

**Tecnológico Nacional de México**

**Instituto Tecnológico de Reynosa**

Materia: Sistemas Operativos 2

ACTIVIDAD11: Investigación exhaustiva del Tema05: Seguridad con sus 4 subtemas.

Tema 5: Seguridad

Alumno: Castillo Jr. Gregorio

Numero de control: 19580589

Correo electrónico: L19580589@reynosa.tecnm.mx

7mo Semestre Matutino Salón 7

Docente: Mario José Santiago Sánchez

Fecha de entrega: 03/12/2022



|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Consecutivo** | **Numero de**  **control** | **Apellido Paterno** | **Apellido Materno** | **Nombres(s)** | **Correo electrónico**  **Institucional** | **Firma de que está**  **autorizando que se**  **entregue esta Publicación** | **Fotografía del rostro de cada**  **Integrante del equipo (selfie)** |
| **1** | **19580589** | **Castillo Jr** |  | **Gregorio** | **L@19580589@reynosa.tecnm.mx** |  |  |
| **2** | **19580595** | **Flores** | **Acosta** | **Sheila Lizeth** | **L@19580595@reynosa.tecnm.mx** |  |  |
| **3** | **19580867** | **Morales** | **Calixto** | **Daniel Alexander** | **L@19580867@reynosa.tecnm.mx** | No hay descripción disponible. |  |

Equipo #12

# Tabla de contenidos

[Tabla de contenidos 3](#_Toc120037254)

[Contenido 4](#_Toc120037255)

[**3.1 Tolerancia a fallas** 4](#_Toc120037256)

[**3.2 Introducción atenuación de un proceso** 6](#_Toc120037257)

[**3.3 Comunicación confiable cliente-servidor** 14](#_Toc120037258)

[**3.4 Comunicación confiable en grupo** 19](#_Toc120037259)

[**3.5 Recuperación** 26](#_Toc120037260)

[**Conclusiones** 34](#_Toc120037261)

[**Herramientas y recursos** 35](#_Toc120037262)

[**Bibliografía** 36](#_Toc120037263)

# Contenido

## **5.1 Introducción a la seguridad**

Podemos definir la seguridad de un sistema como “Conjunto de métodos y herramientas destinados a proteger la información y por ende los sistemas informáticos”

Cuando hablamos de la seguridad de un sistema informático, hablamos de todos aquellos métodos que tenemos para proteger a nuestros dispositivos, y más esencialmente la información que estos contienen, información que puede ser delicada, ya sea que hablamos de datos personales como nombre, dirección, cuentas de banco etc. Hasta incluso proyectos o algún medio para el cual estamos trabajando y esto no se debe de saber, es por ello que a lo largo de los años se han ido implementando métodos para hacer nuestros dispositivos mas seguros, per estos muchas veces vuelven a ser vulnerados, a través de las diversas técnicas que se han inventado para vulnerar los dispositivos.

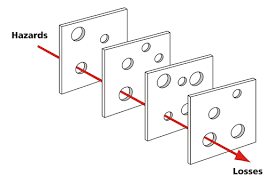
Para considerar seguro un sistema este tiene que contar con 3 requisitos esenciales:

Confiabilidad: Tenemos que estar seguros de que nuestro sistema podrá detener en caso de ser necesario la amenaza, siendo este de los factores mas importantes en cuanto a seguridad se refiere

Integridad : Buscamos que el sistema no sea destruido con facilidad garantizando que es capaz de realizar las tareas para las cuales está hecho

Disponibilidad: De nada nos sirve tener el sistema mas seguro y sofisticado del mundo, si cuando lo necesitamos, es decir bajo un ataque, este sistema no puede ser accionado o implementado de forma correcta

Pues bien cuando hablamos de técnicas para vulnerar dispositivos, nos referimos desde las mas sencillas como lo es el “phishing” hasta las mas complejas como lo es un “zero-day-exploid” estas vulnerabilidades son propias de los sistemas, ya que al estar construidos por personas, nunca podemos asegurarnos en que no existan fallas pues bien, si esto existe, siempre hay una forma de corregir un error, de esta forma entra lo que conocemos como “parches”, y son aquellas actualizaciones que nos pide el sistema cada cierto tiempo, si bien los parches son creados para corregir errores, muchas veces estos también agregan nuevos errores, a lo que estamos en un ciclo repetitivo donde para solucionar un problema agregamos un parche, pero este tiene otro problema que solucionamos con un segundo parche, y este último tiene otro problema que solucionamos con otro parche y así sucesivamente, si bien parece ilógico querer solucionar de esta forma los sistemas, la verdad es que hoy en día es lo más efectivo y se le conoce como “modelo de queso suizo”, si bien parecerá extraño la verdad es que es bastante fácil de explicar, puesto que este queso cuenta con agujeros que vendrían a ser los errores, y el queso es el parche el cual repara el agujero de la capa anterior, o en este caso, tapa el agujero del queso anterior, pues bien, hoy en día es sumamente difícil vulnerar sistemas gracias a este modelo, ya que se tiene que tener las condiciones exactas para que la seguridad de un sistema falle, sin embargo no es imposible



Entonces, si bien decimos que los sistemas son seguros, como es que a diario nos topamos con información que dice lo contrario, pues bueno esto es por aquellos mecanismos del sistema que son mas débiles, estamos hablando del factor humano, el factor humano aunque no lo parezca, es uno de los mecanismos más débiles de un sistema, si bien este no está interno dentro del sistema, interactúa con el y es donde entran las vulnerabilidades y ataques como lo es el phishing, que es cuando un atacante se hace pasar por una entidad oficial, ya sea un banco, empresa, persona, tienda, o incluso páginas, el individuo ingresa datos personales como por ejemplo las tarjetas de crédito y débito, siendo así el atacante a tener accesos a sus cuentas de forma “legitima” y todo gracias al factor humano, o incluso información que pareciera tan irrelevante como el nombre de su perro, cuantos tíos tiene, a donde va comúnmente de vacaciones o etc. Datos que se pueden extraer en una conversación normal y pueden ser indicios para ayudar a “recuperar una contraseña” gracias a las preguntas de seguridad

Pues bien, gracias a todo lo anterior nace la necesidad de corregir esos errores y prevenirlos, el factor humano al ser más débil, solo queda aprender sobre las tecnologías y saber diferenciar cuando una entidad es oficial o no, e incluso instruirse en como es que puede evitar caer en este tipo de fallas, pero en cuanto a la tecnología hablamos, cada vez se implementan más y nuevas funciones, entre ellas podríamos decir la de doble cifrado o ingreso de dos pasos, donde nosotros ponemos la contraseña y adicional a ello, el sistema manda un código al correo, teléfono, etc. De dicha cuenta anidada, permitiendo que, de entrar a un sistema por una clave, se dificulte mas teniendo que tener ambas, ya que, de no tener ambos códigos, no tendríamos acceso, del mismo modo de recibir un mensaje con el código, y no ser nosotros quien está intentando acceder, sabríamos que nuestro sistema se habría visto comprometido, y es necesario hacer una actualización de nuestra contraseña.

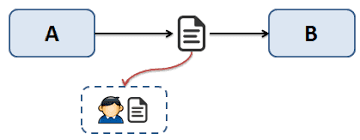
Pero si bien, los sistemas pueden ser seguros, también existen otros factores que tenemos que tener en cuenta a la hora de hacer el envió de información o uso de aplicaciones y es el como es que los mandamos, si bien podemos controlar es acceso y envío de ciertos documentos, esto no impide que las redes no sean vulnerables, es por eso que a lo largo de los años se ha trabajado en métodos como lo es el SSH y demás métodos para permitir una conexión segura y funcional

## **5.2 Canales seguros**

**¿Qué son los canales seguros?**

Los canales seguros están enfocados en la protección de la comunicación dentro de un sistema distribuidos principalmente, los servidores suelen ser distribuidos o replicados que también actúan como cliente con respecto a los demás servidores. En sí, implementar un sistema distribuido se reduce en dos temas predominantes entre cliente-servidor y la autenticación.

