera el mensaje para que diga algo completamente diferente
4. C firma el nuevo mensaje con su K_{Cs} y se lo envía a B
5. B verifica la firma con K_{Cp}
6. Ya que B cree que K_{Cp} es la clave pública de A, queda convencido de la autenticidad e integridad del mensaje.
7. En cualquier momento C puede crear un nuevo mensaje haciéndose pasar por A.

Certificación

Para evitar el ataque anterior, la copia de K_{Ap} que recibe B debe estar **certificada**, lo que puede ocurrir de dos formas:

- B ha recibido K_{Ap} por un canal seguro, directamente de A
- B ha recibido K_{Ap} por un canal seguro, directamente de otro **intermediario** D, quien *certifica* que es auténtica.

Para el segundo caso, además debe darse que:

- B confía en D como certificador
- D ha obtenido K_{Ap} directamente de A o de otro certificador en quien confíe y que se lo haya certificado.

Implementación de la certificación

Un certificado de A emitido por X es la clave pública de A firmada digitalmente por X, es decir:

$$\text{Certificado}_{A,X} = E(H(K_{Ap}), K_{Xs})$$

Confiamos en que X verifica la identidad de A y la validez de K_{Ap} antes de estampar su firma.

Si recibo $\text{Certificado}_{A,X}$ ¿cómo puedo verificar que es válido?

Verificación del certificado

Se validaría igual que una firma digital, es decir, dado K_{Xp} se comprobaría si

$$D(\text{Certificado}_{A,X}, K_{Xp}) == K_{Ap}$$

Por tanto

- Para validar un certificado emitido por X necesitamos K_{Xp}
- ¿Cómo obtener K_{Xp} evitando "el hombre de enmedio"?
- ¿A través de otro certificado emitido por Y?
- Entonces necesitaré K_{Yp}
- ¿Cómo lo obtengo? etc...

¿No es esto un problema recursivo? ¿Dónde se acaba?

Certificados-raíz

- Un certificado-raiz es una K_p *autofirmada* (usando su propia K_x)
- Emitida por un organismo de confianza (FNMT, Verisign)
- Pero que llega a nosotros por canal seguro
- Y por tanto *confiamos* en esa K_p para validar otros certificados.

Los certificados-raíz vienen formando parte del operativo o de cierto *software* como los navegadores.

Se supone que el soporte de instalación de ese *software* ha llegado por un canal seguro.

Conclusión

- La base matemática y las tecnologías criptográficas son muy maduros, seguros y confiables.
- Los conceptos subyacentes son muy complejos.
- El usuario medio no los entiende, y no da importancia a la seguridad.
  - Aceptar certificados inválidos
  - Caer en *phising*
  - `"password"` fue la contraseña más usada (entre las *hackeadas*) en 2011.
  - En 2019 fue `"123456"`, seguida de `"123456789"`, `"qwerty"` y `"password"`. La cosa no mejora.

> El eslabón más débil es el *usuario*