



DESDE LAS HORAS EXTRAS

CRIPTOGRAFÍA, CERTIFICADOS Y PRIVACIDAD EN INTERNET

Apuntes del curso

Desde las horas extras

Realizado por
José Luis Bautista Martín

Abril de 2022



Esta hoja está en blanco para facilitar la impresión de este documento a doble página.



ACERCA DEL INSTRUCTOR

José Luis Bautista Martín, Ingeniero de Sistemas, con maestría en “investigación en ingeniería de Software”.

Mi experiencia laboral en cuanto a desarrollo de software abarca desde tecnologías “legacy”, hasta tecnologías de vanguardia, poniendo siempre una especial atención en la construcción de software escalable, modular y sostenible.

Igualmente estoy especializado en la interconexión de diversos sistemas y plataformas para conseguir una solución coherente entre la tecnología actual en producción y nuevas tecnologías del mercado.

Una de mis inquietudes actuales es simplificar el desarrollo de software, permitiendo mediante herramientas generadoras de código, patrones de software, programación orientada a aspectos o simplemente interfaces sencillas y claras que el programador se concentre en resolver los problemas propios de la solución a implementar (esto es, los requisitos de negocio a representar en forma de software) y no se tenga que preocupar de tareas repetitivas, generalidades de los sistemas, o problemas técnicos, que no hacen más que distraerle de sus verdaderos objetivos.



ACERCA DE ESTE DOCUMENTO

La misión de este documento es exponer los objetivos, mecánica y temerarios planteados, así como la documentación para el curso “Criptografía, Certificados y privacidad en internet”.

OBJETIVOS DEL CURSO

Este curso tiene como objetivo, ser una introducción acerca de la realidad de la privacidad y seguridad en Internet, revisando los motivos y circunstancia (actuales) en la que esta es violada, para posteriormente presentar las herramientas de criptografía, y a los certificados como los medios actuales para garantizarla.



CONTENIDO

Acerca del instructor	3
Acerca de este documento	4
Objetivos del curso	4
Contenido	5
Metodología	7
Requisitos	7
Introducción	8
Breve historia de las criptografía	9
Breve historia de Internet	12
Seguridad en internet.....	15
Privacidad en internet.....	16
Caso espionaje NSA	17
Caso del teléfono el terrorista de San Bernardino.....	17
Caso Cambridge Analítica	18
Seguridad y factores de autenticación. Casos de Fraude y Vulnerabilidades.....	19
Diferencia entre privado y secreto	20
Factores de autenticación.....	20
¿Dónde están los “malos”?	30
Objetivos de la criptografía.....	32
Integridad	33
Confidencialidad.....	34
Identidad.....	34
Tipos de criptografía.....	35
Criptografía simétrica	35
Criptografía asimétrica	36
Certificados digital.....	40
Elementos involucrados en un certificado.....	42
Proceso para generar un certificado	44
Uso de un certificado	45
Autenticación de operaciones.....	46
características de un certificado.....	48
Elementos de un certificado	49
Generacion de CA con OpenSSL	55



Entidades certificadoras.....	55
Entidades intermediarias.....	56
Certificados personales	57
Configuración de certificados con GnuPG	58
Almacén de certificados	66
Almacén de certificado de Windows	66
Java Keystore	67
Resumen de operaciones con OpenSSL	72
Generar un certificado	72
Revocar un certificado	73
Renovar con la misma clave.....	73
Plantilla.....	74
Solicitud de certificados a Entidad Certificadora de un Directorio Activo	77
Solicitud de certificados desde IIS.....	79
Anexo I: Ejemplos de certificados	81
Anexo II: Extensiones de archivos criptograficos	82



METODOLOGÍA

Sera una conferencia de cuatro horas, en la que la primera hora se comentará sobre las circunstancias actuales de la seguridad en Internet, y la segunda tendrá un escenario más práctico al usar las herramientas de criptografía y generación de certificados.

REQUISITOS

No requiere ningún requisito en especial, si los asistentes quieren acompañar al instructor en la realización de los ejemplos de criptografía, deberán llevar una laptop, pero no es imprescindible.



INTRODUCCIÓN

- ¿Qué es una información secreta?
- ¿Qué es una información privada?

¿Qué es una información secreta?

Un secreto es una información que no debe ser compartida en prácticamente ningún ámbito. Pueden considerarse secretos elementos tales como NIP o contraseña.

¿Qué es una información Privada?

Una información privada, es aquella que, aunque no se es publica, se puede proporcionar en algunos ámbitos restringidos. Algunos ejemplos significativos son:

- Datos económica personales (como el salario o el patrimonio personal)
- Datos familiares (información de los hijos tal como escuelas, y horarios).
- Datos de índole médica.

Mucha de nuestra información (secreta y/o privada) viaja a través de internet, a diario y frecuentemente sin que seamos constantes de ellos.

Este curso tiene como objetivo ser una introducción acerca de la realidad de la privacidad y seguridad en Internet, revisando los motivos y circunstancia (actuales) en la que esta es violada, para posteriormente presentar herramientas de criptografía, y manejo de certificados.



BREVE HISTORIA DE LAS CRIPTOGRAFÍA

La palabra "criptografía" se compone de dos partes "cripto", que quiere decir oculto y "grafía", que quiere decir escritura.

La criptografía garantiza que un mensaje solo puede ser entendido por su destinatario, aunque otras personas puedan ver o conseguir dicho mensaje.

El origen de la criptografía se asocia frecuentemente con la política y con la guerra.

Uno de los algoritmos clásicos más populares es el de Julio Cesar, que consiste en desplazar tres posiciones en el abecedario cada letra.

En la historia moderna, la criptografía se ha usado ampliamente en los dos escenarios bélicos más representativos del siglo XX, pero sobre todo en la segunda guerra mundial.

La representación más importante de la criptografía en la segunda guerra mundial es la maquina enigma:

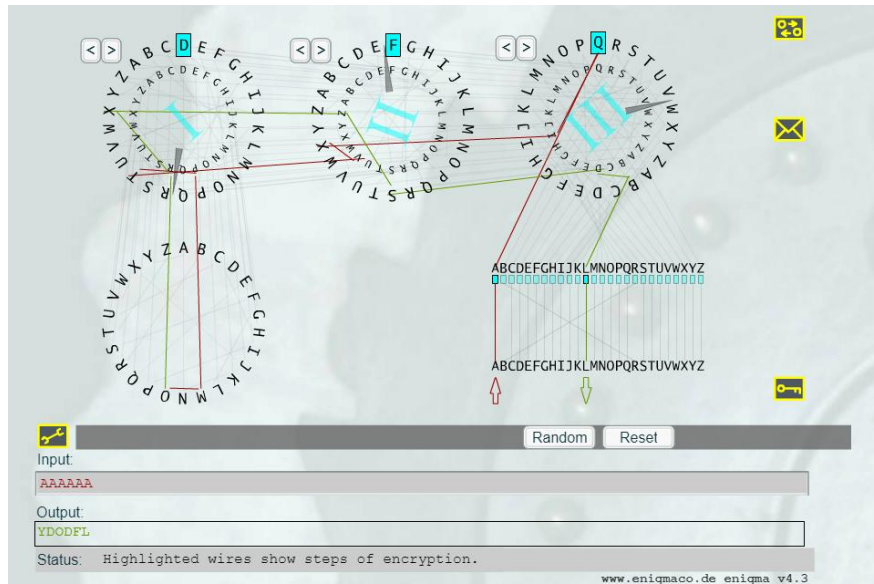


Es una maquina electromecánica de cifrado.

Tiene unos rotores en la maquina superior con el abecedario y un cableado que los conectaba.

Cada vez que pulso una tecla, gira un rotor (de forma parecida a un tacómetro), cambiando la configuración.

Pudiera generar unos 159 trillones combinaciones de encriptación para un texto.



- **Simulador enigma**

<http://enigmaco.de/enigma/enigma.html>

- **Explicación de la maquina enigma**

https://www.youtube.com/watch?v=XK_1gUo8YDE



BREVE HISTORIA DE INTERNET

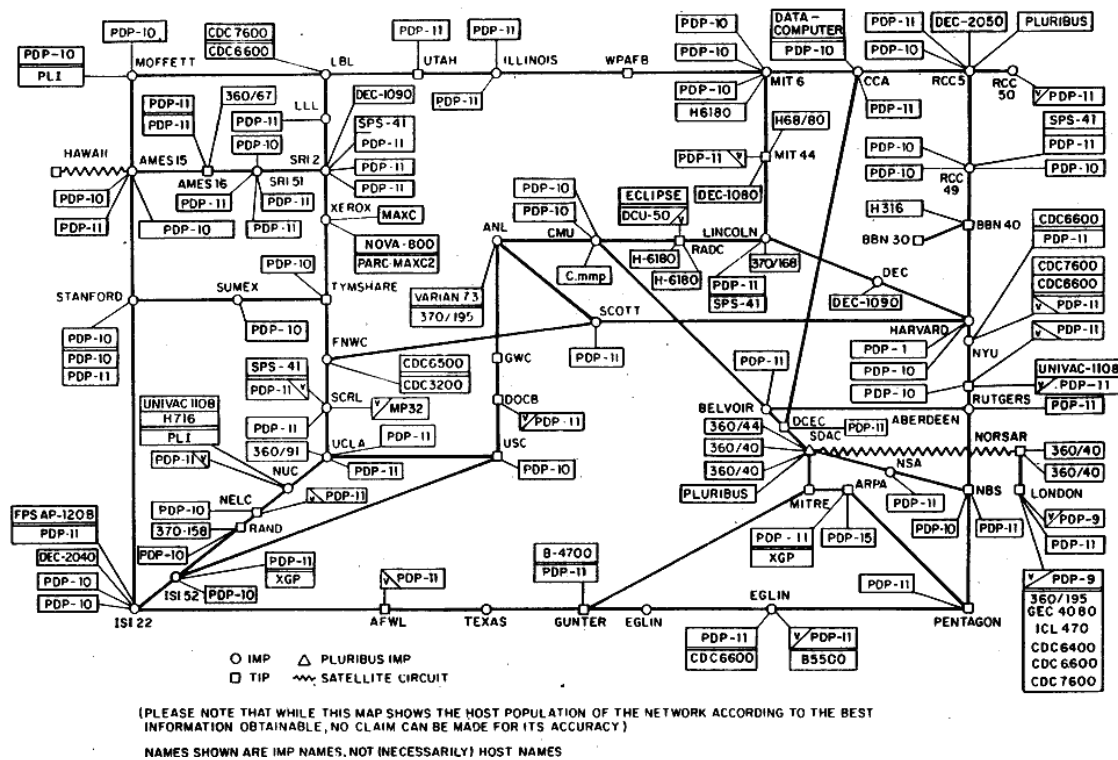
La semilla de lo hoy conocemos por internet, tiene su origen en plena guerra fría (conflicto entre, principalmente estados Unidos y la Unión Soviética, en distintos niveles, social, político, económico, militar o científico, con distintos periodos de intensidad se alargó desde finales de la segunda guerra mundial hasta el año 91).

Algunos hechos cronológicos que se dieron en este periodo son:

- **1962:** Crisis de los misiles.
- **1966:** Planeación de ARPANET (Advanced Research Projects Agency Network, es decir, la Red de la Agencia de Proyectos de Investigación Avanzada), los objetivos son:
 - El uso de una red descentralizada con múltiples caminos entre dos puntos.
 - La división de mensajes completos en fragmentos que seguirían caminos distintos.
- **1969:** ARPANET transporta sus primeros paquetes
- **1977:** Ejemplo de mapa (completo de Internet):



ARPANET LOGICAL MAP, MARCH 1977



- **1981:** Implementación del protocolo TCP/IP
- **1989:** Caída del muro de Berlín.
- **1990:** Desaparece ARPANET y comienza el giro a lo que conocemos actualmente como Internet (comercialmente hablando), en la misma época se considera el fin de la guerra fría.
- **1991:** Fecha que se considera el fin guerra fría.

Es muy posible que el miedo inicial a un ataque nuclear definiera él como debiera ser la arquitectura de una red de comunicaciones descentralizada que siempre estuviera disponible aun incluso cuando se atacaran nodos de la red.

- **1992 (aproximadamente):** Se crea la WWW.
- **1995:** Difusión pública de internet a través de proveedores de internet.



- **1998:** Se crea Google.
- **2001:** Explosión de la burbuja .com
- **2001:** Wikipedia.
- **2004:** Se crea Facebook.
- **2005:** Creación del (concepto) de la web 2.0, el internet enfocado en servicios y comienza la época de la banca electrónica
- **2007:** Se presenta el iPhone y con el comienza la era de los smartphones entre los usuarios comunes (no empresariales).

En la actualidad es común compartir en internet información personal (incluyendo la familiar) en Internet (Facebook, Instagram, Twitter), y usar la banca móvil (o digital). Es un desafío que esta información siga siendo privada y que no se use de forma perjudicial para nosotros.



SEGURIDAD EN INTERNET

El diseño de internet y los protocolos que lo sustentan de origen, no están pensando para garantizar la privacidad.

Los paquetes TCP/IP (el protocolo base de internet), eran recibidos por todos los nodos de una red. Cada equipo decidía si el paquete era para él o lo desechaba. Esto implica que cualquier nodo pudiera analizar los envíos de una red (a esto se llama “modo promiscuo”).

De origen, protocolos como la recepción de emails, ftp, incluso páginas web, no contemplaban seguridad exponiendo todo el flujo de información a cualquier persona que estuviera conectada a la red.

Es un tanto curioso que, en algún momento, se separen la necesidad de disponibilidad, de la necesidad de privacidad en la red, incluso más si consideramos el temprano uso de la criptografía de forma militar.

En la actualidad la criptografía nos ayuda en los siguientes aspectos:

- Asegurar la identidad del emisor de la información y del receptor (se identifican mutuamente, mediante un proceso de autenticación).
- Proteger la privacidad de los mensajes que se transfieren por una red (solo él envía la información y el destinatario de esta son capaces de comprenderla apropiadamente).
- Asegurar la integridad de la información, es asegurar que la información que estamos compartiendo no ha sido manipulada de alguna forma.
- Poder garantizar irrefutablemente el responsable de un mensaje, mediante firmas electrónicas.



PRIVACIDAD EN INTERNET.

En la actualidad es prioritario garantizar nuestra privacidad en internet. El mal uso de nuestra información personal nos pone en situaciones de riesgo económico y de otra índole, como físico.

Mucha gente comparte en Facebook su perfil de forma pública, sin siquiera enterarse, proporcionando información de sus horarios, hábitos e incluso hijos y familiares que no pueden defenderse por sí mismo, aceptamos “amigos” que son prácticamente desconocidos para nosotros, y con los que compartimos información sin tener una plena confianza real en ellos.

En cuanto a nuestras operaciones bancarias, es posible que seamos víctimas de fraude, que nos conectemos a páginas en las que se hagan pasar por nuestras páginas y extraiga información de nuestras cuentas o de nuestra identidad (y hagan operaciones nuestro nombre), también es posible que nos estemos conectando a la página real pero nuestra información sea interceptada, recolectada y manipulada.