La comunicación segura requiere autentificación de las partes que se comunican. En varios casos es necesario garantizar la integridad y posiblemente la confidencialidad de un mensaje. Como parte del problema, se tiene que considerar a fondo la protección de la comunicación dentro de un grupo de servidores. Vemos que, dentro de la autenticación, una vez que el servidor haya aceptado la solicitud de un cliente ¿Cómo sabremos si dicho cliente esta autorizado para poder realizar la solicitud? Si, pues aquí es donde entra la autorización que controla el problema de controlar los accesos a los recursos, lo cual es analizado extensamente.



**AUTENTICACIÓN**

Para empezar, debemos de saber que la autenticación y la integridad de los mensajes no se puede realizar una sin la otra. Es decir, consideremos que un sistema distribuido que soporta la autenticación de dos partes en comunicación, pero que no nos proporciona mecanismos para poder garantizar la integridad de los mensajes. En un sistema así, se puede saber con seguridad que la otra persona es el remitente del mensaje. Sin embargo, si a nosotros no se nos puede garantizar que el mensaje no se ha modificado en el momento de la transmisión ¿de qué nos servirá saber que la otra persona nos mandó el mensaje? Bueno, se puede decir que únicamente la integridad del mensaje para informarle que gano como 10 billones en la lotería ¿Qué tan feliz se puede estar si no puede verificar si el mensaje fue enviado por el organizador de dicha lotería?

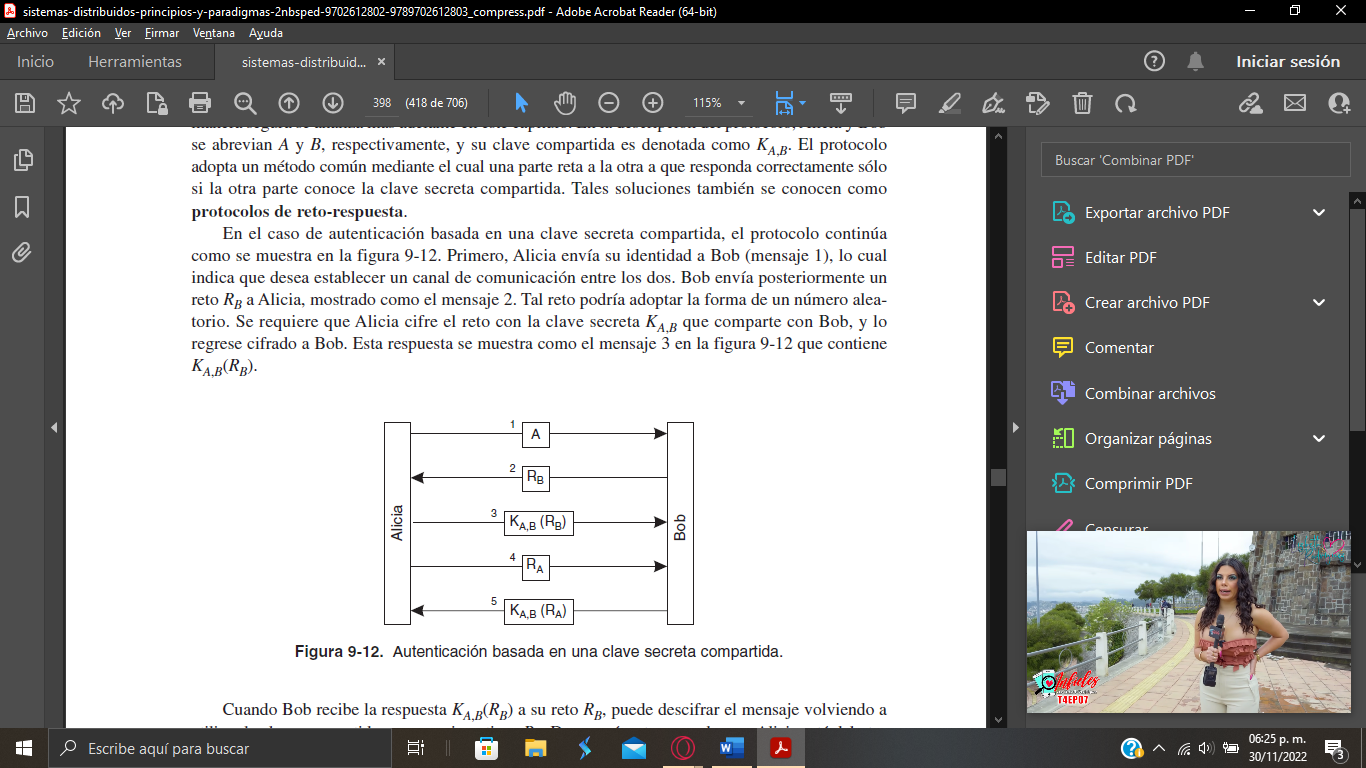
Por eso la autentificación y la integridad del mensaje deben de ir de la mano como si fuesen hermanos. En los protocolos, la combinación funciona aproximadamente como sigue. Nuevamente podemos suponer que nosotros nos queremos comunicar con la otra persona y la otra persona toma la iniciativa al querer establecer un canal. La otra persona nos envía un mensaje, o de lo contrario a una tercera parte confiable que le ayudaría establecer el canal. Una vez que se pudo establecer el canal, la otra persona que nos esta hablando y sepamos con toda la seguridad que, si estamos hablando con dicha persona, podremos intercambiar mensajes.

Para poder garantizar la integridad de los mensajes de datos intercambiados una vez autentificados, es común utilizar la criptografía de clave secreta por medio de claves de sesión. La clave de sesión es una clave secreta compartida que es utilizada para cifrar mensajes en cuanto a la integridad y la confidencialidad. Tal clave se usa generalmente solo para el canal existente. Cuando el canal se cierra, la clave de sesión asociada es desechada (destruida).

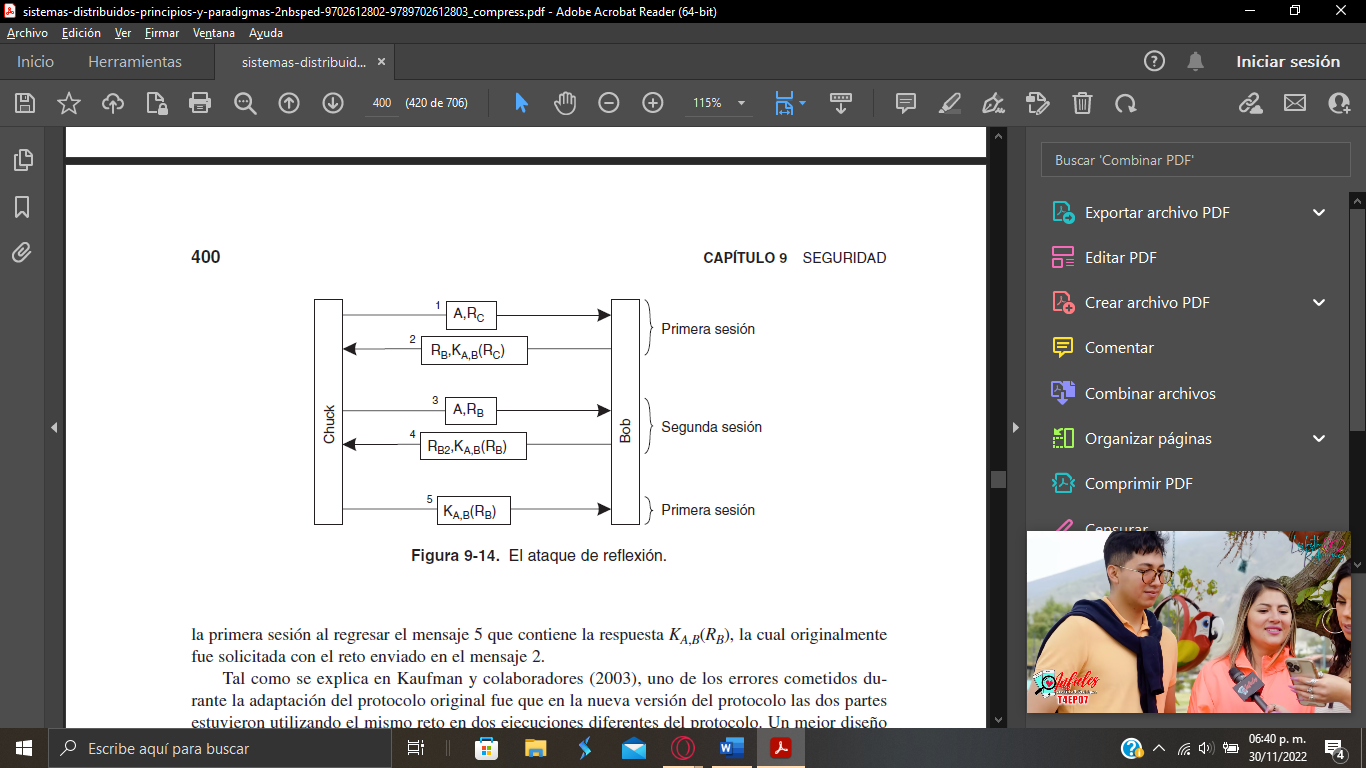
***Autenticación basada en una clave secreta compartida***

