Lección 12
Seguridad y criptografía
Ejemplos de aplicación

Ejemplos de aplicaciones criptográficas

- Criptografía de clave simétrica
  - Kerberos
- Criptografía de clave pública
  - PGP, PKI
- Sistemas mixtos
  - SSH

Clave simétrica: Kerberos

- Desarrollado en los 80 en el MIT. La versión 5 (1993) se ha convertido en un estándar de Internet (RFC 1510).
- Se usa en Windows 2000, XP y Server 2003. Es el sistema de autenticación por defecto de DCE.
- Evita que las contraseñas viajen a través de la red (en claro o cifradas).

El nombre está basado en Cancerbero, el perro de la mitología griega que, con tres cabezas y el lomo erizado de serpientes, cuidada las puertas del Hades. Hércules le venció.

Objetos de Kerberos

- **Ticket**.- Elemento que prueba ante un servicio determinado, que un cliente se ha autenticado recientemente con Kerberos. Tienen tiempo de expiración.
- **Autenticación**.- Elemento encriptado con la clave de sesión apropiada que contiene el nombre del cliente y una marca temporal. Demuestra la identidad del usuario.
- **Clave de sesión**.- Clave secreta generada aleatoriamente por Kerberos y enviada a un cliente para su uso en una comunicación particular con algún servidor.

Elementos de Kerberos

- Un servidor Kerberos se conoce como **Centro de Distribución de Claves** (**KDC**, *Key Distribution Center).*
- Cada KDC ofrece dos servicios:
  - **Servicio de Autenticación** (**AS**, *Autentication Service)*.- Es el que se encarga de validar al usuario frente al sistema. Sustituye al login clásico.
  - **Servicio de Concesión de Tickets** (**TGS**, *Ticket Granting Service)*.- Emite tickets que autorizan al usuario para acceder a un servicio determinado.
- Un ticket Kerberos tiene un período de validez fijo que comienza en t1 y acaba en t2. El ticket que autoriza al cliente C a acceder al servicio S es de la forma:

$$\{C, S, t1, Kcs\}_{Ks} \rightarrow \{ticket(C,S)\}_{Ks}$$

Protocolo Kerberos. Autenticación

1. Al iniciar sesión, el cliente se autentifica con el AS y recibe un ticket para poder acceder al TGS:

Dirección	Mensaje	Contenido
C → A	C, T, n	El cliente C le pide a Kerberos un ticket para comunicarse con el TGS. La ocasión n contiene la fecha y la hora.
A → C	$\{K_{CT}, n\}_{Kc}$, $\{ticket(C,T)\}_{KT}$	A le devuelve el ticket que autoriza el acceso al TGS. K_{CT} es una clave de sesión para utilizar con T. Al retornar n cifrado para C, A demuestra ser quién dice ser.

Protocolo Kerberos. Acceso al TGS

2. El cliente solicita al TGS un ticket para acceder al servicio S:

Dirección	Mensaje	Contenido
C → T	$\{C,t\}_{K_{CT}}$, $\{ticket(C,T)\}_{K_T}$ S, n	C le pide a T un ticket para acceder a S. La clave de sesión a usar la envía en el ticket. $\{C,t\}_{K_{CT}}$ es un autenticador de C ante T. Incluye el tiempo para evitar reutilizaciones.
T → C	$\{K_{CS},n\}_{K_{CT}}$, $\{ticket(C,S)\}_{K_S}$	Si el ticket es válido, T genera una clave de sesión para usar entre C y S. También envía el ticket para S.

Protocolo Kerberos. Acceso a un servicio

3. El cliente accede al servicio S:

Dirección	Mensaje	Contenido
C → S	$\{C,t\}_{K_{CS}}$, $\{ticket(C,S)\}_{K_S}$ petición, n	C le pide acceso a S. La clave de sesión a usar la envía en el ticket. $\{C,t\}_{K_{CS}}$ es un autenticador de C ante S. Incluye el tiempo para evitar reutilizaciones.
S → C	$\{n\}_{K_{CS}}$	Para que el cliente se asegure de que S es auténtico, se debería devolver la ocasión cifrada con la clave de sesión. (Paso opcional: puede ir incluido en los mensajes de respuesta del servicio).