Otro posible uso de nuestra información es integrándola en un sistema de Big Data. Se recolectan nuestros datos, la forma de usar servicios, e incluso nuestra ubicación. Mediante esto sistemas como Amazon, Netflix, o Facebook, nos da sugerencias basada no solo en nuestros gustos, sino en los gustos de personas que tengan un perfil parecido al nuestro. Esto es realmente muy práctico, pero en cierta forma, cuando el sistema privilegia cierta información en beneficio de un vendedor (o un candidato político, por ejemplo), es muy posible que nos demos cuentas, y que seamos manipulados para tomar acciones que de otra forma no tomaríamos.

A continuación, una serie de ejemplos de uso de nuestros datos de sin nuestra autorización



Caso espionaje NSA

En junio de 2013, se difundió que el FBI y la NSA, recolectaba datos directamente de los servidores de Microsoft, Yahoo, Google, Facebook, PalTalk, AOL, Skype, YouTube y Apple.

Más información:

<https://www.20minutos.es/noticia/1850380/0/caso-snowden/cronologia/espionaje-ee-uu/>

En diciembre del 2013, se rebeló que la NSA, pago 10 millones para que la librería de encriptación de RSA BSAFE (en el 2004).

Caso del teléfono el terrorista de San Bernardino

El tiroteo de San Bernardino ocurrió el 2 de diciembre de 2015, a las 10:59 de la mañana (UTC -8) en el Inland Regional Center en San Bernardino, California, que resultó 14 muertos y 21 heridos al menos

En febrero de 2016, una jueza pidió a Apple que ayuda desbloquear el teléfono móvil del terrorista implicado.

Apple se negó alegando que los que estaba pidiendo era una herramienta que pudiera desbloquear cualquier virtualmente cualquier iPhone,



Caso Cambridge Analítica

Cambridge Analítica era empresa dedica a la segmentación publicitaria, esto es dar mensajes extremadamente personalizados a un sector en particular de la población, a veces influenciando su opinión gracias a la recolección previa de datos del individuo.

En el 2014, Cambridge hizo una prueba de personalidad (aplicación de Facebook), la cual solicita permiso para acceder a todos los datos del perfil, y la gente inocentemente se los proporcionaba.

Facebook permitía acceder a la información de los usuarios (y de sus amigos), para uso académico (por sus políticas), Cambridge Analítica recolecto ilícitamente estos datos, y lo uso para influir en la opinión política de la población.

La mala praxis de Facebook era que una vez que conoció la fuga de datos, no la notifico, ni la prohibió.

Más información:

<https://www.xataka.com/privacidad/el-escandalo-de-cambridge-analytica-resume-todo-lo-que-esta-terriblemente-mal-con-facebook>



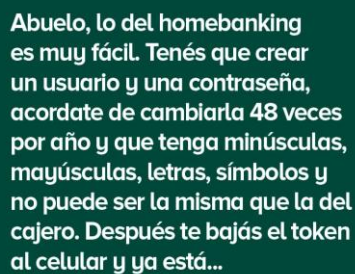
SEGURIDAD Y FACTORES DE AUTENTIFICACIÓN. CASOS DE FRAUDE Y VULNERABILIDADES

¿Por qué cada vez es más complicado seleccionar un password y hay cada vez más medidas y controles de seguridad?

Esto es porque dependemos más de “almacenar” y organizar nuestra vida en medios digitales, y estos datos privados adquiere un cierto valor para usarse de forma maliciosa de diversas formas, como por ejemplo robarnos el dinero que tenemos en nuestras cuentas bancarias.

Los formas tradicionales de protección se hacen obsoletos muy rápido, y las formas de obtener nuestros datos, se vuelve cada vez mas ingeniosas, y se alejan de un clásico ataque de película de hacker, en la que se usan puertas traseras o complicados códigos. La mayoría de los ataques se bajan en ingeniería social aprovechándose de la inocencia e ingenuidad (a veces de la avaricia), de los usuarios.

Vamos a analizar las formas de proteger nuestros sistemas y algunos ataques clásicos que los vulneran.



Abuelo, lo del homebanking es muy fácil. Tenés que crear un usuario y una contraseña, acordate de cambiarla 48 veces por año y que tenga minúsculas, mayúsculas, letras, símbolos y no puede ser la misma que la del cajero. Después te bajás el token al celular y ya está...

Coherencia por favor ☹️🙄



Diferencia entre privado y secreto

Ante de nada hagamos una diferencia entre que es privado y que es secreto:

- **Privado:** Es un dato que solo debemos compartir con personas autorizadas, cuanto menos personas sepan este dato, más seguro es. Generalmente son datos que sirven para identificarnos y se complementan con un dato secreto.
- **Secreto:** Son datos que solo nosotros debemos conocer, no debemos proporcionarlos a nadie, y en el momento de usarlos debemos ser nosotros mismos los que los usemos a través de un sistema, sin ninguna iteración humana adicional.

Factores de autenticación

Los factores de autenticación son “categorías” que clasifican formas que tiene un usuario de autenticarse dentro de un sistema.

En la actualidad se debe combinar más de un factor de autenticación (más de una forma de autenticación) para garantizar la seguridad de un sistema.

Factor cero de autenticación. Algo que se sabe en común

La autenticación se basa en algo que el usuario (o cliente), y la institución (o empresa) conocen. Este dato debe ser privado, pero en muchos casos se acaba convirtiendo en un elemento público, algunos datos son:

- **Número de cuenta, cliente:** Deben ser privados pero generalmente no es así, ya que es el elemento con el cual generalmente nos identifican en un primer nivel y hay que compartir.
- **Números de tarjeta:** Debieran ser secretos, pero el simple de usarlas lo expone continuamente.



Una forma muy sencilla de obtener este dato por parte de empleados maliciosos es la siguiente:

Al momento de pagar, con papel de carbón o algún tipo de superficie que se marque con la presión, graban estos números simplemente apretando la tarjea con la superficie (recordemos que estos números tienen relieve), es algo muy rápido y pasa desapercibido porque puede estar oculto debajo de la terminal (o incluso en el pantalón) posteriormente con la excusa (o el gesto) de revisar la firma aprenden de memoria el CVE. De esta forma ya tienen el número de tarjeta, la fecha y el CVE, con lo que pueden realizar comprar por internet. Esto es muy común en comercios muy congestionados, donde la prisa se junta con el movimiento del empleado como gasolineras que no son de autoservicio.

¿Qué se recomienda en esto casos?:

- Tapar el CVE, con un poco de papel, o cinta, de forma que sea imposible, revisar este número.
- Nunca pagar con debito, sino con crédito. Es mucho más fácil ganar una reclamación cuando estas pagan con crédito (ya que el dinero nunca es tuyo y el banco se puede negar a pagar a un negocio fraudulento), que pagar con debito (donde si es tu dinero, y cuando lo pierdes, es más difícil reclamarlo).
- No activar la compra por internet con tarjetas físicas, sino con tarjetas digitales, que ofrezcan medidas de seguridad como un CVE cambiante, y otras medidas de personalización.
- Nunca pagar por teléfono con la tarjeta.
- Y sobre todo, nunca, nunca, perder de vista tu tarjeta, por ningún motivo, incluso, si el empleado se tarda mucho en cobrar, hay que pedir que se devuelva la tarjeta. Es común que en un escenario de fraude, el empleado haga que no entra la tarjeta y tiene que llamar a alguien o hacer una operación especial, para liberar el cobro, aprovechando para tomar los datos de la tarjeta.
- **Usuarios:** Un usuario que identifica al cliente. Aunque este dato generalmente no se tiene que compartir a una persona (porque es privado y se usa generalmente dentro de sistemas), suele ser conocido por la institución (en decir en ninguna



forma esta encriptado de forma no reversible) ya que es un elemento de referencia y seguimiento para distintas actividades.

- **Email o teléfono:** Se usan frecuentemente y cada vez más como medios de autenticación, debido a que son fáciles de recordar, pero son extremadamente públicos, con lo cual son el primer punto de entrada para un hackeo.

El problema de este factor de autenticación es que a pesar de sus distintos niveles de privacidad, es necesario que tanto la institución, como el usuario, intercambien de forma clara el “dato”, que ambos conocen para garantizar que el otro también lo conoce, pero al hacerlo comprometemos el dato en si, por que no sabemos si realmente la persona o contraparte con lo que estamos hablando es la intuición, el cliente o solo alguien que quiere obtener dicho dato (para efectos maliciosos).

Factor uno de autenticación. Algo que solo el cliente sabe.

Generalmente aquí estamos hablando de los password, pines y demás.

- **PIN (o NIP) numérico:** Es un valor numérico de cuatro o más números, que se usa para acceder a un servicio, como por ejemplo, el pago con una tarjeta.

No ofrece en si mucha seguridad, debido a que con cuatro números, tenemos solo mil combinaciones posibles (no que no es mucho), la seguridad viene en si en que después de un numero de intentos determinado, el dispositivo o la tarjeta se bloquea (en algunos casos puede borrarse la información), lo cual es realmente lo que proporciona la seguridad.

- **Password:** Es una palabra secreta que combina, letras, numero y símbolos, al igual que el pin, no se debe compartir bajo ningún concepto con otra persona.

Tanto el password como el PIN, se guardan (o se debiera) “encriptados” en los sistemas de seguridad, con una encriptación que se llama “de una sola vía” (conocido comúnmente como hash). Esto quiere decir que no se puede llevar al valor original (el



password), descriptando el valor guardado, porque es imposible, con lo que ni la institución sabe cuál es el password que estamos usando.

Cuanta más entropía (menos orden y menos lógica) tenga un password más seguro es, con lo que se establecen una serie de restricciones a su creación, generalmente son las siguientes (cuando más se cumpla, más seguro es el password):

- Una longitud mínima.
- Un uso mínimo de mayúsculas y minúsculas.
- Un uso mínimo de símbolos.
- Un uso mínimo de números.
- No permitir caracteres (o números) igual en secuencia, como “aaa”, o “1111”.
- No permitir caracteres consecutivos de forma ascendente o descendente, como “1234”, “abcd” o “dcdb”.
- No permitir palabras que se encuentren seguidas en el teclado como “qwerty”.

¿CUÁNTO TIEMPO TOMARÁ HACKEAR TU CONTRASEÑA?				
Número de Caracteres	Sólo números	Mayúsculas y minúsculas	Mayúsculas, minúsculas y números	Mayúsculas, minúsculas, números y símbolos
4	Al instante	Al instante	Al instante	Al instante
5	Al instante	Al instante	3 segundos	10 segundos
6	Al instante	8 segundos	3 minutos	13 minutos
7	Al instante	5 minutos	3 horas	17 horas
8	Al instante	3 horas	10 días	57 días
9	4 segundos	4 días	153 días	12 años
10	40 segundos	169 días	1 año	928 años
11	6 minutos	16 años	106 años	71 mil años
12	1 hora	600 años	6 mil años	5 millones de años
13	11 hora	21 mil años	108 mil años	423 millones de años
14	4 días	778 mil años	25 millones de años	5 billones de años



Una vez me preguntaron si hacer públicas esas reglas no hace que un ataque sea “más guiado” y por tanto hacer que sea más posible hackear un password. La respuesta es no, es mas hacer pública estas reglas y obligar a que sea cumplida por los usuarios (o clientes), son realmente un elemento disuasorio para los hackers, por que básicamente saben que con los medios actuales esas contraseñas son prácticamente inhackeables (aunque pueden obtenerlas mediante ingeniería social).

Factor tres de Autenticación. Algo que el cliente tiene

El usuario de autentifica con algo que solo él puede poseer, generalmente un elemento físico que se asocian a una identidad y nos permiten el acceso a una acción, característica o servicio. Algunos son:

- **Tarjetas:** Las típicas tarjetas que nos permite el acceso a zonas restringidas, o para realizar pagos con ellas.

Las tarjetas como elemento único de validación son extremadamente inseguras, deben usarse siempre junto con otro elemento.

En México, por ejemplo, era común en la década pasada que para pagar con tarjeta solo fuera necesario presentar la tarjeta, realizar el pago con el punto de venta y firmar el voucher, en teoría el empleado debiera validar la firma y la identidad con otra autenticación, pero a pesar de ser obligatorio, era a voluntad del empleado y comúnmente nunca se hacía.

Solo fue hasta hace poco que se comenzó a pedir el NIP de la tarjeta al realizar el pago (pero no para todas las tarjetas, ni bancos), de esta forma ya no queda a la voluntad del empleado el validar la identidad del cliente, sino que además es el cliente el que debe acreditarla, y se si equivoca varias veces, puede bloquear su tarjeta. Esta es una medida de protección fundamental para que la tarjeta no sea solo el medio de identificación posible para hacer un pago.



Pero nada sirve totalmente, una vez mas es más sencillo realizar fraudes mediante ingeniería social, que mediante complicados y herramientas tecnológicas, como ejemplo los siguientes escenarios:

Muchos Bancos no tiene personalizadas sus tarjetas de debito (no entiendo el motivo), con que no es fácil ubicar si una tarjeta es tuya entre dos tarjetas iguales del mismo banco, en base a eso, uno de los posibles fraudes es el siguiente:

Una persona parece consternada cerca de un cajero (esta es el timador uno), cuando otra (este el cliente real), va a usar dicho cajero, el timador se acerca y le dice “oiga no se olvide frotar su tarjeta”, cuando el cliente se muestre perplejo, el timador insiste que debe frotar la tarjeta por qué no funciona el cajero, haciendo que se sienta cada vez mas confundido. Cuando sienta que haya bajado la guardia, le toma la tarjeta rápidamente, y le enseña con gesto amable como debe frotarla y se la devuelve rápidamente, pero en el intervalo, le cambia la tarjeta por otra, al final se despide y le desea suerte, todo ocurre tan rápido que la única sensación que tiene el cliente, es que el sujeto es un poco raro y confiado.

Cuando el cliente vaya a usar su tarjeta y le pida el NIP, es donde se acerca el timador dos, para ver disimuladamente que números está marcando. De esta manera los timadores ya tiene por separado el NIP y la tarjeta, y el cliente está confundido sin saber que ha pasado. Dándoles tiempo a los sujetos de usar las tarjetas en un negocio para compra televisiones o algún elemento de valor, que ya tendrá un tercer timado, en la fila de la caja de un comercio cercano. Todo esto puede pasar en menos de diez minutos.

Otro timo que se puso de moda hace unos años, esta vez más sofisticado:



Un timador, con una base de datos de teléfonos y personas (que se puede comprar ilegalmente), se hace pasar por teleoperador de un banco con muchos clientes (lo que aumenta la posibilidad de encontrar un inocente cliente).

Llama a una víctima de sus lista, y le indica que su banco le realizó un cargo incorrecto (el teleoperador no sabe si el cliente tiene cuenta en un banco, solo es suerte y estadística).

Le dice “por su seguridad, voy a decirle los 8 primeros números de su tarjeta”, y aquí reside el engaño, los 8 primeros números de una tarjeta, solo indican a que banco pertenece la tarjeta, y el tipo de producto que es, esto quiere decir, que son iguales para todos los clientes.

El cliente al ver que le han dado algunos números de su tarjeta (que no aportan realmente nada), le proporciona al timador todos los datos de su tarjeta, lleno de confianza, al pensar que habla con su banco y no con un timador. Al final, el resultado, es que de una forma u otra, el cliente se queda sin su dinero, al haber dado sus datos privados al timador.

- **Tag (Etiqueta RFID) o incluso una matrícula:** Los coches pueden tener elementos como tag, que permiten la salida y entrada de vehículos (a veces incluso con reconocimiento óptico de matrículas).

