UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

SISTEMA CENTRALIZADO DE AUTENTICACIÓN DE USUARIOS PARA LA SUBDIRECCIÓN DE SEGURIDAD DE LA INFORMACIÓN UNAM-CERT

TESIS PROFESIONAL

QUE PARA OBTENER EL TÍTULO DE:

INGENIERO EN COMPUTACIÓN

PRESENTA:

CÉSAR IVÁN LOZANO AGUILAR

DIRECTOR DE TESIS:

ING. JESÚS MAURICIO ANDRADE GUZMÁN

Agradecimientos

Primero que nada y sobre todas las cosas a mi Señor Jesucristo por concederme el privilegio de vivir y permitirme culminar la carrera de ingeniería en computación en la Universidad Nacional Autónoma de México; por darme

A mis padres

Que siempre estuvieron apoyándome moral y económicamente; por darme la oportunidad de conocer

A Mauricio

Que aceptó ser mi director de tesis y orientarme en el desarrollo de este proyecto; por permitirme conocer

A mis amigos

Que con su compañía he pasado muy gratos momentos.

Y a todos aquellos que de manera directa o indirecta mi Señor Jesucristo ha puesto en mi camino para forjarme en esta vida.

Gracias Señor Jesucristo por todas las cosas que me has dado

"Acuérdate de tu Creador en los días de tu juventud, antes que vengan los días malos, y lleguen los años, de los cuales digas: No tengo en ellos contentamiento"

Eclesiastés 12:1

Ciertamente vengo en breve. Así sea.

¡Ven Señor Jesús!

Apocalipsis 22:20

CONTENIDO

INTRODUCCIÓN

Un proceso general de autenticación comienza al identificar algo o alguien a través de un procedimiento determinado. Por ejemplo, para distinguir un objeto se pueden referir características propias como el color, la forma, el tamaño, el peso, de qué material está hecho, entre otras, y así asegurar de qué se trata.

Se considera el caso de un animal, en el que existen otros parámetros para su identificación, tales como la especie, el color de ojos, el ruido que emite, el color de pelo o plumaje, el tamaño y una amplia variedad de características propias que ayudan a determinar de qué animal se trata.

Ahora, si se trata de seres humanos se utilizarían parámetros como el género, el color de piel, la estatura, el color y la forma del cabello, el tono de voz, el color de ojos, etcétera.

Siempre que se desee identificar, es necesario utilizar un mecanismo que avale dicha identidad, para ello, y de forma natural, los seres humanos tienen cinco sentidos, pese a su complejidad este proceso resulta útil para distintas finalidades.

Cuando se tiene contacto con un objeto por primera vez, inicialmente se efectúa un reconocimiento, teniendo en cuenta sus características intrínsecas para posteriormente volver a reconocer, no a identificar, ya que se ha llevado a cabo previamente dicho proceso.

Ahora, supóngase que se requiere una identificación estricta y objetiva, para ello se utilizará el siguiente ejemplo:

Imagínese que un estudiante va a realizar un examen extraordinario, y para no ser suplantado, le exigen verificar la veracidad de su identidad; para ello, le solicitan presentarse con una identificación oficial, que incluya nombre completo y fotografía. El profesor encargado de aplicar el examen se convierte en un agente evaluador, y tomará en cuenta parámetros en relación con el estudiante, para determinar si es o no la persona indicada.

De acuerdo al significado del diccionario, autenticar es “Legalizar o autorizar una firma o un documento”[1], otro de ellos es “Acreditar, dar fe de la verdad de un hecho”[2], de manera que para este caso debe existir una autoridad que dé fe a que en realidad se está llevando a cabo un procedimiento, por lo que al validar y aprobar la veracidad de algo o alguien, se está

ejecutando un procedimiento de autenticación. En el ejemplo antes mencionado, el profesor se convierte en un agente evaluador o validador.

Adaptar un proceso de autenticación en un sistema computacional sea éste un sistema web, un software de base o de aplicación, no consistiría en identificar a una persona o sistema, sino asegurar que tal persona es quien dice ser realmente. En el caso de un sistema, es validar el origen de dónde proviene, establecer que la ubicación desde donde proviene sea efectivamente esa ubicación.