Retomando el ejemplo anterior, tomaremos un vistazo al protocolo de autenticación de clave secreta que tiene. Dentro de la descripción del protocolo del ejemplo anterior que vimos la otra persona (pongámosle A) y nosotros (seremos B), respectivamente, y la clave compartida se denota como Ka,b. El protocolo adopta un método común con el que una parte reta a la otra a que responda de manera correcta solo si la otra parte conoce la clave secreta compartida. Esta solución también suele conocerse como protocolos de reto-respuesta.

En el caso de la autentificación basada en una clave secreta compartida, el protocolo todavía siegue, como se muestra en la siguiente imagen. Primero A envía su entidad a B que sería el primer mensaje, lo cual indica que desea establecer un canal de comunicación entre los dos. B envía un reto Rb a la persona A, que sería el segundo mensaje. Ese reto podría adoptar la forma de un número aleatorio. Se requiere que la persona A cifre el mensaje el reto de la clave Ka,b que comparte con B y lo regrese cifrado a B, la respuesta seria el tercer mensaje.



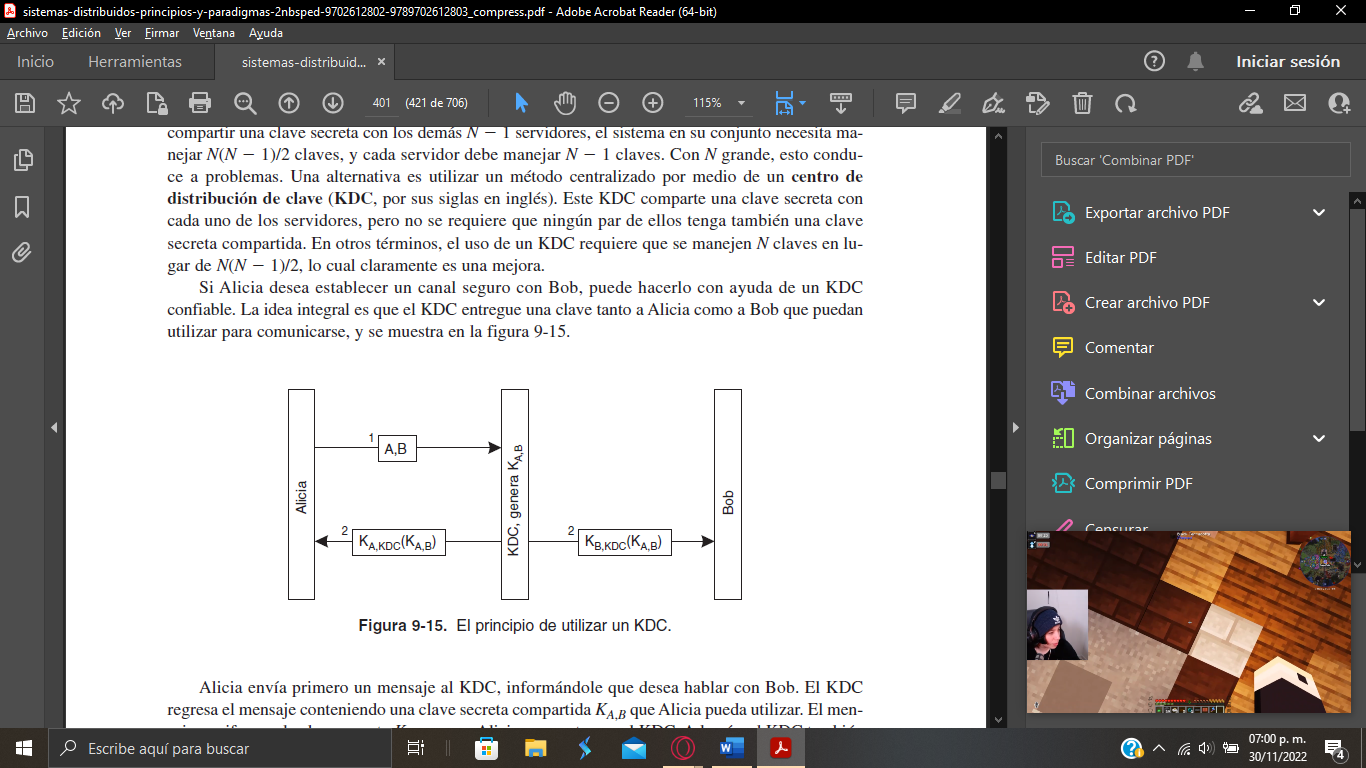
Cuando B reciba la respuesta Ka,b(Rb) a su reto Rb, se puede descifrar el mensaje volviendo a utilizar la clave compartida para ver si contiene Rb. Si resulta todo bien, entonces se sabrá que A esta del otro lado, terminando de comprobar que si es la otra persona, se podrá observar que la persona no sabe si realmente con la que se está comunicando es B. Así que se envía un reto Ra (cuarto mensaje), el cual B responde regresando Ka,b(Ra), llegando así el quinto mensaje. Cuando A lo descifra con Ka,b y ve su Ra, se da cuenta de que esta comunicándose con B.



**Autentificación mediante un centro de distribución clave**

Uno de los problemas que se llegan a presentar con el uso de una clave secreta compartida para autentificación es su estabilidad. Si el sistema distribuido tiene N servidores, y cada servidor requiere compartir una clave secreta con los demás N-1 servidores, el sistema en su conjunto necesita manejar N(N-1)/2 claves y cada servidor debe de manejar N-1 claves. Una de las alternativas es usar el método centralizado por medio de un centro de distribución de clava (KDC). Este comparte una clave secreta con cada uno de los servidores.

Si la persona A quiere crear un canal seguro con B, puede realizarlo con la ayuda del KDC confiable. La idea integral que el KDC entregue una clave tanto A como B puedan utilizar para comunicarse.