Uno de los problemas de usar un tag (o la matrícula) asociado a un vehículo, es que generalmente están pegados al mismo vehículo, con lo que se puede decir que están unidos el medio de autenticación y la acción a realizar. Esto quiere decir que por ejemplo si alguien roba tu vehículo en un estacionamiento, podrá salir con el sin más problemas (por ese falta algo más que impida esta circunstancia como un guardia o cámaras de seguridad).

- **Un teléfono:** Se garantiza que la operación que se desea hacer es desde un teléfono específico, generalmente esto se consigue de una forma parecida a la siguiente:



1. Se genera un identificador único con datos del teléfono.
 2. Se generan claves criptografías (generalmente asimétricas).
 3. Se encriptan estos datos, usando algo mecanismo de protección como un password.
-
- **Un token o generador de números:** Son dispositivos que genera números y se usan en cada operación, por ejemplo, si estoy haciendo una operación bancaria se pedirá un número generado por este dispositivo, y si no es el número correcto, la operación no se realizarán.

Los dispositivos pueden ser físicos, como un aparato dedicado exclusivamente a esa función, o una aplicación instalada en un teléfono.

Estos dispositivos, de la siguiente forma:

1. Existe un servidor y dispositivo, ambos aislados uno de otro, sin comunicación alguna.
2. El servidor y dispositivo, comparte una misma llave de una determinada longitud llamada semilla.
3. Ambos generan números basándose, en la semilla (que comparten), y algún elemento independiente de ellos, como la fecha y hora exacta, esto genera un número, como el servidor y el dispositivo generan comparte la llave, debiera poder generar el mismo número, en el mismo momento (sin tener que estar conectado).

De esta forma el servidor sabe en cada momento que número debe generar el dispositivo y solicitárselo al usuarios para garantizar que tiene el dispositivo consigo.

Uno de los problemas es que el servidor (que está en la empresa que proporciona el servicio, como un banco) sabe cuáles son las semillas que generan el token, es decir es capaz de duplicar dichos números, con lo que en teoría es información que se comparte, y no es algo que tenga exclusivamente el cliente. En otras palabras quien tenga dichas semillas, se pudiera hacer pasar por el cliente.



Otro de los problemas del token, es que el número sirve para autorizar casi cualquier cosa, con lo que podemos ser engañados, para proporcionar nuestro número actual de token, y realizar operaciones fraudulentas.

Factor tres de Autenticación. Algo que el cliente es

Algo único que está ligado a la persona física del cliente, de forma inequívoca, algo que “el cliente es”, de forma inequívoca e irrepetible, como:

- **Firmas manuscrita:** es la clásica firma que hacemos de nuestro puño y letra como prueba de nuestra identidad. Es usual en contratos, y documentos impresos. Aunque se supone que es algo que solo nosotros podemos duplicar exactamente igual, generalmente puede ser copiada y no ofrece mucha seguridad (tal es el caso que en la realización de contratos generalmente hace falta una figura notarial que le de valor, y personas que funjan como testigos).
- **Elementos biométricos:** Son elementos que pertenecen a nuestro cuerpo, y que pueden ser validados por un sistema informático, como

Huellas: Son los surcos únicos que poseemos en la llama de los dedos.

Rostro: Nuestra cara como identificación, el problema es que frecuentemente nuestra cara cambia, con lo que es necesario renovar la autenticación.

Iris del ojo: Al igual que las huellas se supone que son elementos únicos en cada uno de nosotros.



El cambio hacia los biométricos ha sido bastante arduo, no siempre ha dado los resultados requeridos. Unos de los aspectos importantes de los biométricos (y por lo cual han sido hackeados mas de una vez), es que deben garantizar que la persona que está ofreciendo la muestra biométrica está viva y presente, por ejemplo en el caso de las caras es necesario garantizar que realmente estamos leyendo una cara, y no la foto de un sujeto, en el caso del iris, se deben observar movimientos involuntarios del ojo, para garantizarlo.

A veces el uso de biométricos puede ser frustrante, por ejemplo, Es común tener lectores de huella en los teléfonos móviles, estos lectores solemos entrenarlos con nuestra propias huellas y tener un porcentaje de éxito muy elevado, pero lectores empresariales (a veces de poco costo), que se usan másicamente en las empresas para identificar clientes, suelen dar multitud de quebraderos de cabeza.

Autenticación multifactorial

Como hemos visto cualquier solución de autenticación individual, es hackeable en cierto punto, casi siempre usando ingenio más que habilidades tecnológicas.

He aquí el por qué generalmente se usa más de una de forma de autenticación (lo que se conoce por autenticación multifactorial). Por si una es rota, quede la protección de la otra.

De misma forma las técnicas de fraude, avanzan más y de forma muy rápida, con lo que los sistemas se van mas forzados a endurecer sus mecanismos de seguridad, a expensas de la comodidad del usuario.

El principal enemigo del usuario, es su misma ingenuidad, es más fácil que el usuario proporcione sus datos de autenticación mediante engaños, que en si sea hackeado por sofisticados métodos informáticos.



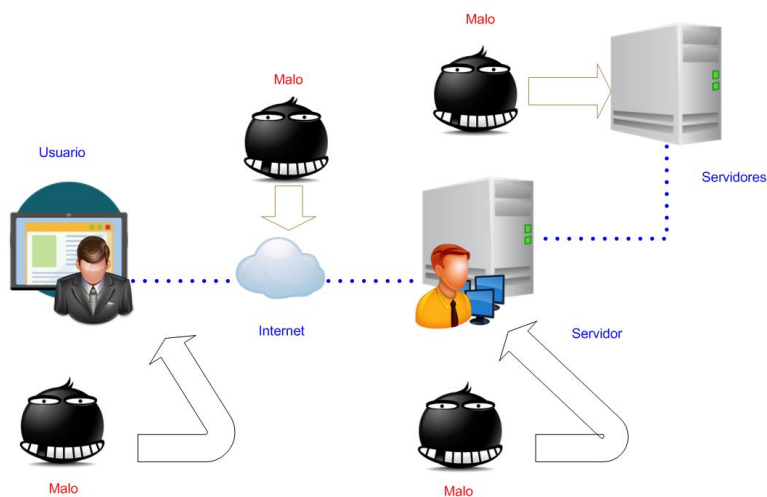
¿DÓNDE ESTÁN LOS “MALOS”?

Los “malos”, son personas individuales, o mafias que se dedican a buscar sujetos o entidades vulnerabilidades para poder obtener nuestros datos personales o financieros, con intención de lucrarse con ellos. En algunos caso puede que no exista lucro, solo existe el afán de hacer daño, de perjudicar o de exponer información de nuestra vida que nos hacen sensibles a ataques futuros, como datos sobre nuestra salud, dinero o familia.

Ahora bien, ¿Dónde están los malos?

Hay una regla sencilla para tener claro, que elementos debemos proteger, por aquí allí están los intrusos. La regla es:

¡¡¡Los malos están en todas partes!!!



- **Están del lado del Usuario:** Pueden usar técnicas de ingeniería social, tener acceso a los medios de autentificar del usuario, o atacar directamente la computadora de la víctima.



- **Están en las comunicaciones:** O bien analizándolas o falseándolas para que pensemos que estamos operando directamente con nuestros servidores o sitios web, cuando no es así.
- **Están en el servidor:** Empleados que tiene acceso a información privada o secreta, y que solo se confía en su "buena" fe para que no usen sus privilegios de forma maliciosa o bien software pernicioso que explota la información directamente de los servidores bancarios.
- **Están en los servidores adicionales:** Además de lo mencionado anteriormente, es posible que aplique solo la seguridad en los puntos de entrada de la red, y se deje toda una intranet desprotegida, confiando en que estamos protegidos de ataques del exterior. La seguridad se debe incluir en cada uno de los elementos de nuestra infraestructura.



OBJETIVOS DE LA CRIPTOGRAFÍA

Es curioso como casi todos los elementos en los que se basa en internet se crearon de una manera más o menos insegura en la que cada elemento viajaba por la red si ningún tipo de protección, libres para que cualquiera pudiera leer su contenido. Esto fue así, hasta que entro en juego la criptografía.

Muchas veces la gente no es consciente de la información personal, privada y secreta que viaja a través de sus computadoras o celular, o si lo es no le da importancia requerida. El tema es que es realmente importancia limitar el acceso a nuestra información, porque nunca sabemos que consecuencia podría tener que un extraño acceda a nuestro teléfono, nuestras fotos o mensajes, lo que para nosotros es inofensivo para otras personas abren posibilidades de extorsión o estafas en contra nuestra.

La criptografía garantiza que solo nosotros (u otras personas autorizadas) puedan acceder a nuestra información, de forma que sepamos que lo que sale de nuestros dispositivos, está debidamente protegida, y es confidencial.

La criptografía es un conjunto de técnicas y métodos en que los que un mensaje origen (o en claro) con información confidencial, se convierte en otro sin aparente significado y que puede viajar por medios no seguros, con la tranquilidad de que el mensaje original no podrá ser descubierto, sin aplicar los correspondientes métodos criptográficos sobre el mensaje original.



La criptografía existe desde hace muchos siglos, siempre asociada al envío de mensaje secretos entre dos partes. Algunos sistemas de criptografía antiguos se basan en la ocultación del método criptográfico. En los sistemas criptográficos modernos, el método de inscripción es público (o podría serlo sin comprometer la seguridad) y al mismo tiempo poseen una clave de encriptación que debe ser secreta, y que si la cual, aunque tengamos el algoritmo y el mensaje cifrado, no conseguiremos tener el mensaje original.

Las herramientas y métodos involucrados en la criptografía se usan en los siguientes aspectos:

Integridad

Garantiza que la información no ha sido alternada y es tal cual se generó. Este proceso se hace realizando una serie de operaciones matemáticas sobre esta, dando como resultado un número, casi único, al que solo se puede llegar realizando los mismos cálculos. El número resultante de estas operaciones se le conoce como “resumen” o más comúnmente como “hash”, cuanto mayor sea este número, más seguridad ofrece.

Algunos algoritmos de integridad son:

- **CRC, CRC32, MD5:** Algoritmos actualmente obsoletos que no debiera usarse actualmente.
- **Familia de algoritmos SHA:** Conjunto de algoritmos de hash, que poseen con subfamilias SHA1, SHA2, SHA3, con las siguientes características:
 - **SHA1:** Genera un numero de 160 bits, actualmente considerado no seguro:
 - **SHA256 y SHA512:** Genera un número de 256 y 512 bits respectivamente, considerados seguros actualmente.

Uno de los problemas de problemas de estos algoritmos son las colisiones esto es el hecho que se puedan genera dos hashes iguales para distinta información, lo cual permite



“adulterarla”. El otro problema es que se pueda llegar a la información original a través del hash, lo cual puede ser un fallo de seguridad importante en el caso que la información original sea una contraseña, o una clave simétrica de encriptación.

Confidencialidad

Es la posibilidad de enviar información desde el punto A al punto B garantizando que solo el emisor y el receptor puedan acceder a ella. En un ambiente inseguro desde su concepción como internet, es fundamental poder disponer de herramientas que nos permitan enviar información privada de forma segura. Nuestra información confidencial son datos biométricos y de salud, datos personales como nuestras direcciones, contactos o datos económicos como nuestras tarjetas de crédito. Generalmente se garantiza la confidencialidad mediante la encriptación, la cual se realiza a través de la una serie de transformaciones matemáticas de la información usando una llave de determinados tamaños.

Algoritmos que confidencialidad son:

- **AES:** Usando tamaños de llave de 128, 192 o 256 bits. Se usa la misma llave para encriptar y para desencriptar.
- **RSA:** Usando tamaño de llaves de 1024 (considerado obsoleto), 2048, o 4096. Se usa una llave diferente para encriptar y otra para desencriptar.

Identidad

Es lo que nos permite garantizar que la persona (o maquina) con la que estamos intercambiando información es realmente quien dice ser.

En estos casos se usa criptografía asimétrica, y principalmente certificados.

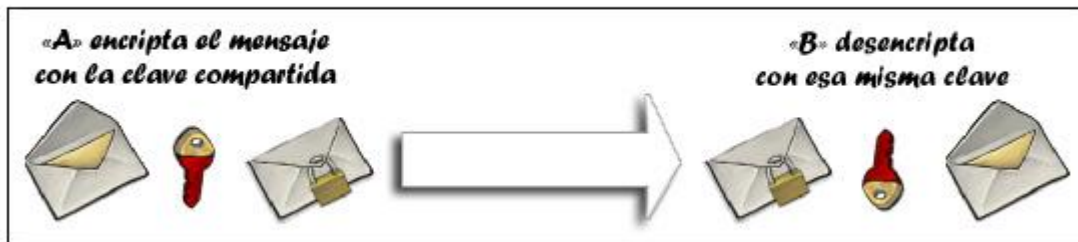
La identidad también nos ayuda a implementar el principio de “no repudiación” esto es que la persona o sistema que realiza una operación no puede negar haberla realizado.



TIPOS DE CRİPTOGRAFÍA

Criptografía simétrica

Tanto el receptor del mensaje encriptado, como la persona que encripta, usan la misma clave para realizar sus operaciones.



Algoritmos que lo usa son por ejemplo AES (Rijndael).

Las **ventajas** de este método son:

- Es la practica tiene una velocidad aceptable
- Es muy seguro

Las **desventajas**:

- La principal desventaja es que un usuario genera la clave y tiene que pasársela a otro usuario, lo cual podría comprometer la conexión desde un inicio

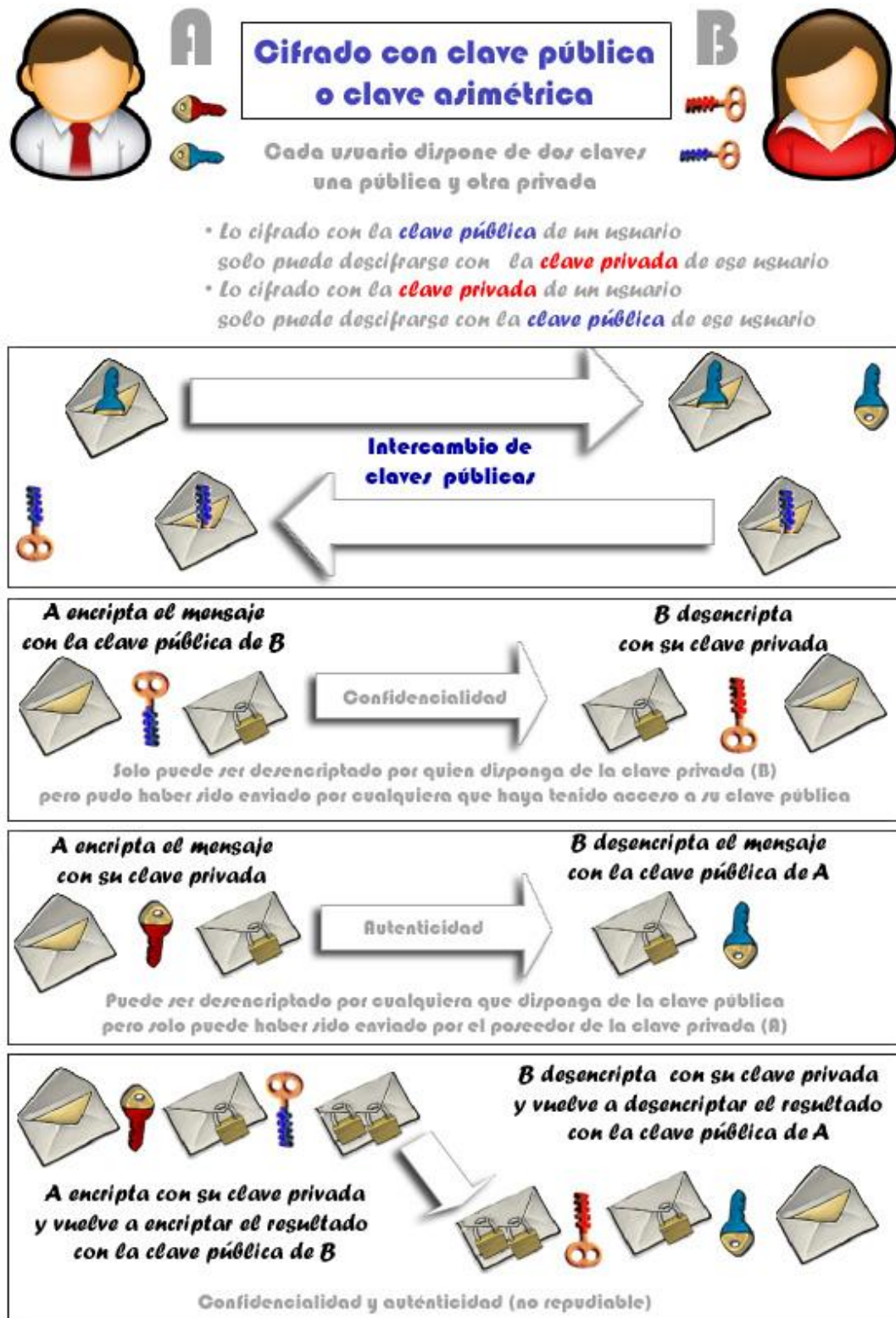


El hecho que compartan la misma clave es un problema, el único que tiene este tipo de criptografía, porque ¿Cómo se hace que el usuario final y destino tengan la misma clave?, solo enviando la clave por un medio que sea seguro, para garantizar que nadie más la tiene, pero precisamente necesito intercambiar la llave porque no tengo un medio seguro para enviar información.