La autenticación, de manera general, no está limitada solamente al área de sistemas o computación, ya que un policía, un detector de voz, un biométrico, etcétera, pueden ser agentes de validación para aprobar la identidad de las personas.

Generalmente un sistema de información tiende a estar protegido, y la forma más común es mediante un nombre de usuario y una contraseña, los cuales nos identifican ante tal sistema. De esta manera, el sistema identifica quién está tratando de acceder, y puede aprobar o denegar el acceso.

El proceso de autenticación a tratar en esta tesis, se refiere a la autenticación de usuarios sobre sistemas computacionales, específicamente sobre los sistemas web con los que cuenta la Subdirección de Seguridad de la Información (SSI) y de los parámetros que se involucran en este procedimiento.

Actualmente la SSI cuenta con varios servicios de información, cada uno tiene su propio mecanismo de autenticación, de tal modo que el personal de la subdirección que tiene acceso a estos servicios, tiene una contraseña diferente, obligándolo a utilizar contraseñas no muy robustas o usar la misma contraseña por mucho tiempo, ocasionando que el usuario pueda ser suplantado en el uso de estos servicios de información.

Otro de los inconvenientes de utilizar contraseñas distintas es que el personal llega a olvidarlas, ya que algunos servicios son utilizados esporádicamente, haciendo que sus administradores con frecuencia tengan que restablecer contraseñas olvidadas. Esto ocasiona que el personal genere nuevas contraseñas que resultan poco robustas para no olvidarlas, o bien robustas con el riesgo de ser olvidadas nuevamente.

El objetivo principal es autenticar a los usuarios sobre los sistemas web de la subdirección mediante un módulo programado, que en conjunto con un agente de seguimiento de sesión de usuario, centralice este procedimiento sobre los demás sistemas web, apoyándose

a su vez de un servicio de directorio (OpenLDAP). Y para ello el presente trabajo se compone de 4 capítulos.

El capítulo 1 describe los conceptos básicos relacionados con la autenticación de usuarios, autorización, control de acceso y autenticación multi-factor en los sistemas computacionales, haciendo algunas analogías y ejemplos básicos cotidianos. Además de abordar algunas tecnologías utilizadas en la autenticación de usuarios, las cuales se involucran, de cierta forma, el uso del módulo de autenticación centralizada.

En el capítulo 2 se mencionan los mecanismos y formas de autenticación desde una autenticación simple mediante el servidor web, hasta autenticar usuarios mediante certificados y un formulario web. A su vez se tratan conceptos y términos involucrados en una autenticación de usuarios en sistemas web, para mejorar este tipo de autenticación e ilustrándolos en un ejemplo práctico. El último subtema abre un preámbulo a un tipo de autenticación ya muy común en la Web: los tokens de acceso.

El capítulo 3 detalla la infraestructura con la que cuenta la SSI (servidor web, servidor de directorio activo y base de datos). Se tratan conceptos fundamentales sobre el servicio de directorio activo y LDAP, su estructura y los elementos que lo componen. También puntualiza cómo utilizar LDAP en tanto sistema de autenticación; la integración de LDAP con el lenguaje de programación PHP y compatibilidad con otros sistemas. Se describe el diseño y el desarrollo del módulo de autenticación, su implementación en un sistema de pruebas y la preparación de un escenario de autenticación centralizada de usuarios sobre el cual va a operar el módulo de autenticación centralizada.

El capítulo 4 corresponde a la fase de pruebas y exposición de los resultados obtenidos, describiendo el proceso de inicio de sesión por primera vez en el sistema de autenticación centralizada, cómo se rige el control de acceso sobre el sistema, las ventajas de una mejor administración de usuarios y la implementación de una buena política de creación de contraseñas, al ingresar a varios sistemas utilizando una única contraseña.