Clave asimétrica: PGP, GnuPG

- Este sistema está orientado a la comunicación entre personas (vía email)
- Se hace uso de una pareja de claves privada/pública para:
  - Cifrar mensajes secretos
  - Firmar mensajes públicos
  - Cifrar y firmar.
- Véanse los documentos de prácticas

Infraestructura de Clave Pública (PKI)

- Una PKI es un conjunto de hardware, software, políticas y procedimientos de seguridad que permiten enlazar claves públicas con identidades de usuario.
- Se garantiza la seguridad de operaciones como el cifrado, la firma digital y las transacciones electrónicas.
- La PKI permite a los usuarios autenticarse ante otros y usar la información de los Certificados de Identidad para cifrar/descifrar información o firmar digitalmente.
- El certificado de identidad contiene la clave pública del usuario y lo firma digitalmente (garantiza su autenticidad) una Entidad Certificadora.
- En España se utiliza una PKI basada en X.509 para implementar los Certificados de Identidad Personal y el nuevo DNI electrónico

Elementos de una PKI

- La Autoridad de Certificación(CA). Es la encargada de emitir y revocar certificados. Da legitimidad a la relación de una clave pública con la identidad del usuario. En España es la Fábrica Nacional de Moneda y Timbre (FNMT) y la Dirección General de la Policía (DNIe).
- La Autoridad de Registro(RA). Es la encargada de verificar el enlace entre los certificados y la identidad de los titulares. En España hay varias: Oficinas de la Agencia Tributaria, Oficinas del Instituto Nacional de la Seguridad Social, algunos Ayuntamientos, etc.
- El repositorio de certificados y el repositorio con la listas de revocación de certificados(CRL). Lugares donde se almacena la información.
- La Autoridad de Validación(VA). Comprueba la validez de los certificados digitales. En España, es la FNMT y el Ministerio de Administraciones Públicas.
- La Autoridad de sellado de tiempo(TSA). Se encarga de firmar documentos para garantizar su existencia antes de un determinado instante de tiempo.
- Los usuarios finales. Poseen un par de claves (privada y pública) y un certificado asociado a su clave pública.

X.509 El estándar de PKI

- Para que un sistema PKI sea efectivo y usable, los algoritmos que usa deben ser conocidos y accesibles a todos.
- X.509 es un estándar que, entre otras cosas, define el formato de los certificados de identificación y el algoritmo de verificación de dichos certificados.
- En X.509 se asume un sistema jerárquico estricto de Autoridades de Certificación. Difiere del sistema de redes de confianza de PGP.
  - Cada usuario tiene un certificado de identidad firmado por una CA.
  - Las Autoridades de Certificación a su vez tienen certificados de identidad firmados por otras CA de mayor nivel.
  - Dos usuarios pueden diferir en la CA que les firma su certificado.
  - Para que puedan confiar uno en el otro tienen que ascender por el árbol de CA's hasta que encuentran una Autoridad de Certificación en la que confíen.
  - El punto más alto del árbol de confianza son los Certificados Raíz. Es un certificado sin firmar o autofirmado. Creemos en él.

X.509 El estándar de PKI (II)

Los Certificados Raíz pueden venir preinstalados en el software.
Ejemplo Internet Explorer y los certificados de Verisign.

La sintaxis de los certificados X.509 se define con ASN.1. Tienen la siguiente estructura:

- **Certificado**
  - **Versión**
  - **Número de serie**
  - **ID del algoritmo**
  - **Emisor**
  - **Validez**
    - **No antes de**
    - **No después de**
  - **Tema**
  - **Tema información de clave pública**
    - **Algoritmo de clave pública**
    - **Tema clave pública**
  - **Identificador único de emisor (opcional)**
  - **Identificador único de tema (opcional)**
  - **Extensiones (optional)**
    - **...**
- **Algoritmo de certificado de firma**
- **Certificado de firma**