Criptografía asimétrica

Consta de dos claves, en lugar de una, llamadas clave privada y clave pública, la clave pública sirve para encriptar, y es de libre acceso, la clave privada para desencriptar y solo la tiene una persona (la que genero las claves). Además, la privada sirve para firmar mensajes y garantizar que la persona que la usa es el dueño de la clave.

En la práctica funciona así: Una persona genera (a través de un algoritmo) una clave privada y pública (que se complementan entre sí), se queda para sí la clave privada, y distribuye la clave pública (a quien desee), a partir de ese momento, quien desee mandarle un mensaje encriptado, puede hacerlo usando la clave pública, y solo el podrá desencriptarlo usando la clave privada. Igualmente, si él quiere mandar un comunicado y garantizar su autenticidad, podrá firmarlo con su llave privada, y todo aquel que tenga la llave publica, podrá validar su origen.





Algoritmos que implementan esta tecnología son por ejemplo RSA o DSA.

Las **ventajas** de estos algoritmos son:

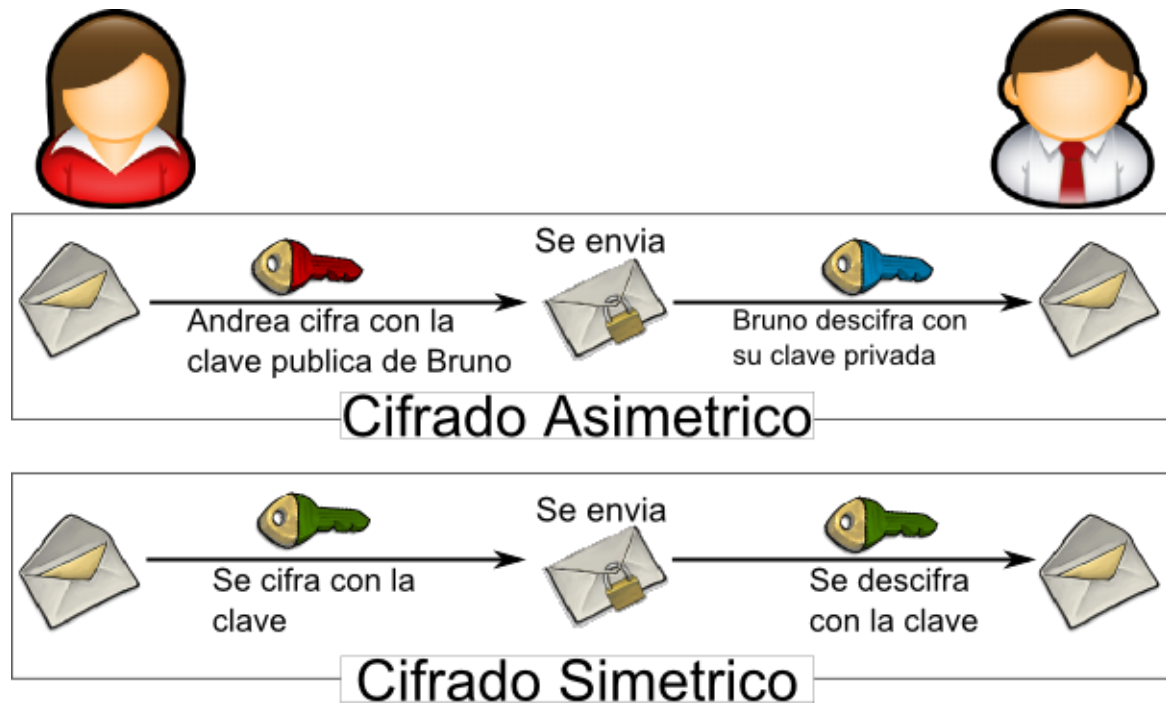
- Seguridad en cuanto al custodio de las claves (nunca se difunde la clave privada)
- Seguro en cuanto los mecanismos de inscripción

Las **desventajas**:

- Increíblemente lento

En esta criptografía se resuelve el problema de la llave compartida, puesto que, al haber dos llaves, se puede distribuir libremente la llave pública (a cualquier persona) para encriptar, y la permanecer debidamente custodiada la llave privada para desencriptar. Así por ejemplo dos personas (o entes), pueden iniciar una comunicación segura, simplemente intercambiando sus llaves públicas y usando sus llaves privadas para desencriptar.

Debido a que la criptografía asimétrica es realmente lenta, en lugar de encriptar un mensaje completo, se genera una llave simétrica aleatoria, con la que se encripta el mensaje y se envía el mensaje encriptado y adjunto con la llave simétrica encriptada (ahora sí) con la llave asimétrica pública, lo cual garantiza la velocidad y la confidencialidad.





CERTIFICADOS DIGITAL

Cuando recibimos un mensaje firmado con una llave privada, podemos validar su autenticidad con nuestra llave pública. Esto quiere decir que podemos garantizar que la persona que tiene la llave privada es la se está comunicado, con nosotros, puesto que debe coincidir con la llave publica que nos proporcionó.

Hay que agregar que la firma es diferente por cada mensaje, por lo que garantiza no solo que el mensaje fue enviado por quien tiene la llave privada, sino además que el mensaje no ha sido modificado en el cambio.

Un detalle importante es que solo garantiza eso, es decir que el mensaje fue firmado con la llave privada, pero no garantiza que esa firma electrónica pertenezca a una persona en particular. La firma podría ser de cualquier persona, y haberse generado en cualquier momento.

Para garantizar además que la firma privada pertenece a una determinada persona o entidad, se usan los certificados, que bien pueden identificar a una persona o a un ente como una empresa o un banco.

Pero ¿En qué consiste exactamente un certificado? Imaginemos que yo quiero comunicarme (a través de una computadora) con otra persona o entidad, por ejemplo, una tienda. Con el uso de llaves públicas (y privada) garantizo que la comunicación es confidencial entre ambos, pero de ninguna forma garantizo que esa llave pertenezca a la tienda, pero igualmente imaginemos que existe un tercer elemento, un ente en el que



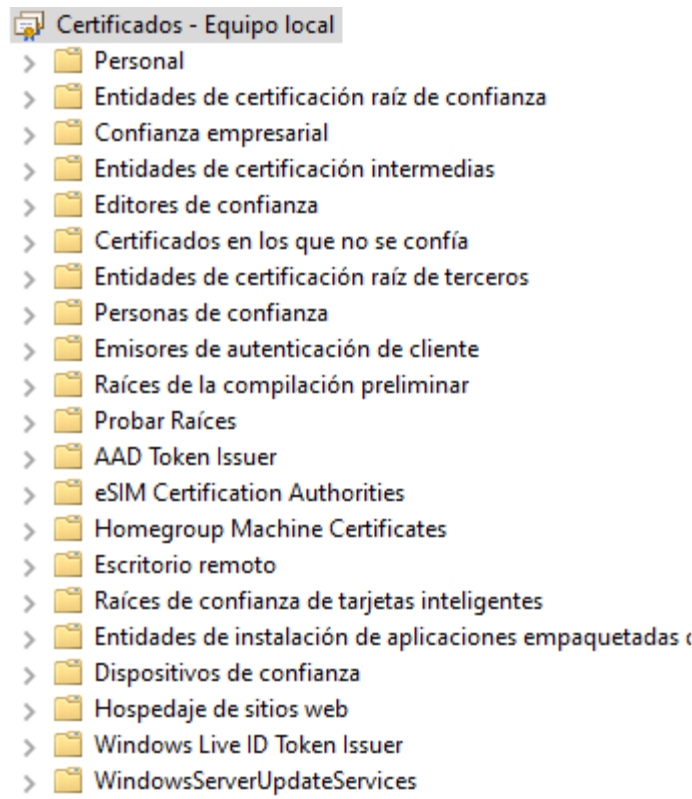
confiamos los dos (la tienda y el usuario), que garantice que esa llave privada pertenece a la tienda, ¿Cómo lo puede hacer? Dando por supuesto que yo confié en el tercero, que se llama entidad certificadora, esta firma (con su propia llave privada), la llave pública y demás información de la tienda, posterior a que valide la identidad de la tienda (mediante documentos, escrituras, registros de empresa, visitas en persona). En este punto ya tenemos un certificado, que contiene la llave pública, y que está firmado por alguien en el que confiamos que ha validado correctamente la identidad de la tienda, de esta forma queda ligada la identidad con la llave pública.

¿Quién son esas entidades certificadoras y por qué confiamos en ella? Son grandes empresas cuyo servicio es precisamente ese garantizar la identidad de entes y personas, y son aceptadas a nivel mundial como VERISIGN, o incluso gobiernos o entidades gubernamentales que realizan dicha función.



ELEMENTOS INVOLUCRADOS EN UN CERTIFICADO

- **Cliente:** Es el ente (maquina, sistema o persona), que desea realizar una operación segura
- **Servidor:** Es el ente (maquina, sistema o empresa), que está involucrada como destino de una operación.
- **Entidad Certificadora:** Es una entidad (organismo o empresa) conocida y respecta con la capacidad de poder validar y garantizar la identidad de un servidor (o de una persona dado el caso).
- **Entidad Intermedia:** Normalmente la entidad Certificadora no se usa para generar certificados directamente, ya que podría ser catastrófico si fuera comprometida. Para evitar esto las Entidades Certificadoras generan “Entidades Intermedias” que son las que se usan en la práctica para generar los certificados clientes.
- **Llave privada y pública:** Son el conjunto de claves a usar, la privada pertenece al cliente, la pública es usada por la entidad certificadora.
- **Petición de certificado (archivo .csr):** Es el conjunto de información del cliente, que además está firmada por su clave pública, la entidad certificadora debe validar y garantizar su autenticidad.
- **Certificado público (archivo .cer):** Es el certificado generado por la entidad y firmado por este, que garantiza la identidad del cliente.
- **Certificado privado (archivo .pfx):** Es la unión del certificado público y la llave privada, para ser usado para fines de firmado y autenticación.
- **Almacén de certificados:** Es un lugar seguro donde se guardan los certificados, generalmente un archivo o base de datos protegido por una contraseña o por un parte del sistema operativo, por ejemplo:
 - **Alcancen de Windows:**

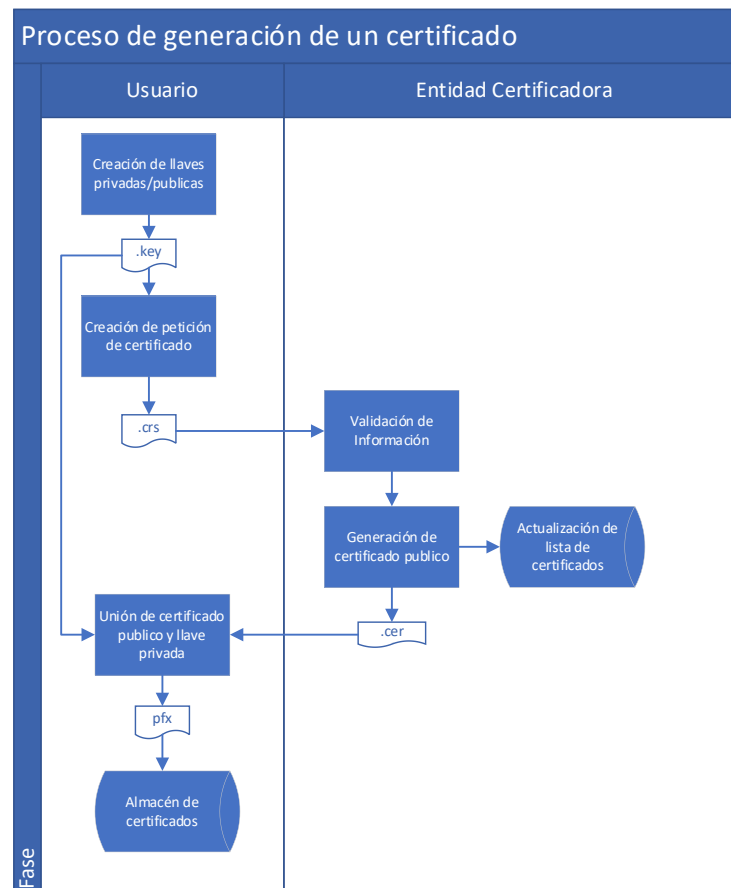


- **Java KeyStore (archivos JKS):** es un repositorio tanto de certificados públicos como privados, generalmente usados en java.
- **Lista de revocación (archivo .crl):** Es un archivo firmado por la entidad certificadora que almacena los certificados que ya no son válidos, generalmente porque su seguridad ha sido comprimida, y ya no es seguro su uso.
- **Online Certificate Status Protocol (OSCP):** Puede considerarse una evolución de los CRL, sirve para consultar la validez de un certificado en línea, parecido a realizar la consulta en una base de datos, con lo cual es más ágil y rápido.