**INTEGRIDAD Y CONFIDENCIALIDAD DEL MENSAJE**

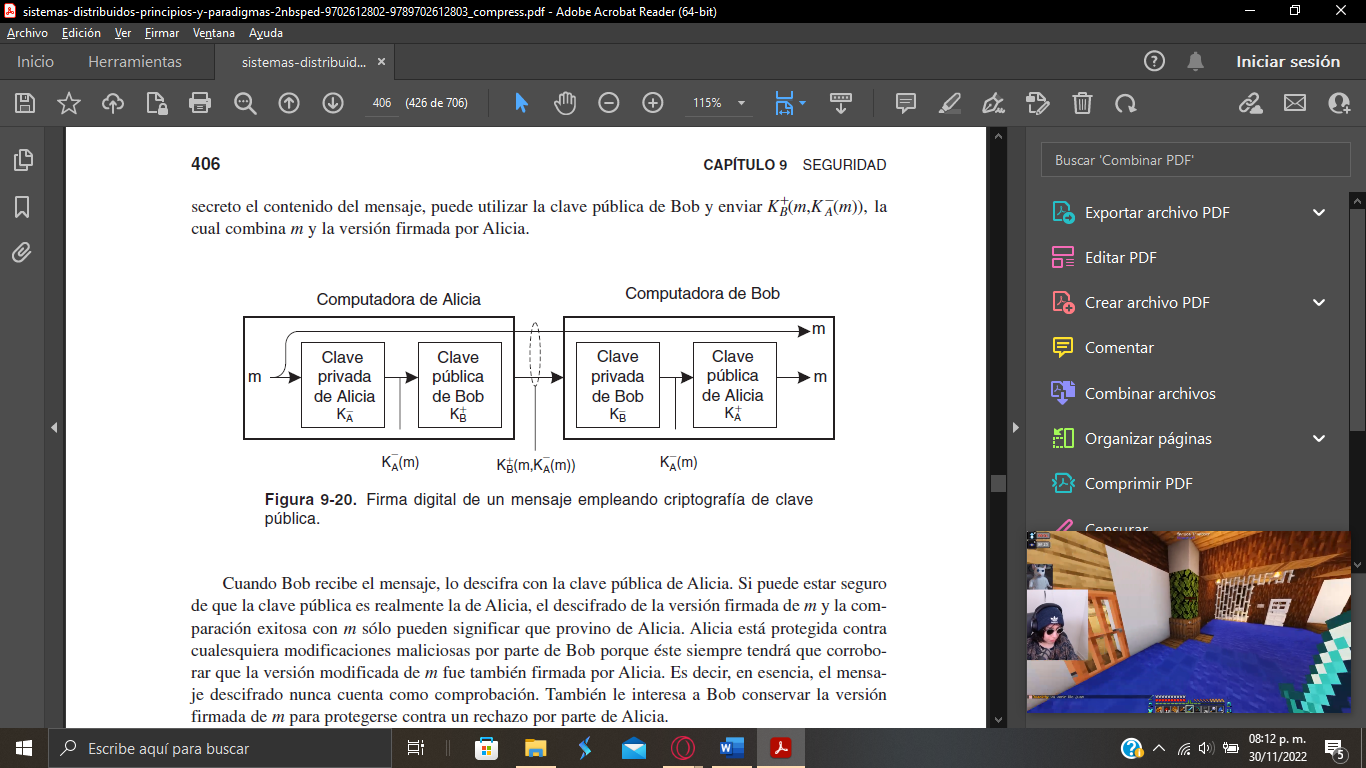
Un canal que es seguro debe de proporcionar garantías en cuanto a integridad y confidencialidad del mensaje. La integridad del mensaje significa que los mensajes están protegidos contra la modificación subrepticia; la confidencialidad asegura que los mensajes no pueden ser interceptados y leídos por fisgones.

**Firmas digitales**

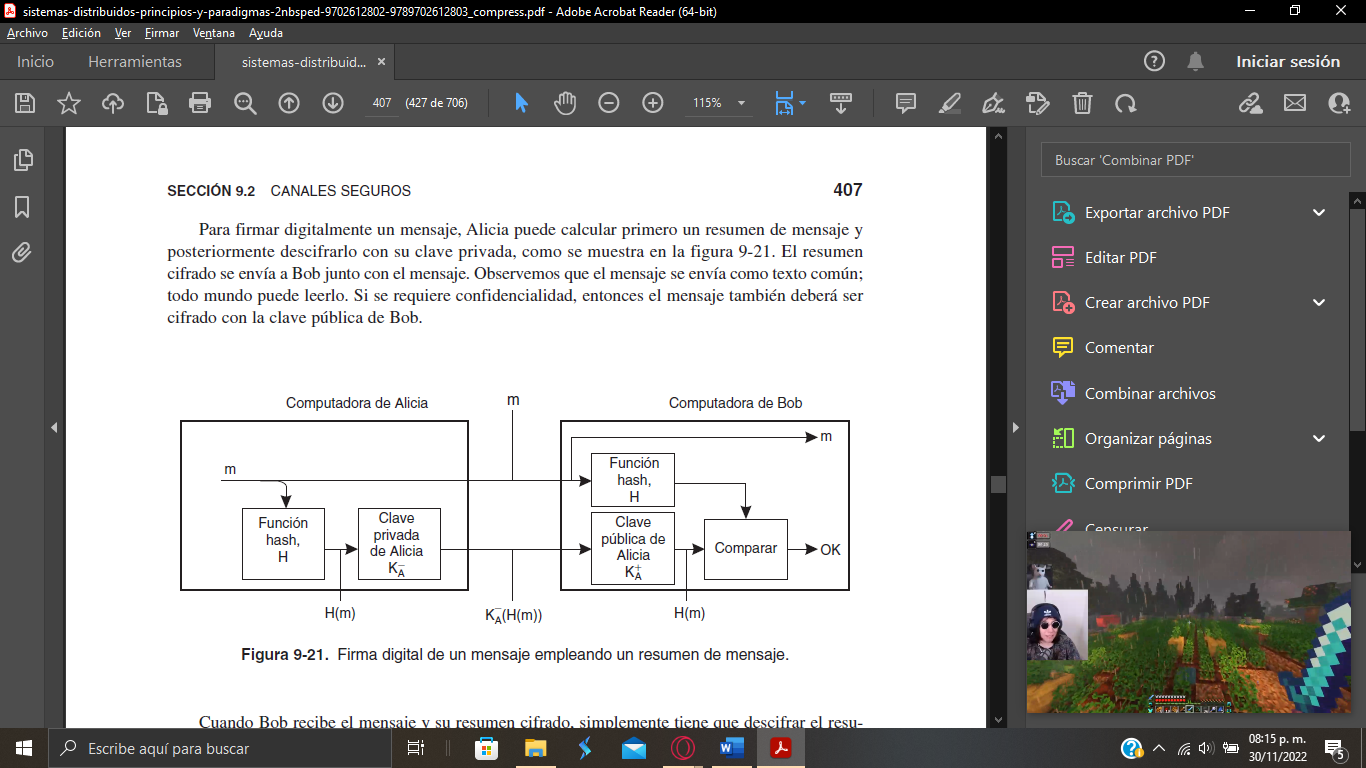
La integridad de un mensaje puede llegar más allá de la transferencia a través de un canal seguro, consideremos que B acaba de venderle a A un disco fonográfico de colección en $500. Esta operación fue realizada en correo electrónico. Al final A manda un mensaje a B para confirmarle que compra el disco. Solo hay dos cuestiones para tener en cuenta aquí:

1. A tiene que estar segura de que B no cambiara el precio del disco y que diga que A prometió más dinero.
2. B tiene que estar seguro de que A no puede retractarse de que envió el mensaje, es decir que cambio de idea.

Estas pueden ser solventas si A firma digitalmente el mensaje, de tal forma que su firma quede vinculada a su contenido. Existen varias formas de colocar firmas digitales, una muy popular es utilizar un criptosistema de clave publica tal como RSA. Cuando A redacta un mensaje m para B, lo cifra con su clave privada Ka- y se lo envía a B. si también desea mantener en secreto el contenido del mensaje, puede utilizar clave publica de B y enviar Kb+(m, Ka-(m)), la cual combina m y la versión firmada por A.



Para poder firmar digitalmente un mensaje, A puede calcular primero un mensaje y posteriormente descifrarlo con su clave privada.



**Claves de sesión**

La clave de sesión se puede desechar en forma segura cuando el canal ya no este en uso. Una alternativa seria utilizar las mismas claves para la confidencialidad que las utilizadas para establecer el canal seguro. No obstante, hay varios beneficios que derivan del uso de la clave de sesión.

Cuando se utiliza una clave a menudo, es más fácil revelarla. En cierto sentido, las claves criptográficas se acaban tal como las claves ordinarias. Lo más básico es que si un intruso es capaz de interceptar una gran cantidad de datos cifrados con la misma clave, llega a ser posible implementar ataques para determinar ciertas características de las claves utilizadas.

**COMUNICACIÓN SEGURA DE UN GRUPO**

A menudo es necesario habilitar la comunicación segura entre más de dos partes. Ya que dentro de los sistemas distribuidos es muy seguido habilitar la comunicación segura entre más de dos partes.