X.509 Ejemplo de certificado

```
Certificate:
    Data:
       Version: 1 (0x0)
       Serial Number: 7829 (0x1e95)
       Signature Algorithm: md5WithRSAEncryption
       Issuer: C=ZA, ST=Western Cape, L=Cape Town, O=Thawte Consulting cc,
               OU=Certification Services Division,
               CN=Thawte Server CA/Email=server-certs@thawte.com
       Validity
            Not Before: Jul 9 16:04:02 1998 GMT
            Not After : Jul 9 16:04:02 1999 GMT
       Subject: C=US, ST=Maryland, L=Pasadena, O=Brent Baccala, OU=FreeSoft,
                CN=www.freesoft.org/Email=baccala@freesoft.org
       Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (1024 bit)
               Modulus (1024 bit):
                  00:b4:31:98:0a:c4:bc:62:c1:88:aa:dc:b0:c8:bb:
                  33:35:19:d5:0c:64:b9:3d:41:b2:96:fc:f3:31:e1:
                  66:36:d0:8e:56:12:44:ba:75:eb:e8:1c:9c:5b:66:
                  70:33:52:14:c9:ec:4f:91:51:70:39:de:53:85:17:
                  16:94:6e:ee:f4:d5:6f:d5:ca:b3:47:5e:1b:0c:7b:
                  c5:cc:2b:6b:c1:90:c3:16:31:0d:bf:7a:c7:47:77:
                  8f:a0:21:c7:4c:d0:16:65:00:c1:0f:d7:b8:80:e3:
                  d2:75:6b:c1:ea:9e:5c:5c:ea:7d:c1:a1:10:bc:b8:
                  e8:35:1c:9e:27:52:7e:41:8f
               Exponent: 65537 (0x10001)
    Signature Algorithm: md5WithRSAEncryption
       93:5f:8f:5f:c5:af:bf:0a:ab:a5:6d:fb:24:5f:b6:59:5d:9d:
       92:2e:4a:1b:8b:ac:7d:99:17:5d:cd:19:f6:ad:ef:63:2f:92:
       ab:2f:4b:cf:0a:13:90:ee:2c:0e:43:03:be:f6:ea:8e:9c:67:
       d0:a2:40:03:f7:ef:6a:15:09:79:a9:46:ed:b7:16:1b:41:72:
       0d:19:aa:ad:dd:9a:df:ab:97:50:65:f5:5e:85:a6:ef:19:d1:
       5a:de:9d:ea:63:cd:cb:cc:6d:5d:01:85:b5:6d:c8:f3:d9:f7:
       8f:0e:fc:ba:1f:34:e9:96:6e:6c:cf:f2:ef:9b:bf:de:b5:22:
       68:9f
```

Autoridad Certificadora

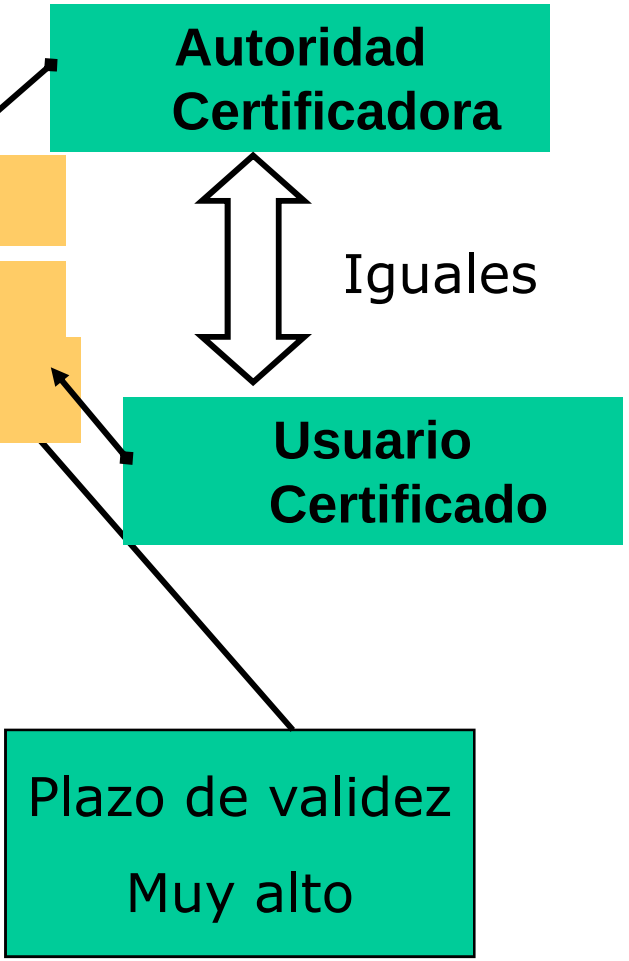
Usuario Certificado

Clave pública usuario

Firma Autoridad Certificadora

X.509 Ejemplo de certificado Raíz

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



Sistemas mixtos: SSH, OpenSSH

- Este sistema está orientado a la comunicación entre máquinas, para la ejecución de comandos remotos.
- Se hace uso de la criptografía de clave pública para:
  - Autenticar al servidor
  - Cifrar una clave simétrica de sesión
  - Opcionalmente, autenticar al usuario
- La criptografía de clave simétrica se usa para cifrar toda la comunicación.