PROCESO PARA GENERAR UN CERTIFICADO

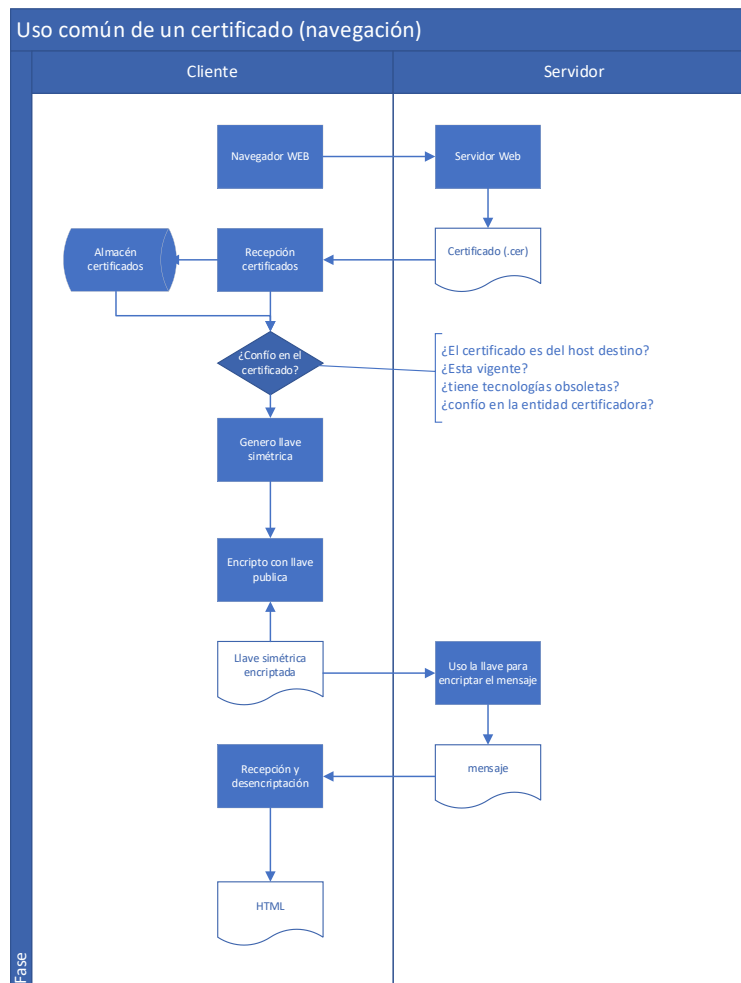
1. Se genera un par de llaves (pública y privada)
2. Se realiza una petición de generación de certificado en base a la clave pública, y a los datos de la empresa (y los hosts que queremos certificar).
3. Se envía la petición a una entidad certificadora, que realizara los pasos necesarios para garantizar que somos la empresa para la cual estamos pidiendo los certificados, y una vez garantizado nos expedida dicho certificado.
4. En nuestro servidor podemos unir la llave privada con el certificado, y en base a eso configurar nuestros servidores para que lo usen apropiadamente





USO DE UN CERTIFICADO

Nuestros clientes al conectarse a nuestros servidores descargarán el certificado (con la clave pública), y al estar firmado por la entidad certificadora (en la confiamos), asumiremos que el servidor al que nos estamos conectando es el correcto. Igualmente, si no está firmado o lo está por alguien en que no confiamos seremos advertidos de esta situación, y estará en nuestra decisión continuar la comunicación con dicho servidor.

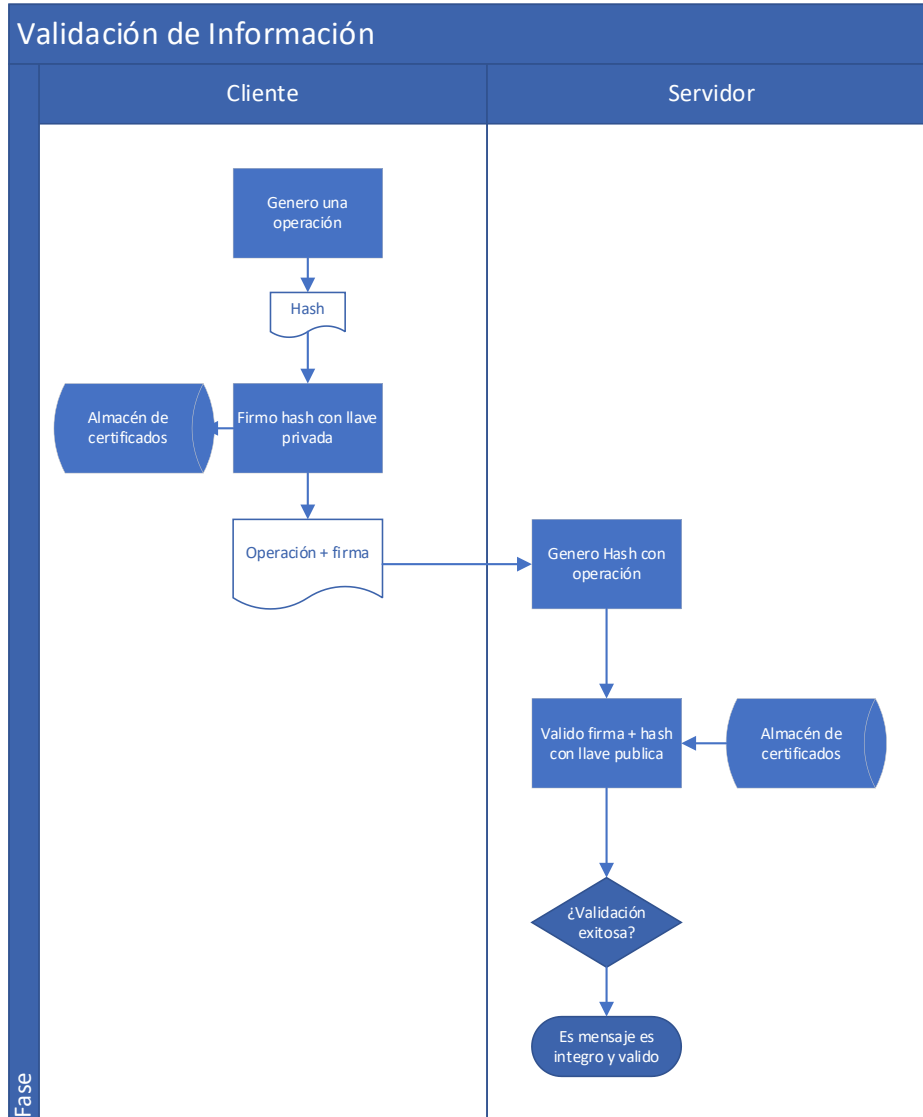




AUTENTIFICACIÓN DE OPERACIONES

Para poder garantizar que las operaciones son realizadas por un usuario mediante un certificado existen diversos patrones, pero todas ellas involucran firmas digitales, y elementos que solo posee el cliente.

1. El cliente genera una operación
2. En el cliente se genera un hash representativo de la operación.
3. El hash se firma digitalmente con una llave privada.
4. Se envía la operación y la firma (el hash no firmado nunca se envía).
5. En el servidor se toma la operación y se vuelve a genera el hash, notase que el hash nunca se envió. Se debe llegar al mismo hash desde el cliente y desde el servidor, con la información disponible, es decir la misma operación.
6. Se valida el hash obtenido con la firma, si coincide la operación esta integra y además la ha generado el cliente (o por lo menos el que tenga la llave privada).

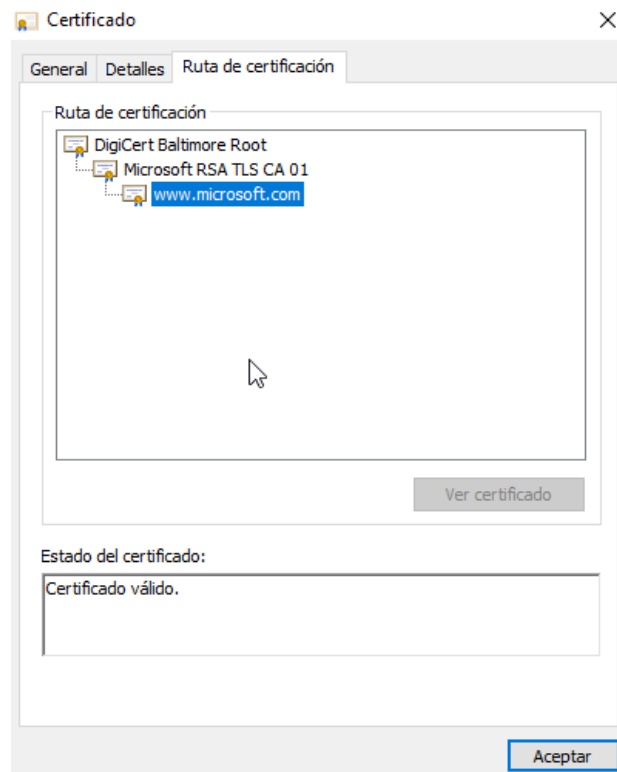




CARACTERÍSTICAS DE UN CERTIFICADO

El formato más popular de certificado público es el X.509, que nos permite agrupar una serie de características y valores que han sido verificados y aprobados por una entidad certificadora.

Los certificados se organizan de forma jerárquica, conteniendo una cadena de certificación. El certificado es creado por un certificado intermedia y el certificado intermedio por un certificado raíz (que representa una entidad certificadora).





Los certificados raíces “confían” en sí mismos. No necesita de una entidad superior para ser confiable, generalmente están instalados por defecto en los navegadores y los sistemas operativos (en los que confiamos), si instalamos navegadores o sistemas operativos de dudosa procedencia, pueden instalarse certificados raíz apócrifos y podemos ser víctimas de un ataque por parte de un intruso.

Elementos de un certificado

- **Fecha de validez:** indica la fecha máxima en la que el certificado es válido.
- **Subject:**

Es la persona, empresa o servicio para el cual se creó el certificado, tiene una estructura en forma de cadena que agrupa elementos como su nombre, su estado, su país, o su correo electrónico, de la siguiente forma:

- **CN:** Nombre del host o persona sobre la que se está expidiendo el certificado.
- **OU:** Unidad Organizativa, en una empresa suele ser el departamento al cual pertenece el certificado
- **O:** Organización, o empresa.
- **L:** Ciudad o municipio
- **S:** Estado
- **C:** País

Por ejemplo:

CN=José Luis Bautista Martín; OU=Desde Las Horas Extras; O=Capicua; L=Guadalajara; S=Jalisco; C=México

- **CLR y OCSP:** Representan consultas de listas de revocación (para comprobar si el certificado sigue siendo válido).
- **KeyUsage:** Es una lista de opciones que indica de forma general para que se va a usar la clave pública del certificado, los posibles valores son:



Tipo	Codigo Hexadecimal	Codigo Decimal	Descripción
digitalSignature	80	128	
nonRepudiation	40	64	
keyEncipherment	20	32	
dataEncipherment	10	16	
keyAgreement	8	8	

- **extendedKeyUsage:** Indica de forma más precisa para que se va a usar el certificado:

Tipo	Codigo OID	Descripción
serverAuth	1.3.6.1.5.5.7.3.1	SSL/TLS WWW Server Authentication
clientAuth	1.3.6.1.5.5.7.3.2	SSL/TLS WWW Client Authentication
codeSigning	1.3.6.1.5.5.7.3.3	Code Signing
emailProtection	1.3.6.1.5.5.7.3.4	E-mail Protection (S/MIME)
timeStamping	1.3.6.1.5.5.7.3.8	Trusted Timestamping
OCSPSigning	1.3.6.1.5.5.7.3.9	OCSP Signing
ipsecIKE	1.3.6.1.5.5.7.3.5	ipsec Internet Key Exchange

Algunos ejemplos de uso de certificados son:

- **Entidad Certificadora**

`keyUsage = critical, digitalSignature, cRLSign, keyCertSign`

- **Entidad Intermedia**

`keyUsage = critical, digitalSignature, cRLSign, keyCertSign`

- **Certificado de usuario:**

`keyUsage = critical, nonRepudiation, digitalSignature, keyEncipherment`
`extendedKeyUsage = clientAuth, emailProtection`

- **Certificado de servidor**

`keyUsage = critical, digitalSignature, keyEncipherment`
`extendedKeyUsage = serverAuth`



- **Certificado OCSP**

```
keyUsage = critical, digitalSignature  
extendedKeyUsage = critical, OCSPSigning
```

- **Algoritmo de firma:** es el algoritmo que va a usar el dueño del certificado para firmar, puede ser alguna versión de SHA junto RSA por ejemplo SHA1RSA
- **Clave pública:** Es la clave pública del certificado con la podremos encriptar la información y proporcionársela al dueño del certificado.
- **Nombres alternativos:** En el caso de un servidor, indica todos los nombres por los cuales responde el servidor, este es un campo obligatorio.
- **Microsoft Cryptographic Service Providers:** Los Cryptographic Service Providers (CSP) son un conjunto de API de Microsoft con una funcionalidad en concreto, los certificados en Windows se configuran para poder usar o varios CSP.



Provider Name & Type	Description	Purposes	Crypto	Default Microsoft Templates
Microsoft Software Key Storage Provider (CNG)	Standard windows software based RSA and ECC provider.	Key Exchange Digital Signature Data Encryption	<i>RSA</i> <i>ECC SHA1</i> <i>SHA2</i>	<i>OCSP Response Signing (KSP Required, Provider not specific)</i>
Microsoft Smart Card Key Storage Provider (CNG)	Supports smart card key creation and use	Key Exchange Digital Signature Data Encryption	RSA ECC SHA1 SHA2	None
Microsoft Platform Crypto Provider (CNG)	Supports smart card key creation and use	Key Exchange Digital Signature Data Encryption	RSA ECC SHA1 SHA2	None
Microsoft Platform Crypto Provider (CNG)	Generates and stores keys in Trusted Platform Modules. Supports Key Attestation to allow CA to ensure key is created in TPM/Virtual smart card	Key Exchange Digital Signature Data Encryption	RSA ECC SHA1 SHA2	None
Microsoft RSA SChannel Cryptographic Provider (CAPI)	Supports hashing, data signing, and signature verification. The algorithm identifier CALG_SSL3_SHAMD5 is used for SSL 3.0 and TLS 1.0 client authentication. This CSP supports key derivation for the SSL2, PCT1, SSL3 and TLS1 protocols.	Key Exchange	RSA SHA1	CEP Encryption Computer Directory Email Replication Domain Controller Domain Controller Authentication IPSec IPSec (Offline) Kerberos Authentication RAS and IAS Server Router (Offline request) Web Server Workstation Authentication



Microsoft Enhanced DSS and Diffie-Hellman Cryptographic Provider (CAPI)	Supports Diffie-Hellman key exchange (a 40-bit DES derivative), SHA hashing, DSS data signing, and DSS signature verification. Derived from Base DSS and Diffie-Hellman Cryptographic Provider. Adds support for RC2/4, DES and 3DES encryption	Digital Signature	RSA SHA1	Authenticated Session Basic EFS CA Exchange Code Signing EFS Recovery Agent Enrollment Agent Enrollment Agent (Computer) Exchange Enrollment Agent (Offline request) Exchange Signature Only Exchange User Key Recovery Agent Trust List Signing User User Signature Only
Microsoft DSS and Diffie-Hellman/Schannel Cryptographic Provider (CAPI)	Supports hashing, data signing with DSS, generating Diffie-Hellman (D-H) keys, exchanging D-H keys, and exporting a D-H key. This CSP supports key derivation for the SSL3 and TLS1 protocols. This CSP supports key derivation for the SSL3 and TLS1 protocols.	Key Exchange	RSA SHA1	Web Server
Microsoft Base Cryptographic Provider (CAPI)	A broad set of basic cryptographic functionality that can be exported to other countries or regions. No 3DES support. RC2/4 limited to 40bits.	Digital Signatures Data Encryption	RSA SHA1	Administrator Authenticated Session Basic EFS Code Signing EFS Recovery Agent Enrollment Agent Enrollment Agent (Computer) Exchange Enrollment Agent (Offline request) Exchange Signature Only Exchange User Trust List Signing User User Signature Only



Microsoft DSS Cryptographic Provider (CAPI)	Provides hashing, data signing, and signature verification capability using the Secure Hash Algorithm (SHA) and Digital Signature Standard (DSS) algorithms.	Digital Signatures	RSA SHA1	Authenticated Session Code Signing Enrollment Agent Enrollment Agent (Computer) Exchange Enrollment Agent (Offline request) Exchange Signature Only Trust List Signing User Signature Only
--	--	--------------------	----------	--

Información de CSP extraída de <https://www.pkisolutions.com/understanding-microsoft-crypto-providers/>



GENERACION DE CA CON OPENSLL

Un certificado digital es un documento electrónico, expedido por una entidad competente para tal efecto, que básicamente garantiza la identidad de una persona o un servidor, ligándolo a una clave pública.