**Comunicación confidencial de un grupo**

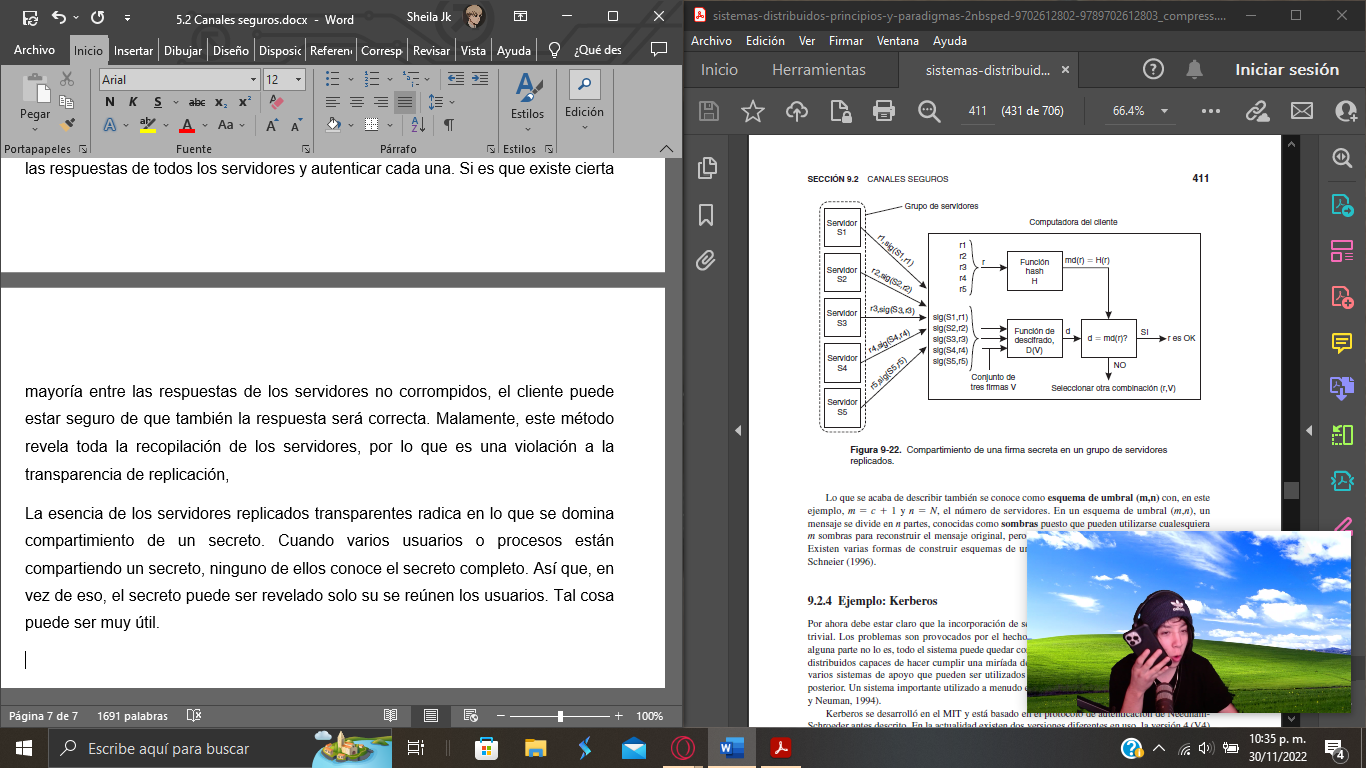
En consideración de que el problema es proteger la comunicación entre un grupo de N usuarios contra el fisgoneo para poder garantizar la confidencialidad, un esquema simple es que todos los miembros del grupo compartan una clave secreta, esta se estará utilizando para cifrar y descifrar todos los mensajes transmitidos entre los miembros del grupo. Todos los miembros estarán compartiendo la clave secreta, es necesario poder confiar en que la conservaran secreta. Es el único requisito previo proporciona que el uso de una sola clave secreta compartida para la comunicación confidencial de un grupo sea vulnerable a ataques en comparación con los canales seguros bipartitas.

**Servidores replicados seguros**

Considerando un problema donde un cliente emite una petición a un grupo de servidores replicados. Estos pueden haber sido replicados por razones de tolerancia o desempeño defectuoso, pero en todo caso, el cliente está esperando una respuesta confiable, en otras palabras, a pesar de que el grupo de los servidores este sujeto a fallas bizantinas como se vio en el capitulo anterior, un cliente espera que la respuesta regresada no haya sido sometida a un ataque. Ese ataque podría ocurrir si solo uno o más servidores fueron corrompidos por un intruso.

La solución para poder proteger al cliente contra los atacantes es poder recopilar las respuestas de todos los servidores y autenticar cada una. Si es que existe cierta mayoría entre las respuestas de los servidores no corrompidos, el cliente puede estar seguro de que también la respuesta será correcta. Malamente, este método revela toda la recopilación de los servidores, por lo que es una violación a la transparencia de replicación,

La esencia de los servidores replicados transparentes radica en lo que se domina compartimiento de un secreto. Cuando varios usuarios o procesos están compartiendo un secreto, ninguno de ellos conoce el secreto completo. Así que, en vez de eso, el secreto puede ser revelado solo su se reúnen los usuarios. Tal cosa puede ser muy útil.



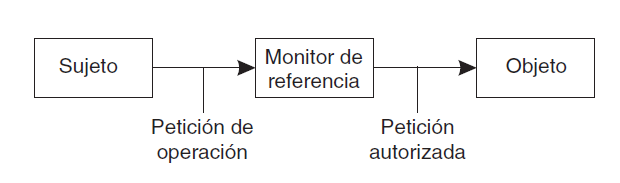
## **5.3 Control de acceso**

En el modelo cliente-servidor, una vez que un cliente y un servidor establecen un canal seguro, el cliente puede emitir peticiones que debemos ejecutar por el mismo servidor. Las peticiones implican realizar operaciones en recursos controlados por el servidor. Una de las situaciones en general es en la que un servidor de objetos que tiene varios objetivos bajo su control. La petición de un cliente implica, por lo general, llegar invocar un método de un objeto especifico hace que dicha petición puede llegar hacer realizada solo si el cliente tiene los derechos de acceso suficientes para conseguir dicha innovación.

De una forma general, la verificación de los derechos de acceso se conoce como control de acceso, mientras que la autorización se refiere a la concesión de derechos de acceso.

Como sabemos existen muchas formas de acceso. En el primer lugar, abordamos algunos de los temas generales prestando atención especial a los diferentes modelos de manejo del control de acceso. Es una de las maneras importantes de poder controlar el acceso a recursos es construir un cortafuego que nos proteja nuestras aplicaciones o incluso nuestra red completa. Los cortafuegos se estudian por separado. Con el advenimiento del código de movilidad, el control de acceso ya no podía realizarse únicamente con los métodos tradicionales.

Para poder entender de mejor manera el control-acceso, podemos adoptar un modelo. Este modelo consta de sujetos que emiten una petición de acceso a un objeto. Como sabemos un objeto es muy parecido a los objetos que han estado analizando hasta ahora. Podemos llegar a pensar que los objetos como encapsular su propio estado e implementar las operaciones en dicho estado. Las operaciones de un objeto que los sujetos pueden solicitar se realicen están disponibles mediante interfaces. Los sujetos pueden ser considerados mejor como procesos que actúan a nombre de los usuarios, aunque en algunas ocasiones pueden ser objetos que necesiten los servicios de otros objetos para realizar su propio trabajo.



Para poder controlar el acceso a un objeto es todo lo que se refiere a protegerlo contra invocaciones realizadas por sujetos a los que no les está permitido tener algún método especifico (o incluso ninguno) de los métodos realizados. También, la protección puede incluir temas de administración de objetos tales como crear, renombrar, o eliminar objetos. La protección a menudo la aplicamos mediante un programa llamado monitor de referencia. El monitor de referencia registra que sujeto puede hacer que cosa, y decide si a un objeto se le permite realizar una operación especifica. A este monitor se le convoca cada que vez que un objeto es invocado. Por consiguiente, es un extremo importante que el propio monitor de referencia sea a prueba de instrucciones: un atacante no debe ser capaz de inmiscuirse con él.