Finalmente, se expresan las conclusiones conforme a los resultados obtenidos a la implementación del módulo de autenticación centralizada en un ambiente de producción, las ventajas y desventajas, y cómo hacerles frente. Por último, se describe el trabajo a futuro para la mejora del módulo de autenticación, y una posible adaptación para nuevos proyectos que requieran una autenticación centralizada de usuarios.

CAPÍTULO 1

AUTENTICACIÓN Y ELEMENTOS QUE LA COMPONEN

1.1 Autenticación en sistemas computacionales

Como pieza fundamental en el uso de aplicaciones remotas, se tiene el concepto de autenticación. La autenticación en los sistemas computacionales es el procedimiento mediante el cual se tiene por auténtica la identidad de un usuario que intenta acceder a un sistema determinado. Comúnmente, para poder acceder requiere de un nombre de usuario y una contraseña, es decir, otorgar el acceso a alguien que reclama ser quien dice ser.

Generalmente, la autenticación se ve aplicada solamente a usuarios de algún sistema, pero en ocasiones un usuario puede ser otro sistema, en otras palabras, es un sistema que intenta acceder a otro. En este caso, la autenticación se define como la verificación verídica de la proveniencia de ese sistema. Un ejemplo claro puede ser un servidor web que pretende acceder al servidor de bases de datos para realizar las consultas respectivas. En la figura 1.1 se ilustra este concepto.

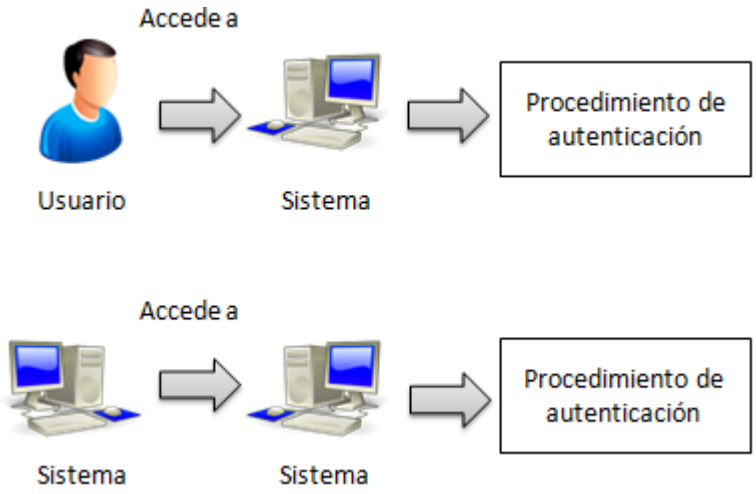

Figura 1.1 Acceso de un usuario y de un sistema.

1.2 Autorización en sistemas computacionales

La autenticación y la autorización van de la mano, aunque en primera instancia sucede la autenticación y posteriormente la autorización. Esta última se hace conocer también como el

perfil del usuario[3] (ver figura 1.2); el cual indica el nivel o grado de acceso que tiene el usuario o sistema interventor.

No todos los usuarios tienen el mismo grado de acceso, aquellos que se dedican a dar mantenimiento o que juegan un papel de administrador sobre el sistema en cuestión, normalmente tienen un perfil establecido, al cual se le conoce como cuenta de administrador o *súper usuario.*

Por otro lado, existe el usuario genérico, el cual no posee ciertas opciones o privilegios dentro del sistema a pesar de estar autenticado. Por ejemplo, en un sistema en el que se gestionan incidentes de seguridad informática, pudieran existir dos tipos de roles, sean estos usuario reportador y usuario administrador. El usuario reportador en primera instancia, puede tener la característica de registrar incidentes, además de poder imprimir reportes de los incidentes generados en un periodo determinado. El usuario administrador, aparte de poder registrar incidentes e imprimir reportes, puede gestionar a los usuarios que utilizan el sistema dando altas y bajas, e inclusive hacer un cambio de roles sobre algún usuario.

De esta forma, se observa que no todos los usuarios pueden realizar las mismas tareas, dado que cada rol tiene un nivel de autorización determinado para llevar a cabo acciones específicas dentro del sistema.