Se basa en la confianza que se tenga en la entidad certificadora, si confiamos en la entidad, también confiamos en la identidad de los certificados que expida.

Se realiza un ejemplo completo de creación de certificados, se toma como manual, el instructivo de la siguiente página:

<https://jamielinux.com/docs/openssl-certificate-authority/>

Se incluye una carpeta de ejemplo llama "Certificadora Inicial" y "Certificadora Final", **los certificados aquí incluidos son para efectos de capacitación y nunca deben usarse productivamente.**

La descripción de todos los certificados a crear se incluye en **el anexo uno.**

Entidades certificadoras

Las entidades certificadoras son las encargadas de asociar una llave pública, con una identidad, además de garantizar mediante su confiabilidad dicha identidad.

Los certificados de las entidades certificadores están firmados a sí mismos, se llaman certificados autofirmados.



1. Genero las llaves privadas

```
openssl genrsa -aes256 -out private/ca.key 4096
```

2. Genero el certificado autofirmado

```
openssl req -config openssl.cnf -key private/ca.key -new -x509 -days 7300 -sha256 -  
extensions v3_ca -out certs/ca.cer
```

3. Verifico el certificado

```
openssl x509 -noout -text -in certs/ca.cer
```

Entidades intermediarias

Las entidades intermediarias son certificadoras en las que la certificadora raíz confía para realizar su trabajo.

1. Creo las llaves

```
openssl genrsa -aes256 -out intermediate/private/intermediate.key 4096
```

2. Creo la petición de generación del certificado

```
openssl req -config intermediate/openssl.cnf -new -sha256 -key  
intermediate/private/intermediate.key -out intermediate/csr/intermediate.csr
```

3. Firmo el certificado

```
openssl ca -config openssl.cnf -extensions v3_intermediate_ca -days 3650 -notext -  
md sha256 -in intermediate/csr/intermediate.csr -out  
intermediate/certs/intermediate.cer
```

4. Verifico el resultado

```
openssl x509 -noout -text -in intermediate/certs/intermediate.cer  
openssl verify -CAfile certs/ca.cer intermediate/certs/intermediate.cer
```




5. Creo la cadena de certificación

```
cat intermediate/certs/intermediate.cer certs/ca.cer > intermediate/certs/ca-chain.cer
```

Certificados personales

Son los que encargan de identificar de una maquina o una persona.

1. Establezco el nombre del host

```
set hostcrt=ejemplohost
```

2. Copio y edito la plantilla

```
copy intermediate\template.cnf intermediate\csr\%hostcrt%.cnf
```

En el archivo csr\ejemplohost.cnf, tengo que agregar una configuración alternativa para el nombre del DSN, con poner el mismo nombre que el host, es suficiente

```
[ alt_names ]
DNS.1 = ejemplohost
```

3. Genero las llaves para el servidor

```
openssl genrsa -aes256 -out intermediate/private/%hostcrt%.key 2048
```

4. Creo la petición del certificado

```
openssl req -config intermediate/csr/%hostcrt%.cnf -key intermediate/private/%hostcrt%.key -new -sha256 -out intermediate/csr/%hostcrt%.csr
```

5. Firmo el certificado



```
openssl ca -config intermediate/csr/%hostcrt%.cnf -extensions server_cert -days 375  
-notext -md sha256 -in intermediate/csr/%hostcrt%.csr -out  
intermediate/certs/%hostcrt%.cer
```

6. Verifico que se hayan creado adecuadamente

```
openssl x509 -noout -text -in intermediate/certs/%hostcrt%.cer  
openssl verify -CAfile intermediate/certs/ca-chain.cer  
intermediate/certs/%hostcrt%.cer
```

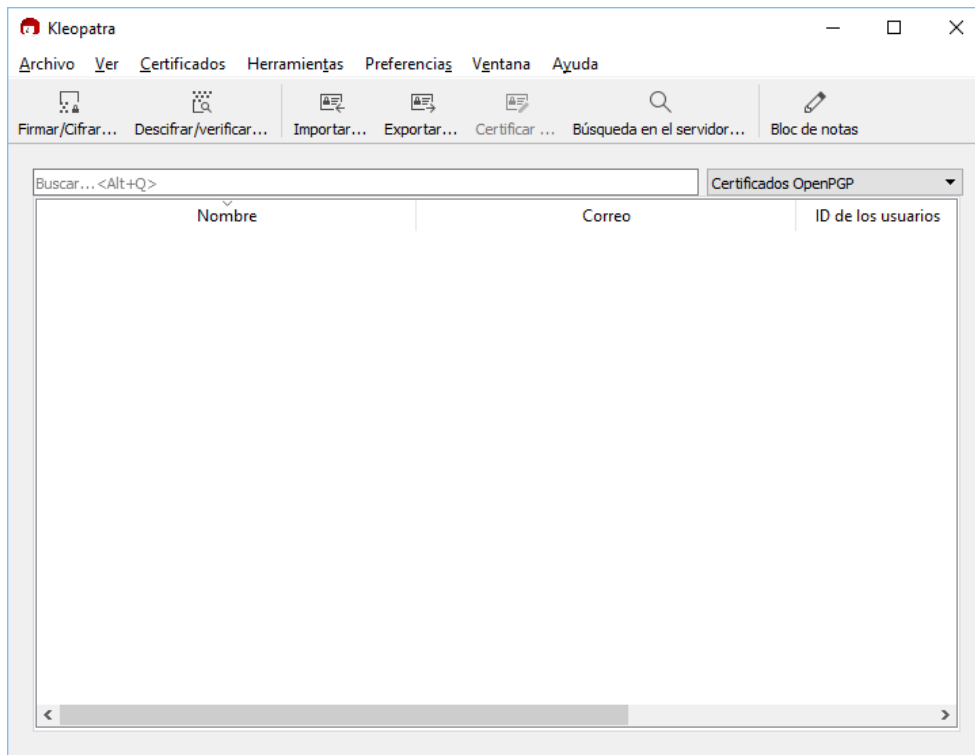
7. Creo el certificado que incluya la llave privada

```
openssl pkcs12 -export -out intermediate/private/%hostcrt%.pfx -inkey  
intermediate/private/%hostcrt%.key -in intermediate\certs\%hostcrt%.cer
```

Configuración de certificados con GnuPG

GnuPG, es una herramienta de software libre para la encriptación y firmado de documentos, tiene varias interfaces gráficas, y se integran directamente con algunos gestores de correo electrónico.

Se basa en la combinación de criptográfica asimétrica y simétrica para cifrar y firmar los mensajes.



Puede manejar sus propios certificados o certificados más estándares como X509

Generar un certificado con el asistente



? X

← Asistente de creación de par de claves

Elegir formato

Por favor, elija que tipo quiere usted crear.

→ Crear un par de claves personales OpenPGP
Los pares de claves OpenPGP están certificados por la confirmación de la huella digital de la clave pública.

→ Crear un par de claves personales X.509 y una solicitud de certificación
Los pares de claves X.509 se certifican por una autoridad de certificación (CA). La petición generada necesita enviarse a la cA para finalizar la creación.

Next Cancel

? X

← Asistente de creación de par de claves

Introduzca detalles

Por favor, introduzca sus detalles personales debajo. Si desea más control sobre los parámetros, pulse el botón «Configuración avanzada».

Nombre de pila (CN): (requerido)

Dirección de correo (EMAIL): (requerido)

Ubicación (L): (opcional)

Unidad organizativa (OU): (opcional)

Organización (O): (requerido)

Código de país (C): (requerido)

CN=Certificado Personal gpg 001,O=Capacitacion Criptografia Certificados Privacidad,C=MX

☐ Añadir dirección de correo al DN (solo para CA defectuosas)

Configuración avanzada...

Next Cancel

Firmado de las solicitudes de certificados con OpenSSL

Los pasos para firmar una solicitud de certificado con OpenSSL son:



1. Establezco el nombre del host

```
set hostcrt=certificadogpg001
```

2. Convierto el p10 a csr

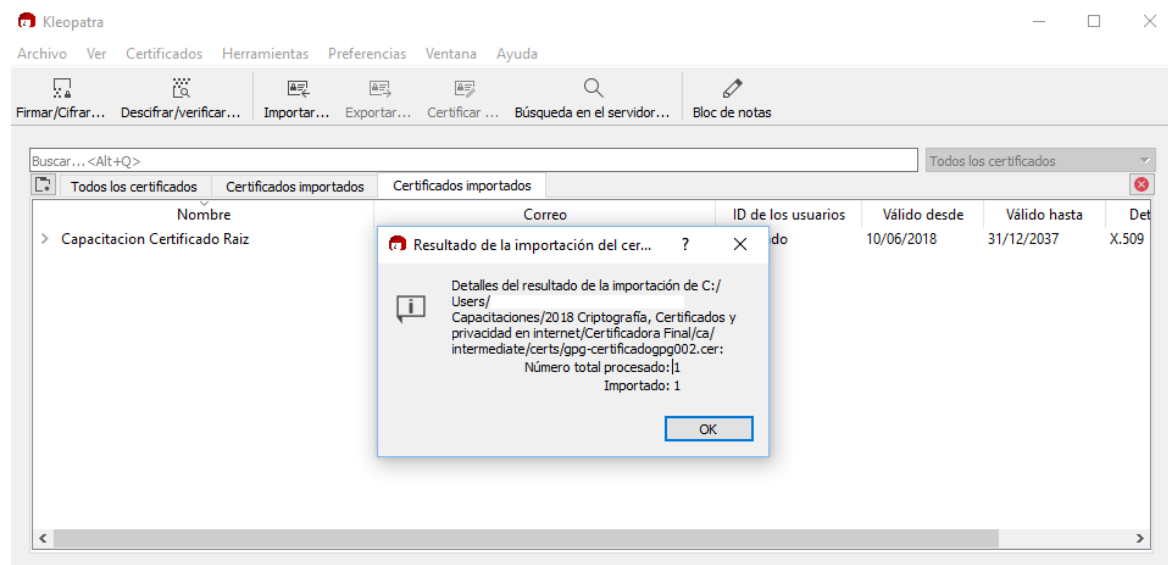
```
openssl req -in intermediate\csr\%hostcrt%.p10 -inform der -out  
intermediate\csr\%hostcrt%.csr
```

3. Firmo el certificado

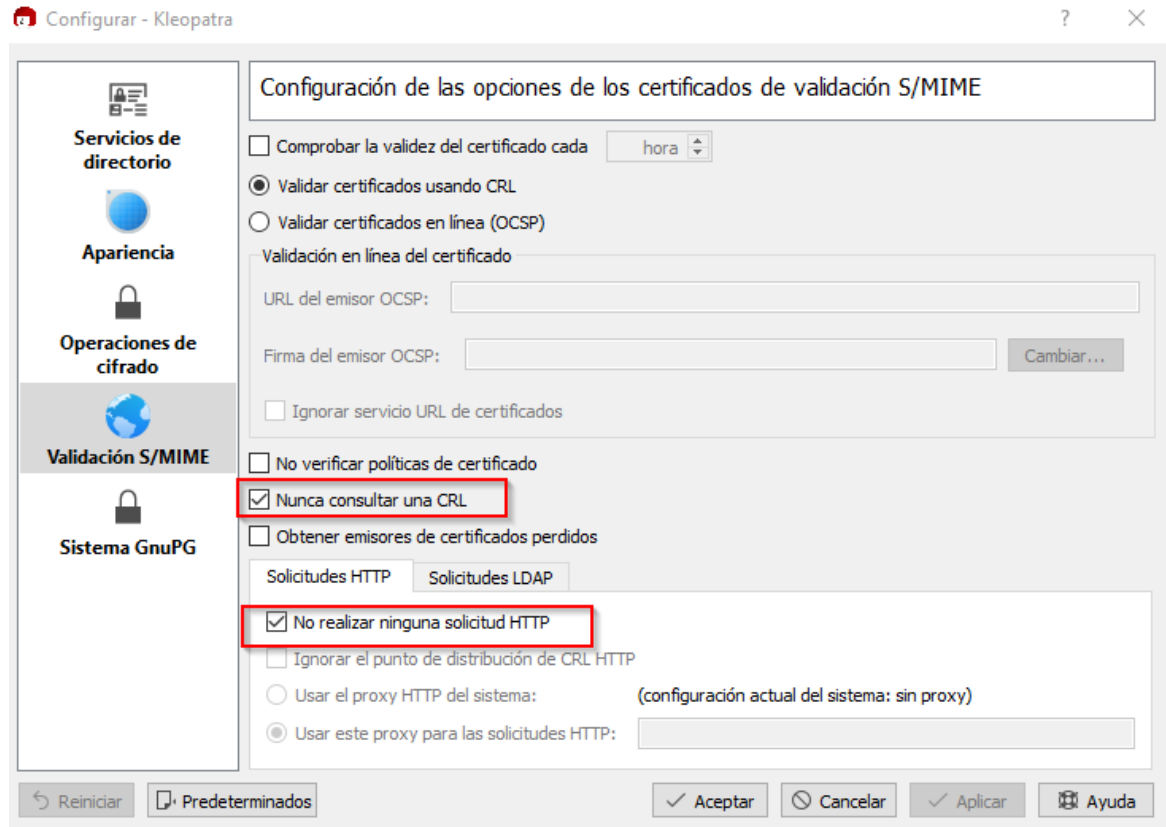
```
openssl ca -config intermediate/openssl.cnf -extensions usr_cert -days 375 -notext  
-md sha256 -in intermediate/csr/%hostcrt%.csr -out intermediate/certs/gpg-  
%hostcrt%.cer
```

Importación de los certificados

Se importan el certificado en Kleopatra




Exclusivamente para este ejemplo vamos a deshabilitar la lista de revocación de certificados:



Uso de los certificados: Encriptación y firma



 Firmar/cifrar archivos - Kleopatra?✕

Firmar/cifrar archivos

Probar autenticidad (firmar)

☒ Firmar como: Certificado Personal gpg 001 (certificado, S/MIME, creado: 10/06/2018) ▼

Cifrar

☐ Cifrar para mí: Certificado Personal gpg 002 (certificado, S/MIME, creado: 10/06/2018) ▼


☒ Cifrar para otros: Certificado Personal gpg 002 (certificado, X.509, creado: 10/06/2018) ✕

? Por favor, introduzca un nombre o dirección de correo...