**Matriz de control de acceso.**

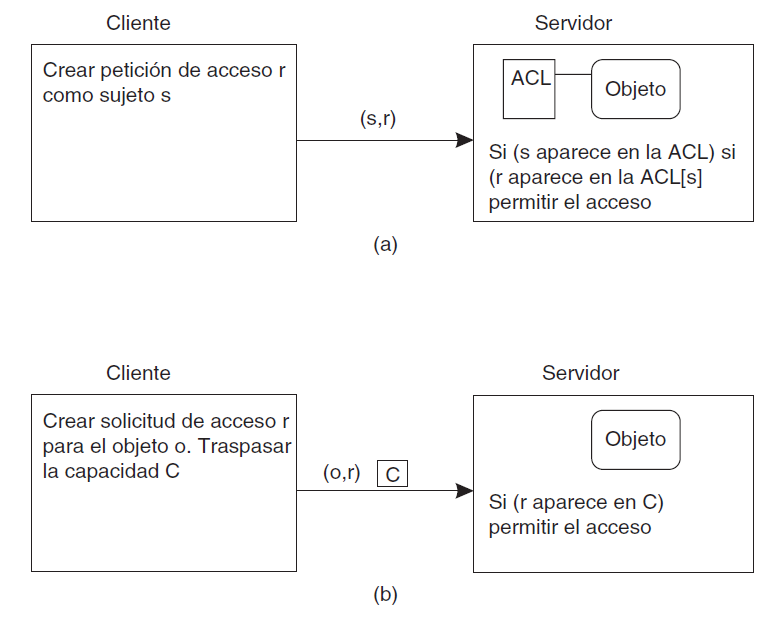
Uno de los métodos comunes que se usa para poder modelar los derechos de acceso a sujetos con respecto a objetos es poder construir una matriz de control de acceso. En matriz, cada sujeto esta representado por una fila y cada uno de los objetos por una columna. Si la matriz se denota mediante *M,* entonces una entrada *M [{s, o}]* indica con precisión que operaciones pueden solicitar el sujeto *s* para que se realicen en el objeto *o.* Se podría decir que en otros términos, siempre que un sujeto *s* solicite la invocación del método *m* del objeto *o,* el monitor de referencia deberá verificar si *m* aparece en *M [s, o].* Si *m* no aparece en *M[s, o],* la invocación falla.

Si consideramos que un sistema fácilmente puede necesitar dar soporte a miles de usuarios y que millones de objetos requieren protección, implementar una matriz de control de acceso como matriz verdadera no es la forma correcta de proceder. Muchas entradas de la matriz estarán vacías: en general, se podría decir que un solo sujeto tendrá acceso relativamente pocos bajos. Por consiguiente, se siguen otras formas más eficientes para implementar una matriz de control de acceso.

Uno de los métodos que podríamos ver mas ampliamente aplicado es hacer que cada objeto mantenga una lista de derechos de acceso de sujetos que desean acceder a el. En esencia, esto significa que la matriz está distribuida en columnas a través de todos los objetos y que las entradas vacías se eliminan. Este tipo de implementación conduce a lo que se llama ACL (Access Control List; lista de control de acceso). Se supone que cada objeto tiene su propia ACL asociada.

Otro método es distribuir la matriz en filas dando a cada sujeto una lista de las capacidades que tiene para cada objeto. En otras palabras, una capacidad corresponde a una entrada en la matriz de control de acceso. No tener una capacidad para un objeto específico significa que el sujeto no tiene derechos de acceso para dicho objeto.

La diferencia entre cómo se utilizan las ACL y las capacidades para proteger el acceso a un objeto Utilizando las ACL, cuando un cliente envíe una petición a un servidor, el monitor de referencia del servidor verificará si conoce al cliente y si éste tiene permiso de realizar la operación solicitada.



Sin embargo, cuando utiliza capacidades, un cliente simplemente envía su solicitud (petición) al servidor. A éste no le interesa si conoce al cliente; la capacidad declara lo suficiente. Por tanto, el servidor necesita comprobar únicamente si la capacidad es válida y si la operación solicitada aparece en la capacidad. Este método de proteger objetos por medio de capacidades.

**Dominios de protección**

Las ACL y las capacidades ayudan a implementar con eficiencia una matriz de control de acceso al ignorar todas las entradas vacías. No obstante, una ACL o una lista de capacidades pueden llegar a ser lo bastante grandes si no se toman medidas adicionales. Una forma general de reducir las ACL, es utilizar dominios de protección. Formalmente, un dominio de protección es un conjunto de pares (objeto, derechos de acceso). Para un objeto dado, cada par especifica con exactitud qué operaciones pueden realizarse (Saltzer y Schroeder, 1975). Las peticiones para realizar una operación siempre se emiten dentro de un dominio. Por consiguiente, siempre que un sujeto solicita realizar una operación en un objeto, el monitor de referencia busca primero el dominio de protección asociado con dicha petición. Acto seguido, dado el dominio, el monitor de referencia puede verificar si la petición tiene permiso de ser realizada. Existen diferentes usos de los dominios de protección.

La ventaja de tener grupos jerárquicos es que la administración de la membresía para un grupo es relativamente fácil y que se pueden construir grupos muy grandes con eficiencia. Una desventaja evidente es que la búsqueda de un miembro puede resultar bastante costosa si la base de datos de membresías es distribuida.

En lugar de dejar que el monitor de referencia haga todo el trabajo, una alternativa es permitir que cada sujeto porte un certificado que contenga una lista de los grupos a los que pertenece. Para garantizar que el certificado es genuino y que no ha sido manipulado, deberá estar protegido mediante, por ejemplo, una firma digital. Los certificados se consideran comparables a las capacidades.

En relación con el hecho de contar con dominios de protección, también es posible implementar los dominios de protección como roles. En un control de acceso basado en roles, un usuario siempre entra al sistema con un rol específico, el que a menudo está asociado con la función que el usuario cumple dentro de la organización (Sandhu y cols., 1996). Un usuario puede tener varias funciones.

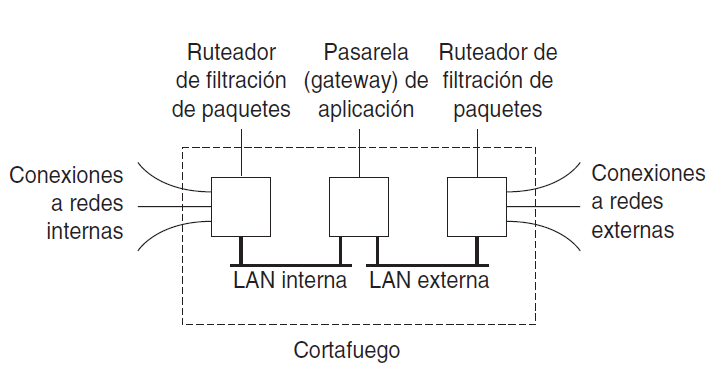
Además de utilizar dominios de protección, la eficiencia puede mejorarse aún más si los objetos se agrupan (jerárquicamente) con base en las operaciones que proporcionan. Por ejemplo, en lugar de considerar objetos individuales, éstos se agrupan de acuerdo con las interfaces que proporcionan, tal vez mediante una clasificación por subtipos [también conocida como herencia de interfaz, vea Gamma y cols., (1994)] para lograr cierta jerarquía. En este caso, cuando un sujeto solicita realizar una operación en un objeto, el monitor de referencia busca a qué interfaz pertenece la operación. Acto seguido verifica si el sujeto tiene permiso de invocar la operación que pertenece a dicha interfaz en lugar de si tiene permiso de invocar la operación para el objeto específico.

También es posible combinar los dominios de protección y agrupar los objetos. Utilizando ambas técnicas, junto con estructuras de datos específicas y operaciones restringidas en objetos, Gladney (1997) describe cómo implementar las ACL para conjuntos muy grandes de objetos utilizados en bibliotecas digitales.

**Cortafuegos (Firewall)**

Hasta ahora, se ha demostrado cómo se puede establecer la protección mediante técnicas criptográficas, combinadas con alguna implementación de una matriz de control de acceso. Estos métodos funcionan muy bien en tanto que todas las partes en comunicación actúen de acuerdo con el mismo conjunto de reglas. Tales reglas pueden ser aplicadas cuando se desarrolle un sistema distribuido autónomo aislado del resto del mundo. Sin embargo, las cosas se complican más cuando se permite el acceso a extraños a los recursos controlados por un sistema distribuido. Ejemplos de tales accesos incluyen envío de correo, descarga de archivos, carga de formas fiscales, etcétera.

Para proteger los recursos en estas circunstancias, se requiere un método diferente. En la práctica, sucede que el acceso externo a cualquier parte de un sistema distribuido es controlado por una clase especial de monitor de referencia conocido como firewall (Cheswick y Bellovin, 2000, y Zwicky y cols., 2000). En esencia, un cortafuego desconecta cualquier parte de un sistema distribuido del mundo externo. se canalizan a través de una computadora especial y se inspeccionan antes de permitirles el paso. El tráfico no autorizado se desecha y no se le permite continuar. Un tema importante es que el cortafuego propiamente dicho debe estar fuertemente protegido contra cualquier clase de amenaza de seguridad: nunca deberá fallar.