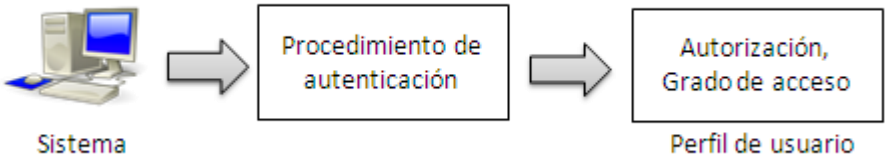

Figura 1.2 Acceso al perfil de usuario.

1.3 Control de acceso en los sistemas de cómputo

Una vez que un usuario o sistema han pasado por la autenticación y la autorización, el control de acceso rige la entrada de los usuarios al sistema, implícitamente tiene que ver con el procedimiento de autenticación, ya que se trata de la interfaz que interactuará con el usuario para poder autenticarse, pero también consiste en un mecanismo de seguridad empleado para regir la interacción entre los usuarios y las tareas que están realizando sobre el sistema en el que fueron autenticados y autorizados a utilizar, de igual forma verifica que los privilegios concedidos a los usuarios se lleven a cabo correctamente.

Una vez que un usuario se ha autenticado, registrar el acceso de ingreso al sistema y los privilegios con los que cuenta no son suficientes.[4] Este mecanismo debe ser capaz de controlar

lo que los usuarios hacen mientras están utilizando el sistema al cual fueron autorizados, en este punto es donde resalta la importancia del control de acceso.

Para un administrador de red, el control de acceso permite supervisar la hora en la que los usuarios se conectaron a un sistema, los recursos utilizados por él mientras estaba conectado y la información o los datos manipulados. En la figura 1.3, se puede observar la intervención del control de acceso sobre las etapas de autenticación y autorización de usuarios.

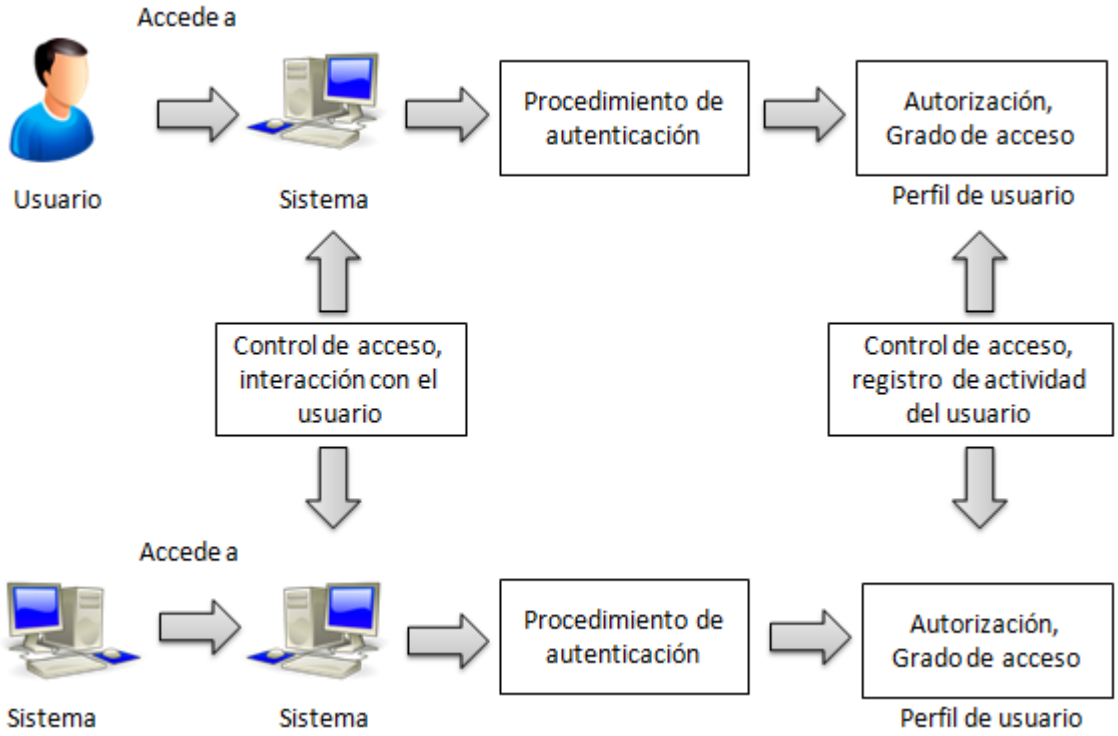

Figura 1.3 Interacción del control de acceso, autenticación y autorización.