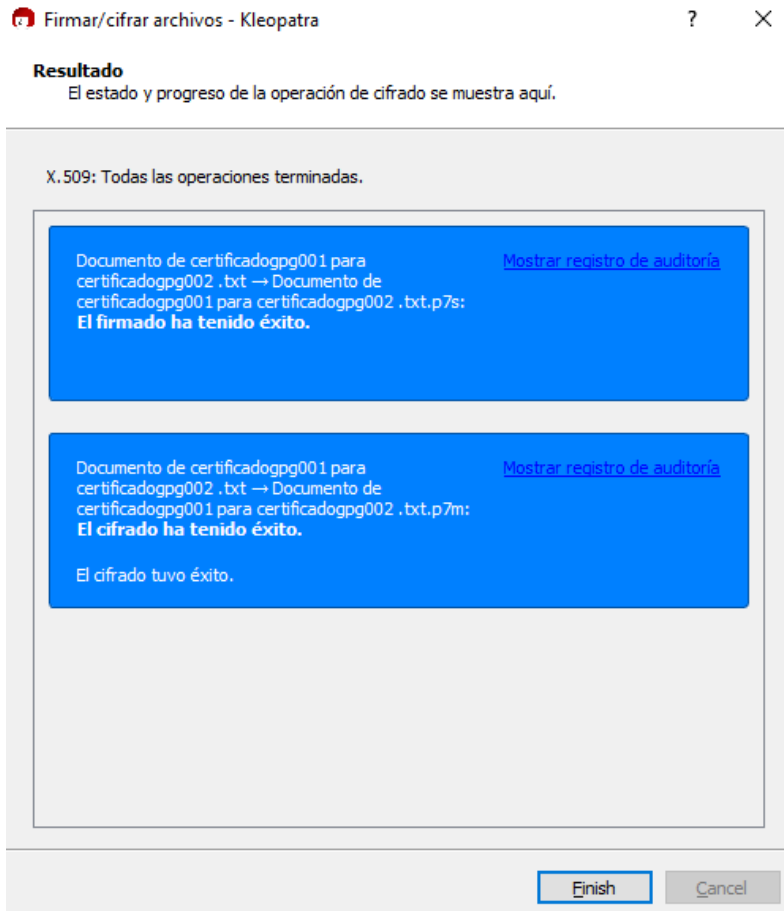
☐ Cifrar con contraseña. Cualquiera con el que usted comparta la contraseña podrá ver los datos.

Salida

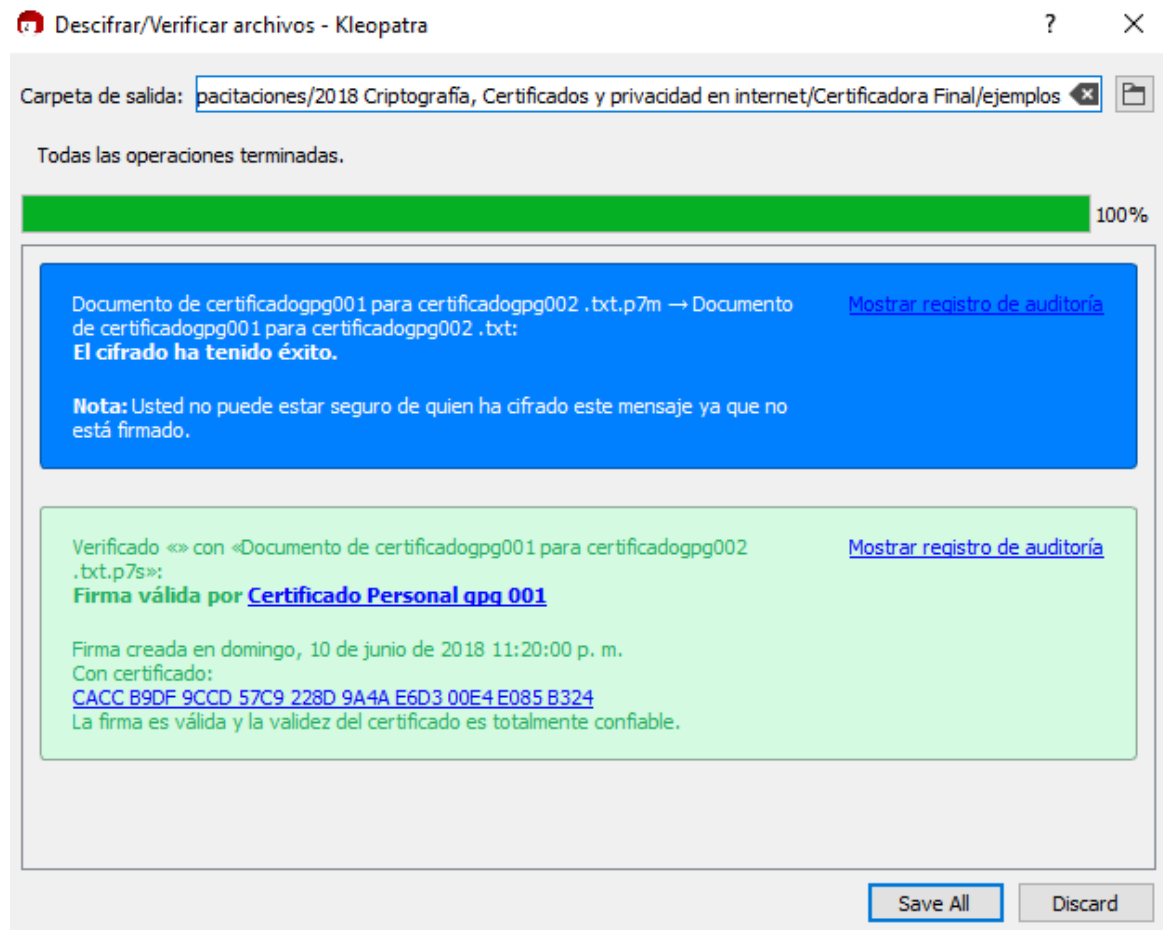
☒ Cifrar / Firmar cada archivo por separado.

2018 Criptografía, Certificados y privacidad en internet/Certificadora Final/ejemplos ✕ 

Firmar/cifrar Cancel



Uso de los certificados: Descriptación y validación



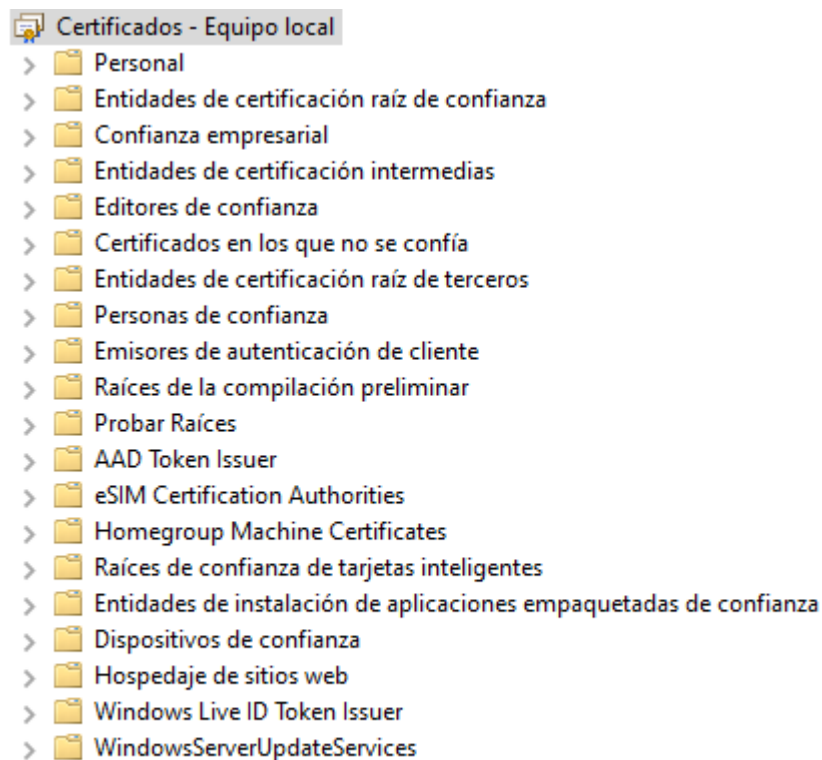


ALMACÉN DE CERTIFICADOS

Los almacenes son lugares seguros, generalmente protegidos con una contraseña, o una cuenta de usuario, que almacenan certificados, dejándolo disponibles para su uso cuando sea conveniente.

Almacén de certificado de Windows

Es un almacén de certificados que se ligan a permisos de usuarios. El almacén es a nivel individual para cada usuario, o a nivel de máquina para todos los usuarios.



El almacén se clasifica en subalmacenes, los más importantes son:

- **Entidades de Certificación Raíz de Confianza:** Almacenen de certificados autoafirmados de los cuales derivan todos los demás.



- **Entidades de Certificación Intermedias:** Certificados intermedios con los que se firman el resto de los certificados.
- **Personal:** Certificados del usuario generalmente no es necesario tenerlos instalados (a no ser que de deba usar la llave privada), ya que el confiar en el certificado raíz y la entidad intermedia, se confía automáticamente en los certificados firmados con estos.

Java Keystore

Java Keystore (JKS) es un almacén de certificados usado principalmente por los sistemas creados en Java. La herramienta principal para trabajar con este almacén es keytool.

- **Generar un almacén (con un certificado):**

```
keytool -genkey -keyalg RSA -alias ca -keystore almacen.jks
...
Enter keystore password:
Re-enter new password:
What is your first and last name?
  [Unknown]: Jose Luis Bautista Martin
What is the name of your organizational unit?
  [Unknown]: Desde las Horas Extras
What is the name of your organization?
  [Unknown]: Capicua
What is the name of your City or Locality?
  [Unknown]: Guadalajara
What is the name of your State or Province?
  [Unknown]: Jalisco
What is the two-letter country code for this unit?
  [Unknown]: MX
Is CN=Jose Luis Bautista Martin, OU=Desde las Horas Extras, O=Capicua, L=Guadalajara, ST=Jalisco, C=MX correct?
  [no]: yes

Generating 2,048 bit RSA key pair and self-signed certificate (SHA256withRSA) with a validity of
90 days
    for: CN=Jose Luis Bautista Martin, OU=Desde las Horas Extras, O=Capicua, L=Guadalajara,
ST=Jalisco, C=MX
```

- **Listas los certificados de un almacén:**

```
keytool -list -v -keystore almacen.jks
```



```
...
Enter keystore password:
Keystore type: PKCS12
Keystore provider: SUN

Your keystore contains 1 entry

Alias name: ca
Creation date: 16 abr. 2022
Entry type: PrivateKeyEntry
Certificate chain length: 1
Certificate[1]:
Owner: CN=Jose Luis Bautista Martin, OU=Desde las Horas Extras, O=Capicua, L=Guadalajara,
ST=Jalisco, C=MX
Issuer: CN=Jose Luis Bautista Martin, OU=Desde las Horas Extras, O=Capicua, L=Guadalajara,
ST=Jalisco, C=MX
Serial number: da8ea20304d35bef
Valid from: Sat Apr 16 21:51:36 CDT 2022 until: Fri Jul 15 21:51:36 CDT 2022
Certificate fingerprints:
    SHA1: 7F:E2:59:C3:E9:B7:E2:D7:13:96:57:EE:14:CC:2E:15:B8:0F:0A:9F
    SHA256:
D1:C1:37:D7:3C:D4:06:68:B3:A3:E7:4A:12:94:F4:79:35:27:38:41:5F:16:D3:AB:26:6D:00:CD:B2:45:EA:08
Signature algorithm name: SHA256withRSA
Subject Public Key Algorithm: 2048-bit RSA key
Version: 3

Extensions:

#1: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: 87 BF 83 59 FC 28 47 22  44 2F AA E5 DC E3 72 BD  ...Y.(G"D/....r.
0010: 22 55 73 5B              "Us[
]
]
```

- **Exportar certificado:**

```
keytool -exportcert -alias ca -file ca.cer -keystore almacen.jks
```

```
...
Enter keystore password:
Certificate stored in file <ca.cer>
```

- **Exportar con llave privada:**

```
keytool -importkeystore -srckeystore almacen.jks -destkeystore ca.pfx -deststoretype PKCS12
```

```
...
Importing keystore almacen.jks to ca.pfx...
Enter destination keystore password:
Re-enter new password:
Enter source keystore password:
```



```
Entry for alias ca successfully imported.  
Import command completed: 1 entries successfully imported, 0 entries failed or cancelled
```

- **Borrar del almacén:**

```
keytool -delete -alias ca -keystore almacen.jks  
...  
Enter keystore password:
```

- **Importar como CA:**

```
keytool -import -trustcacerts -alias ca -file ca.cer -keystore almacen.jks  
...  
Enter keystore password:  
Owner: CN=Jose Luis Bautista Martin, OU=Desde las Horas Extras, O=Capicua, L=Guadalajara,  
ST=Jalisco, C=MX  
Issuer: CN=Jose Luis Bautista Martin, OU=Desde las Horas Extras, O=Capicua, L=Guadalajara,  
ST=Jalisco, C=MX  
Serial number: da8ea20304d35bef  
Valid from: Sat Apr 16 21:51:36 CDT 2022 until: Fri Jul 15 21:51:36 CDT 2022  
Certificate fingerprints:  
    SHA1: 7F:E2:59:C3:E9:B7:E2:D7:13:96:57:EE:14:CC:2E:15:B8:0F:0A:9F  
    SHA256:  
D1:C1:37:D7:3C:D4:06:68:B3:A3:E7:4A:12:94:F4:79:35:27:38:41:5F:16:D3:AB:26:6D:00:CD:B2:45:EA:08  
Signature algorithm name: SHA256withRSA  
Subject Public Key Algorithm: 2048-bit RSA key  
Version: 3  
  
Extensions:  
#1: ObjectId: 2.5.29.14 Criticality=false  
SubjectKeyIdentifier [  
KeyIdentifier [  
0000: 87 BF 83 59 FC 28 47 22    44 2F AA E5 DC E3 72 BD    ...Y.(G"D/....r.  
0010: 22 55 73 5B                "Us[  
]  
]  
  
Trust this certificate? [no]: yes  
Certificate was added to keystore
```

- **Importar certificado con llave privada:**

```
keytool -importkeystore -srckeystore ca.pfx -keystore almacen.jks -srcstoretype PKCS12  
...  
Importing keystore ca.pfx to almacen.jks...
```



```
Enter destination keystore password:  
Enter source keystore password:  
Existing entry alias ca exists, overwrite? [no]: yes  
Entry for alias ca successfully imported.  
Import command completed: 1 entries successfully imported, 0 entries failed or cancelled  
keytool -list -v -keystore intermediate/private/%hostcrt%.ssl-keystore
```

- **Resumen**

- Generar un almacén (con un certificado):

```
keytool -genkey -keyalg RSA -alias ca -keystore almacen.jks
```

- Listas los certificados de un almacén:

```
keytool -list -v -keystore almacen.jks
```

- Exportar certificado:

```
keytool -exportcert -alias ca -file ca.cer -keystore almacen.jks
```

- Exportar con llave privada:

```
keytool -importkeystore -srckeystore almacen.jks -destkeystore ca.pfx -  
deststoretype PKCS12
```

- Borrar del almacén:

```
keytool -delete -alias ca -keystore almacen.jks
```

- Importar como CA:

```
keytool -import -trustcacerts -alias ca -file ca.cer -keystore almacen.jks
```

- Importar certificado con llave privada:

```
keytool -importkeystore -srckeystore ca.pfx -keystore almacen.jks -  
srcstoretype PKCS12
```



- Cambiar alias:

```
keytool -changealias -keystore almacen.jks -alias ca -destalias ca2
```



RESUMEN DE OPERACIONES CON OPENSSL

OpenSSL es un conjunto de herramientas para trabajar con certificados y criptografía en general.