Los cortafuegos, en esencia, vienen en dos sabores diferentes que a menudo se combinan. Un tipo importante de cortafuego es una compuerta de filtración de paquetes. Este tipo de cortafuego opera como ruteador y toma decisiones en cuanto a dejar pasar o no un paquete de red con base en la dirección de origen o destino contenida en el encabezado del paquete. Típicamente, la compuerta de filtración de paquetes mostrada en la LAN externa protege contra paquetes entrantes, en tanto que la LAN interna filtra los salientes.

Por ejemplo, para proteger un servidor web interno contra peticiones de hosts que no están en la red interna, una compuerta de filtración de paquetes podría decidir rechazar todos los paquetes entrantes dirigidos al servidor web.

Más sutil es la situación en que la red de una compañía consta de múltiples redes de área local conectadas, por ejemplo, mediante una red SMDS como se vio antes. Cada LAN puede estar protegida por medio de una compuerta de filtración de paquetes, configurada para que deje pasar el tráfico entrante sólo si proviene de un host localizado en una de las otras LAN. De este modo, se puede establecer una red virtual privada.

El otro tipo de cortafuego es una compuerta a nivel de aplicación. Por contraste con una compuerta de filtración de paquetes, la cual inspecciona sólo el encabezado de paquetes de red, este tipo de cortafuego en realidad inspecciona el contenido de un mensaje entrante o saliente. Un ejemplo típico es una compuerta de correo que desecha el correo entrante o saliente que excede cierto tamaño. Existen compuertas de correo más complejas que son, por ejemplo, capaces de filtrar correo electrónico basura.

Otro ejemplo de una compuerta a nivel de aplicación es una implementación que permite el acceso a un servidor de biblioteca externo, pero que proporciona sólo extractos de documentos. Si un usuario externo desea más, se inicia un protocolo de pago electrónico. Los usuarios localizados en el interior del cortafuego tienen acceso directo al servicio de biblioteca.

Una clase especial de compuerta a nivel de aplicación es la conocida como compuerta proxy. Este tipo de cortafuego funciona como etapa frontal ante una clase específica de aplicación, y garantiza que pasarán sólo aquellos mensajes que satisfagan ciertos criterios. Consideremos, por ejemplo, navegar por la web. Tal como analizamos en la siguiente sección, muchas páginas web contienen scripts o applets que deben ser ejecutados en el buscador de un usuario. Para evitar que semejante código se descargue hacia la LAN interna, todo el tráfico web podría ser encauzado a través de una compuerta proxy. La compuerta acepta peticiones http regulares, o desde adentro o de afuera del cortafuego. En otros términos, aparece ante sus usuarios como un servidor web normal. No obstante, filtra todo el tráfico saliente o entrante, ya sea desechando ciertas peticiones y páginas o modificándolas cuando contienen un código ejecutable

## **5.4 Administración de la seguridad**

Primero, se tiene que considerar la administración general de claves criptográficas, y en especial la forma en que se distribuyen las claves públicas. Como resulta ser, los certificados desempeñan un rol importante aquí.

En segundo lugar, abordamos el problema de administrar con seguridad un grupo de servidores concentrándonos en el problema de agregar un nuevo miembro confiable para los miembros actuales. Claramente, de cara a los servicios distribuidos y replicados, es importante que no se comprometa la seguridad admitiendo un proceso malicioso para un grupo.

En tercer lugar, consideramos la administración de la autorización atendiendo las capacidades y lo que se conoce como certificados de atributo. Un tema importante en sistemas distribuidos, con respecto a la administración de la autorización, es que un proceso puede delegar algunos o todos sus derechos de acceso a otro proceso. La delegación de derechos en forma segura tiene sus propias sutilezas, tal como también se analiza en esta sección.

**Administración de claves**

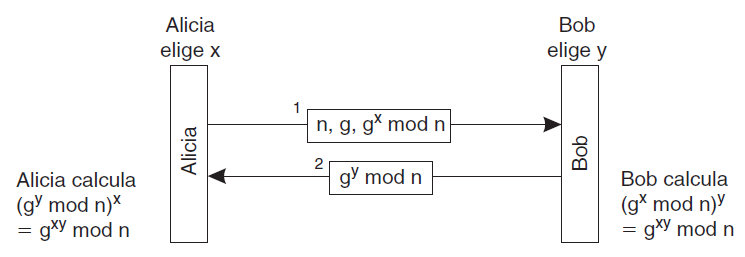
Hemos descrito varios protocolos criptográficos en los cuales se asumió (implícitamente) que estaban disponibles varias claves. Por ejemplo, en el caso de criptosistemas de clave pública, se supuso que el remitente de un mensaje tenía a su disposición la clave del destinatario, de tal suerte que podía cifrar el mensaje para asegurar su confidencialidad. Asimismo, en el caso de autenticación por medio de un centro de distribución de claves (KDC), supusimos que cada una de las partes ya compartía una clave secreta con el KDC.

Sin embargo, el establecimiento y la distribución de claves no es un asunto trivial. Por ejemplo, la distribución de claves secretas por medio de un canal inseguro es impensable y en muchos casos se tiene que recurrir a métodos que no están disponibles. También, se requieren mecanismos para revocar las claves, es decir, para impedir que una clave sea utilizada una vez que ha sido comprometida o invalidada. Por ejemplo, la revocación es necesaria cuando una clave ha sido comprometida.

**Establecimiento de claves**

Se puede utilizar un esquema similar para generar y distribuir una clave de sesión cuando Alicia y Bob ya comparten una clave secreta. Sin embargo, ambos métodos requieren que las partes que se están comunicando dispongan de una forma de establecer un canal seguro. En otros términos, ya debe haber ocurrido alguna forma de establecimiento y distribución de claves. El mismo argumento es aplicable cuando se establece una clave secreta compartida por medio de una tercera parte confiable, tal como un KDC.

Un esquema elegante y ampliamente utilizado para establecer una clave compartida a través de un canal inseguro es el intercambio de claves de Diffie-Hellman (Diffie y Hellman, 1976). El protocolo funciona como sigue. Supongamos que Alicia y Bob desean establecer una clave secreta compartida. El primer requerimiento es que se pongan de acuerdo en dos números grandes n y g sujetos a varias propiedades matemáticas (las cuales no se analizan aquí). Tanto n como g pueden hacerse públicos; no es necesario ocultarlos de los extraños. Alicia elige un número aleatorio grande, por ejemplo x, el cual mantiene secreto. Asimismo, Bob selecciona su propio número grande secreto, digamos, y. En este momento existe suficiente información para construir una clave secreta.



Alicia comienza enviando g^x mod n a Bob, junto con n y g. Es importante señalar que esta información puede ser enviada como texto común, ya que es virtualmente imposible calcular x dado g^x mod n. Cuando Bob recibe el mensaje, posteriormente calcula (g^x mod n)y que matemáticamente es igual a g^xy mod n. Además, envía g^y mod n a Alicia, que luego puede calcular (g^y mod n)^x = g^xy mod n. Por consiguiente, tanto Alicia como Bob, y sólo ellos, tendrán ahora en su poder la clave secreta compartida g^xy mod n. Observemos que ninguno de ellos tiene que revelar su número privado (x y y, respectivamente) al otro.

Diffie-Hellman puede ser considerado como un criptosistema de clave pública. En el caso de Alicia, x es su clave privada, en tanto que g^x mod n es su clave pública. Como se verá a continuación, la distribución segura de la clave pública es esencial para que Diffie-Hellman funcione en la práctica.