SSH: Arquitectura

- **El protocolo de transporte**. Proporciona autenticación del servidor, confidencialidad (conexión cifrada) e integridad (verificación de los datos). Opcionalmente compresión. Funciona sobre TCP/IP, puerto 22.
- **El protocolo de autenticación del usuario**. Funciona sobre el anterior. Proporciona diferentes mecanismos mediante los que el usuario puede "demostrar" al servidor su identidad.
- **El protocolo de conexión**. Funciona sobre el anterior. Permite multiplexar una conexión cifrada (túnel), para que sea usada por varios canales lógicos (puertos/sockets).

SSH: Métodos criptográficos soportados

- Los algoritmos para cifrar la comunicación, verificar su integridad, calcular *hashes*, verificar identidades mediante claves públicas, etc. no están prefijados. Cliente y servidor se ponen de acuerdo en cuáles usar durante una **negociación inicial.**
- Algunos de los habituales:
  - Clave simétrica: AES, 3DES, Twofish, RC4, ...
  - Clave asimétrica: RSA, DSS, PGP
  - *Hashes*: MD5, SHA-1 (usados para verificar integridad mediante HMAC)

SSH: Protocolo

1. **Negociación**
   - Cliente y servidor se ponen de acuerdo en los algoritmos criptográficos que usarán
2. **Autenticación del servidor + clave sesión**
   - El cliente verifica la autenticidad del servidor (clave pública)
   - Se intercambian una clave secreta K
3. **Autenticación del cliente**
   - El servidor verifica la autenticidad del cliente (varios métodos)
4. **Sesión**
   - Todos los datos durante la sesión van cifrados con la clave secreta K
   - La clave secreta se puede renovar tras un cierto tráfico
   - El canal seguro así creado puede multiplexar varios canales lógicos

SSH: Autenticación del servidor

Se basa en **criptografía de clave pública**.

- El servidor tiene una pareja de claves K_s y K_p, y los clientes deben tener una copia de la pública.
  - Si el cliente no la tiene, el servidor se la puede enviar, pero es poco seguro.
- El servidor envía al cliente un mensaje firmado con K_s
- El cliente verifica la firma con K_p y así garantiza la autenticidad del servidor
- Cliente y servidor intercambian una clave secreta (K) con la que cifrarán el resto de las comunicaciones
  - Algoritmo Diffie-Hellman

SSH: Autenticación del cliente

- Mediante **criptografía de clave pública**.
  - Este método debe ser soportado obligatoriamente.
  - Cada cliente tiene una pareja de claves K_{Cs} y K_{Cp}.
    El servidor mantiene una lista de todas las K_{Cp}
  - El cliente se autentica enviando un mensaje pre-acordado (conteniendo el login y servicio solicitados), cifrado con K_{Cs}. También envía K_{Cp}
  - El servidor comprueba que la K_{Cp} recibida coincide con la que tiene almacenada en su lista y la usa para decodificar el mensaje y así validar al cliente.
- Mediante **contraseña.**
  - Este método es opcional (aunque es el más usado)
  - El cliente envía una contraseña que pide al usuario (el envío es seguro, pues va cifrado con K)
  - El servidor la valida (p.ej: frente a /etc/passwd)

SSH: Multiplexación del canal seguro

- Una vez autenticadas ambas partes, se crea un **canal seguro** a través del cual ejecutar aplicaciones remotas (típicamente un *shell*)
- El canal puede llevar también información de otras aplicaciones en la máquina cliente que quieran comunicarse con aplicaciones en la máquina servidora (*port forwarding ó tunneling*).
- **Ejemplo**:
  - En la máquina servidora hay un programa "MiServidor" escuchando en el puerto 12345
  - En la máquina cliente hay un programa "MiCliente" que quiere comunicarse con "MiServidor"
  - Queremos que la comunicación sea segura, a través del canal (túnel) creado por SSH

SSH: Multiplexación del canal seguro

- **Solución del ejemplo:**
  - Se elige un número de puerto no usado en la máquina cliente (p.ej: 7890)
  - Se configura SSH en la máquina cliente para que el puerto 7890 sea llevado por túnel al puerto 12345 de la máquina servidora.
  - Se ejecuta el programa “MiCliente” para que se conecte con localhost:7890
  - SSH transportará esa conexión al otro lado del túnel y “MiCliente” hablará con “MiServidor” a través del canal seguro.

SSH: Multiplexación del canal seguro