Generar un certificado

- Preparo el ambiente:

```
cd c:\ca
set hostcrt=ejemplohost
```

- Edito la plantilla:

```
copy intermediate\template.cnf intermediate\csr\%hostcrt%.cnf
"C:\Program Files (x86)\Notepad++\notepad++.exe" intermediate\csr\%hostcrt%.cnf
```

- Genero la llave privada:

```
openssl genrsa -aes256 -out intermediate/private/%hostcrt%.key 2048
```

- Genero la petición:

```
openssl req -config intermediate/csr/%hostcrt%.cnf -key intermediate/private/%hostcrt%.key -new -sha256 -out intermediate/csr/%hostcrt%.csr
```

- Genero el certificado:

```
openssl ca -config intermediate/csr/%hostcrt%.cnf -extensions server_cert -days 3650 -notext -md sha256 -in intermediate/csr/%hostcrt%.csr -out intermediate/certs/%hostcrt%.cer
```

- Uno la llave privada el certificado:

```
openssl pkcs12 -export -aes256 -CSP "Microsoft Enhanced RSA and AES Cryptographic Provider" -out intermediate/private/%hostcrt%.pfx -inkey intermediate/private/%hostcrt%.key -in intermediate/certs/%hostcrt%.cer
```




Revocar un certificado

- Preparo el ambiente:

```
set hostcrt=ejemplohost
```

- Revoco el certificado:

```
openssl ca -config intermediate/csr/%hostcrt%.cnf -extensions server_cert -revoke  
intermediate/certs/%hostcrt%.cer
```

- Genero la CRL:

```
openssl ca -config intermediate/openssl.cnf -gencrl -out intermediate/crl/intermediate.crl
```

Renovar con la misma clave

- Preparo el ambiente:

```
set hostcrt=ejemplohost
```

- Envío la petición:

```
openssl req -config intermediate/csr/%hostcrt%.cnf -key intermediate/private/%hostcrt%.key -new -  
sha256 -out intermediate/csr/%hostcrt%.csr
```

- Genero el certificado:

```
openssl ca -config intermediate/csr/%hostcrt%.cnf -extensions server_cert -days 3650 -notext -md  
sha256 -in intermediate/csr/%hostcrt%.csr -out intermediate/certs/%hostcrt%.cer
```

- Uno la llave privada al certificado:

```
openssl pkcs12 -export -aes256 -CSP "Microsoft Enhanced RSA and AES Cryptographic Provider" -out  
intermediate/private/%hostcrt%.pfx -inkey intermediate/private/%hostcrt%.key -in  
intermediate\certs\%hostcrt%.cer
```



Plantilla

Se muestra una plantilla básica con los campos modificables más comunes marcados.

```
# OpenSSL intermediate CA configuration file.
# Copy to `/root/ca/intermediate/openssl.cnf`.

[ ca ]
# `man ca`
default_ca = CA_default

[ CA_default ]
# Directory and file locations.
dir                = ./intermediate
certs              = $dir/certs
crl_dir            = $dir/crl
new_certs_dir      = $dir/newcerts
database           = $dir/index.txt
serial             = $dir/serial
RANDFILE           = $dir/private/randfile.rand

# The root key and root certificate.
private_key        = $dir/private/intermediate.key
certificate         = $dir/certs/intermediate.cer

# For certificate revocation lists.
crlnumber          = $dir/crlnumber
crl                = $dir/crl/intermediate.crl.pem
crl_extensions     = crl_ext
default_crl_days   = 3650

# SHA-1 is deprecated, so use SHA-2 instead.
default_md         = sha256

name_opt           = ca_default
cert_opt          = ca_default
default_days       = 375
preserve          = no
policy             = policy_loose

[ policy_strict ]
# The root CA should only sign intermediate certificates that match.
# See the POLICY FORMAT section of `man ca`.
countryName       = match
stateOrProvinceName = match
organizationName  = match
organizationalUnitName = optional
commonName        = supplied
emailAddress      = optional

[ policy_loose ]
```



```
# Allow the intermediate CA to sign a more diverse range of
certificates.
# See the POLICY FORMAT section of the `ca` man page.
countryName          = optional
stateOrProvinceName  = optional
localityName         = optional
organizationName      = optional
organizationalUnitName = optional
commonName           = supplied
emailAddress          = optional

[ req ]
# Options for the `req` tool (`man req`).
default_bits          = 2048
distinguished_name    = req_distinguished_name
string_mask           = utf8only

# SHA-1 is deprecated, so use SHA-2 instead.
default_md            = sha256

# Extension to add when the -x509 option is used.
x509_extensions       = v3_ca

[ req_distinguished_name ]
# See <https://en.wikipedia.org/wiki/Certificate\_signing\_request>.
countryName           = Country Name (2 letter code)
stateOrProvinceName    = State or Province Name
localityName          = Locality Name
0.organizationName     = Organization Name
organizationalUnitName = Organizational Unit Name
commonName            = Common Name
emailAddress           = Email Address

# Optionally, specify some defaults.
countryName_default    = MX
stateOrProvinceName_default = Jalisco
localityName_default   = Guadalajara
0.organizationName_default = Capicua
organizationalUnitName_default = Desde las horas extras
commonName_default     = desdelashorasextras
emailAddress_default    = correo@dominio.com

[ v3_ca ]
# Extensions for a typical CA (`man x509v3_config`).
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid:always,issuer
basicConstraints = critical, CA:true
keyUsage = critical, digitalSignature, cRLSign, keyCertSign

[ v3_intermediate_ca ]
# Extensions for a typical intermediate CA (`man x509v3_config`).
```



```
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid:always,issuer
basicConstraints = critical, CA:true, pathlen:0
keyUsage = critical, digitalSignature, cRLSign, keyCertSign

[ usr_cert ]
# Extensions for client certificates (`man x509v3_config`).
basicConstraints = CA:FALSE
nsCertType = client, email
nsComment = "OpenSSL Generated Client Certificate"
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid,issuer
keyUsage = critical, nonRepudiation, digitalSignature, keyEncipherment
extendedKeyUsage = clientAuth, emailProtection

[ server_cert ]
# Extensions for server certificates (`man x509v3_config`).
basicConstraints = CA:FALSE
nsCertType = server
nsComment = "OpenSSL Generated Server Certificate"
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid,issuer:always
keyUsage = critical, digitalSignature, keyEncipherment
extendedKeyUsage = serverAuth
crlDistributionPoints = URI:http://host/crl/intermediate.crl

[ crl_ext ]
# Extension for CRLs (`man x509v3_config`).
authorityKeyIdentifier=keyid:always

[ ocsp ]
# Extension for OCSP signing certificates (`man ocsp`).
basicConstraints = CA:FALSE
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid,issuer
keyUsage = critical, digitalSignature
extendedKeyUsage = critical, OCSPSigning
```



SOLICITUD DE CERTIFICADOS A ENTIDAD CERTIFICADORA DE UN DIRECTORIO ACTIVO

Si nuestra empresa maneja Directorio Activo es posible configurar una entidad certificadora para poder generar certificados para los servidores o máquinas de nuestra intranet.

A pesar de que se pueda hacer gráficamente, la forma más sencilla y rápida de generarlo es median el uso y archivos inf, a las que solo vamos a cambiar los datos que requerimos para cada servidor, el proceso es el siguiente:

1. Genere un archivo de nombre host.inf (siendo host el nombre del servidor para el cual se quiere el certificado) con la siguiente información:

```
;-----  
[Version]  
Signature="$Windows NT$"  
  
[NewRequest]  
Subject = "CN=desdelashorasextras, OU=Desde las Horas Extras, O=Capicua,  
L=Guadalajara, S=Jalisco, C=MX"  
KeySpec = 1  
KeyLength = 2048  
Exportable = TRUE  
MachineKeySet = TRUE  
SMIME = False  
PrivateKeyArchive = FALSE  
UserProtected = FALSE  
UseExistingKeySet = FALSE  
; Seleccione el proveedor adecuado  
ProviderName = "Microsoft Enhanced RSA and AES Cryptographic Provider"  
ProviderType = 24  
;ProviderName = "Microsoft RSA SChannel Cryptographic Provider"  
;ProviderType = 12  
RequestType = PKCS10  
KeyUsage = 0xa0  
  
[RequestAttributes]
```



CertificateTemplate = WebServer

[EnhancedKeyUsageExtension]

OID=1.3.6.1.5.5.7.3.1 ; this is for Server Authentication / Token

Signing

;-----

2. Ejecute el siguiente comando (en el host donde estará la llave privada) para generar la petición:

```
certreq -new host.inf host.csr
```

3. Ejecute el siguiente comando (en la entidad certificadora) para generar el certificado público:

```
certreq -submit <#=NombreCertificado#>.csr
```

4. Importe el certificado público en el host donde genero la petición, se generará automáticamente el certificado privado.
5. Respalde apropiadamente el .pfx (certificado con llave privada).




SOLICITUD DE CERTIFICADOS DESDE IIS

Es posible crear una solicitud directamente desde IIS, para lo cual entramos en el administrador y seleccionamos la opción de certificados, para posteriormente seleccionar **“Solicitud de certificados”**.

A partir de allí, mediante un asistente podremos generar una “Solicitud de Certificados”, que podemos enviar a una entidad certificadora para que lo firme.

Cuando nos devuelva el archivo .cer, usáramos la opción “Completar Solicitud” para obtener el archivo .pfx.

Solicitar certificado ? X

 **Propiedades de nombre distintivo**


Especifique la información requerida para el certificado. Estado o provincia y Ciudad o localidad deben ser nombres oficiales y no deben contener abreviaturas.

Nombre común:	<input type="text" value="desdelashorasextras"/>
Organización:	<input type="text" value="Capicua"/>
Unidad organizativa:	<input type="text" value="Desde las Horas Extre"/>
Ciudad o localidad:	<input type="text" value="Guadalajara"/>
Estado o provincia:	<input type="text" value="Jalisco"/>
País o región:	<input type="text" value="MX"/>

Anterior **Siguiente** Finalizar Cancelar



Solicitar certificado ? X


 **Propiedades de proveedor de servicios criptográficos**

Seleccione un proveedor de servicios criptográficos y una longitud en bits. La longitud en bits de la clave de cifrado determina la seguridad de cifrado del certificado. Cuanto mayor sea la longitud en bits, más segura. Sin embargo, una longitud en bits grande puede mermar el rendimiento.

Proveedor de servicios criptográficos:

Longitud en bits:

Solicitar certificado ? X

 **Nombre de archivo**

Especifique un nombre para la solicitud de certificado. Esta información se puede enviar a una entidad de certificación para que la firme.

Especificar un nombre de archivo para la solicitud de certificado:



ANEXO I: EJEMPLOS DE CERTIFICADOS

Tipo	Nombre Empresarial	Nombre Común	Nombre archivo	Password	email
Certificadora Raiz	Capacitacion Criptografia Certificados Privacidad	Capacitacion Certificado Raiz	ca	ejemplo	
Certificadora Intermediaria	Capacitacion Criptografia Certificados Privacidad	Capacitacion Certificado Intermediario	intermediate	ejemplo	
Certificado Personal Servidor	Capacitacion Criptografia Certificados Privacidad	ejemplohost	ejemplohost	ejemplo	
Certificado Personal GPG	Capacitacion Criptografia Certificados Privacidad	Certificado Personal GPG 001	certificadogpg00 1	ejemplo	certificadogpg001@e jemplo.com
Certificado Personal GPG	Capacitacion Criptografia Certificados Privacidad	Certificado Personal GPG 002	certificadogpg00 2	ejemplo	certificadogpg002@e jemplo.com



ANEXO II: EXTENSIONES DE ARCHIVOS CRIPTOGRAFICOS

Esta información esta extraída de:

<https://serverfault.com/questions/9708/what-is-a-pem-file-and-how-does-it-differ-from-other-openssl-generated-key-file>

- **.csr** this is a Certificate Signing Request. Some applications can generate these for submission to certificate-authorities. The actual format is PKCS10 which is defined in [RFC 2986](#). It includes some/all of the key details of the requested certificate such as subject, organization, state, whatnot, as well as the *public key* of the certificate to get signed. These get signed by the CA and a certificate is returned. The returned certificate is the public *certificate* (which includes the public key but not the private key), which itself can be in a couple of formats.
- **.pem** Defined in RFC's [1421](#) through [1424](#), this is a container format that may include just the public certificate (such as with Apache installs, and CA certificate files `/etc/ssl/certs`), or may include an entire certificate chain including public key, private key, and root certificates. Confusingly, it may also encode a CSR (e.g. as used [here](#)) as the PKCS10 format can be translated into PEM. The name is from [Privacy Enhanced Mail \(PEM\)](#), a failed method for secure email but the container format it used lives on, and is a base64 translation of the x509 ASN.1 keys.
- **.key** This is a PEM formatted file containing just the private-key of a specific certificate and is merely a conventional name and not a standardized one. In Apache installs, this frequently resides in `/etc/ssl/private`. The rights on these files are very important, and some programs will refuse to load these certificates if they are set wrong.
- **.pkcs12 .pfx .p12** Originally defined by RSA in the [Public-Key Cryptography Standards](#) (abbreviated PKCS), the "12" variant was originally enhanced by Microsoft, and later submitted as [RFC 7292](#). This is a passworded container format that contains both public and private certificate pairs. Unlike .pem files, this container is fully encrypted. Openssl can turn this into a .pem file with both public and private keys: `openssl pkcs12 -in file-to-convert.p12 -out converted-file.pem -nodes`

A few other formats that show up from time to time:

- **.der** A way to encode ASN.1 syntax in binary, a .pem file is just a Base64 encoded .der file. OpenSSL can convert these to .pem (`openssl x509 -inform der -in to-convert.der -`



out converted.pem). Windows sees these as Certificate files. By default, Windows will export certificates as .DER formatted files with a different extension. Like...

- **.cert .cer .crt** A .pem (or rarely .der) formatted file with a different extension, one that is recognized by Windows Explorer as a certificate, which .pem is not.
- **.p7b .keystore** Defined in [RFC 2315](#) as PKCS number 7, this is a format used by Windows for certificate interchange. Java understands these natively, and often uses .keystore as an extension instead. Unlike .pem style certificates, this format has a *defined* way to include certification-path certificates.
- **.crl** A certificate revocation list. Certificate Authorities produce these as a way to de-authorize certificates before expiration. You can sometimes download them from CA websites.

In summary, there are four different ways to present certificates and their components:

- **PEM** Governed by RFCs, it's used preferentially by open-source software. It can have a variety of extensions (.pem, .key, .cer, .cert, more)
- **PKCS7** An open standard used by Java and supported by Windows. Does not contain private key material.
- **PKCS12** A Microsoft private standard that was later defined in an RFC that provides enhanced security versus the plain-text PEM format. This can contain private key material. It's used preferentially by Windows systems, and can be freely converted to PEM format through use of openssl.
- **DER** The parent format of PEM. It's useful to think of it as a binary version of the base64-encoded PEM file. Not routinely used by much outside of Windows.