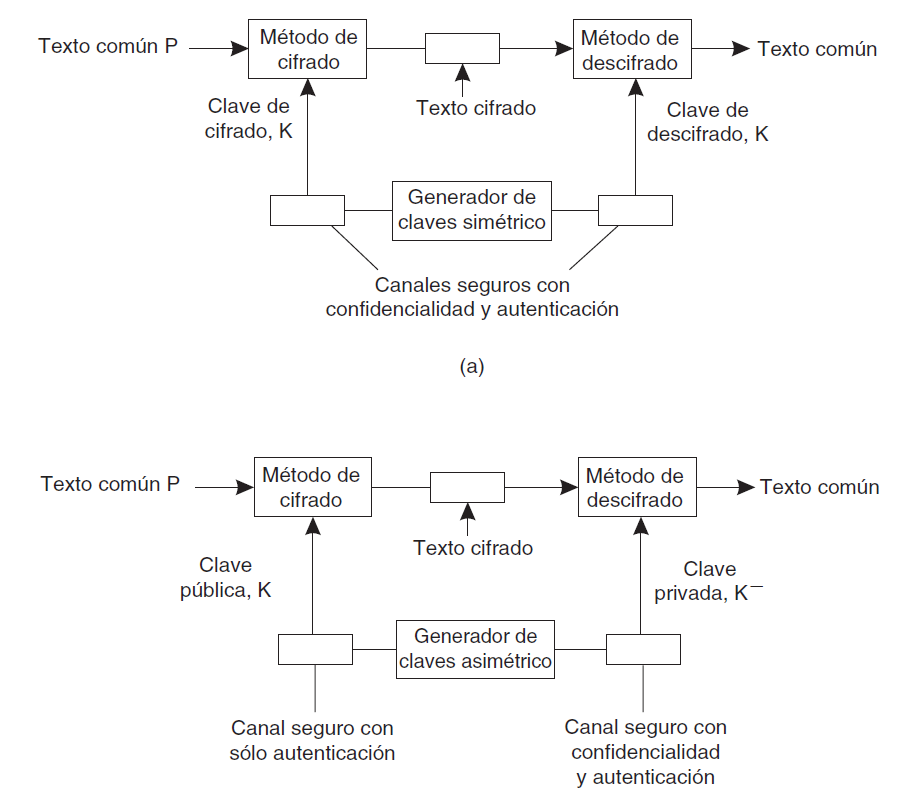
**Distribución de claves**

Una de las partes más difíciles de realizar en la administración de claves es la distribución de las claves iniciales. En un criptosistema simétrico, la clave secreta compartida inicial debe ser comunicada a través de un canal seguro que proporcione autenticación y confidencialidad.

En el caso de un criptosistema de clave pública, la clave pública tiene que ser distribuida en tal forma que los destinatarios estén seguros de que sí está pareada con una clave secreta.

Aunque la clave pública puede ser enviada como texto común, es necesario que el canal a través del cual es enviada pueda proporcionar autenticación. La clave privada, desde luego, tiene que ser enviada a través de un canal seguro que proporcione autenticación y confidencialidad. Cuando se trata de distribución de claves, la distribución autenticada de claves públicas es tal vez lo más interesante. En la práctica, la distribución de una clave pública ocurre por medio de certificados de clave pública. Tales certificados consisten en una clave pública junto con una cadena que identifica la entidad con la cual dicha clave está asociada. La entidad podría ser un usuario, aunque también un servidor o algún dispositivo especial. Una autoridad certificadora firma la clave pública y el identificador, y la firma también se incluye en el certificado. (La identidad de la autoridad certificadora, naturalmente, no forma parte de la autoridad certificadora.) La firma se realiza por medio de una clave privada KCA que pertenece a la autoridad certificadora. Se supone que la clave pública correspondiente K CA es conocida. Por ejemplo, las claves públicas de varias autoridades certificadoras vienen incluidas en la mayoría de los navegadores de la web y se envían junto con los binarios.

El uso de un certificado de clave pública funciona como sigue. Supongamos que un cliente desea asegurarse de que la clave pública encontrada en el certificado pertenece realmente a la entidad identificadora. El cliente utiliza entonces la clave pública de la autoridad certificadora asociada para verificar la firma del certificado. Si la firma que aparece en el certificado concuerda con el par (clave pública, identificador), el cliente acepta que la clave pública pertenece a la entidad identificada.



Es importante señalar que cuando se acepta que el certificado está en orden, el cliente en realidad confía en que no ha sido falsificado. En particular, el cliente debe asumir que la clave pública K^CA sí pertenece a la autoridad certificadora asociada. En caso de duda, deberá ser posible verificar la validez de K^CA mediante otro certificado que provenga de una autoridad certificadora diferente, quizá más confiable.

Tales modelos confiables jerárquicos, donde todos deben confiar en la autoridad certificadora de más alto nivel, no son comunes. Por ejemplo, el PEM (Privacy Enhanced Mail; correo de privacidad mejorada) utiliza un modelo confiable de tres niveles en el que las autoridades certificadoras de más bajo nivel pueden ser autenticadas por PCA (Policy Certification Authorities; autoridades certificadoras de políticas), las que a su vez pueden ser autenticadas por la IPRA (Internet Policy Registration Authority; autoridad de registro de políticas en internet). Si un usuario no confía en la IPRA, o piensa que no es seguro hablar con ella, no es de esperarse que algún día confíe en que los mensajes de correo electrónico confiables sean enviados de una forma segura cuando se utiliza el PEM. Más información sobre este modelo puede consultarse en Kent (1993). En Menezes y colaboradores (1996) se analizan otros modelos confiables.

**Duración de los certificados**

En primer término, consideremos la situación en que una autoridad certificadora entrega certificados de por vida. En esencia, lo que el certificado manifiesta es que la clave pública siempre será válida para la identidad identificada por el certificado. Claramente, esta manifestación no es deseable. Si alguna vez se compromete la clave privada de la entidad identificada, un cliente poco sospechoso jamás deberá ser habilitado para utilizar la clave pública (mucho menos los clientes maliciosos). En ese caso, se requiere de un mecanismo útil para revocar el certificado haciendo que se sepa que ya no es válido. Existen varias formas de revocar un certificado. Un método común es con una CRL (Certificate Revocation List; lista de revocación de certificado) publicada con regularidad por la autoridad certificadora. Siempre que un cliente verifique un certificado, tendrá que consultar la CRL para ver si ha sido o no revocado. Esto significa que el cliente deberá ponerse en contacto con la autoridad certificadora, por lo menos, cada vez que se publique una CRL nueva. Observemos que la CRL se publica a diario, también se requiere un día para revocar un certificado. Mientras tanto, un certificado comprometido puede ser utilizado en forma dolosa hasta ser publicado en la siguiente CRL.

En consecuencia, el tiempo entre la publicación de las CRL no debe ser demasiado largo. Además, la obtención de una CRL presupone algunos gastos indirectos. Un método alternativo es restringir la duración de cada certificado. La validez de un certificado expira automáticamente después de cierto tiempo. Si por cualquier razón el certificado debe revocarse antes de que expire, la autoridad certificadora aún lo puede publicar en una CRL. Sin embargo, este método seguirá obligando a que los clientes consulten la CRL más reciente siempre que verifiquen un certificado. En otros términos, tendrán que ponerse en contacto con la autoridad certificadora o con una base de datos confiable que contenga la CRL más reciente.

Un caso final extremo es reducir la duración de un certificado a casi cero. En realidad, esto significa que no se utilizarán certificados; en su lugar, un cliente siempre tendrá que ponerse en contacto con la autoridad certificadora para verificar la validez de una clave pública. En consecuencia, la autoridad certificadora debe estar continuamente en línea. En la práctica, los certificados se entregan con duraciones restringidas. En el caso de aplicaciones de internet, el tiempo de expiración a menudo es de más de un año (Stein, 1998). Tal método requiere que las CRL se publiquen con regularidad, pero que también sean inspeccionadas cuando se verifiquen certificados. La práctica indica que las aplicaciones de cliente rara vez consultan las CRL, y simplemente asumen que un certificado es válido hasta que expira. A este respecto, cuando se trata de seguridad en internet en la práctica, sigue habiendo mucho por mejorar, desafortunadamente.

# **Conclusiones**

# **Herramientas y recursos**

* Imágenes de Google
* Power Point para crear la exposición
* Youtube
* El documento de nuestra investigación del tema
* Scribbr para el APA
* Equipo de cómputo.

# **Bibliografía**

iVoox. (2021, 28 octubre). ¿Qué son los canales seguros? - Canales Seguros - Podcast en. <https://www.ivoox.com/que-son-canales-seguros-audios-mp3_rf_77475303_1.html>