1.4 Autenticación multi-factor

Los mecanismos de autenticación permiten establecer cierto nivel de seguridad, al momento de solicitar la autenticidad de los usuarios en los sistemas. Estos niveles de seguridad están en función del grado de fortaleza dentro de las siguientes tres categorías:

- Basado en "*lo que se sabe*"
- Basado en "*lo que se tiene*"
- Basado en "*lo que se es*"

1.4.1 Mecanismo basado en "*lo que se sabe*"

Este mecanismo se considera como el de menor grado de fortaleza, ya que se vincula directamente a término custodia; algo que se conoce y que puede ser comunicado a alguien

más por descuido, tal es el caso de los nombres de usuario el cual es una palabra corta o seudónimo acompañado por una contraseña conformada por una combinación de caracteres alfanuméricos y símbolos especiales en el mejor de los casos, de tal modo que el usuario no la olvide. Ambos elementos describen este mecanismo de autenticación, el nombre de usuario y la contraseña son custodiados por el usuario que ha de autenticarse en un sistema, pero son susceptibles al olvido. El método no es el más seguro, pero sí el más utilizado en la mayoría de los sistemas que requieren autenticación por parte de los usuarios que los utilizan.[5]

Como ejemplo se pueden mencionar las credenciales de acceso a una red social (Facebook, Twitter, etc.), en los que para acceder se necesite de una cuenta de correo (usada como cuenta de acceso) y una contraseña.

1.4.2 Mecanismo basado en "*lo que se tiene*"

El mecanismo basado en "lo que tengo" corresponde a parte de lo que un usuario posee, siendo esto un elemento físico que puede corroborar su identidad para el acceso a determinados sistemas o servicios. Este tipo de elementos o dispositivos puede albergar claves más sofisticadas y complejas, haciendo más difícil la labor de descubrirlas, incrementando un poco el nivel de fortaleza que el basado sobre algo que conoce el usuario (nombre de usuario y contraseña). Ahora, el inconveniente es que estos dispositivos pueden ser extraídos al ser elementos físicos.[6]

Para ejemplificar este mecanismo, supóngase que se desea realizar una transferencia bancaria, para tal efecto, además de requerir un nombre de usuario y una contraseña es necesario de manera adicional un elemento físico, para este caso un token bancario, que se trata de un dispositivo electrónico que genera constantemente números aleatorios válidos durante un tiempo determinado. Sin este elemento electrónico sería imposible realizar una transferencia bancaria, aunque se tengan credenciales de acceso válidas.

1.4.3 Mecanismo basado en "*lo que se es*"

Este último mecanismo posee el mayor grado de fortaleza, ya que consiste en algo que describe al usuario, algo que lo caracteriza frente a los demás, que define lo que es o cómo es. Estas características de los usuarios van íntimamente ligadas con técnicas biométricas informáticas, analizándolas con patrones conocidos de antemano para verificar la autenticidad de los usuarios. En este mecanismo no existe directamente inconveniente, pues las características de los usuarios no es algo que pueda robarse; ni tampoco olvidar de manera tan sencilla.[7] Y para el caso en el que un usuario sea un sistema se puede considerar su dirección física (MAC) para la autenticación.

Para incrementar la fortaleza del proceso de autenticación y tener la completa certeza de que los usuarios son quienes dicen ser, el método de autenticación que sea utilizado puede

complementarse con alguno otro, es decir, llevar a cabo una autenticación doble o multi-factor en función del número de métodos de autenticación combinados. La figura 1.4 muestra ejemplos de los elementos involucrados en una autenticación de múltiples factores.

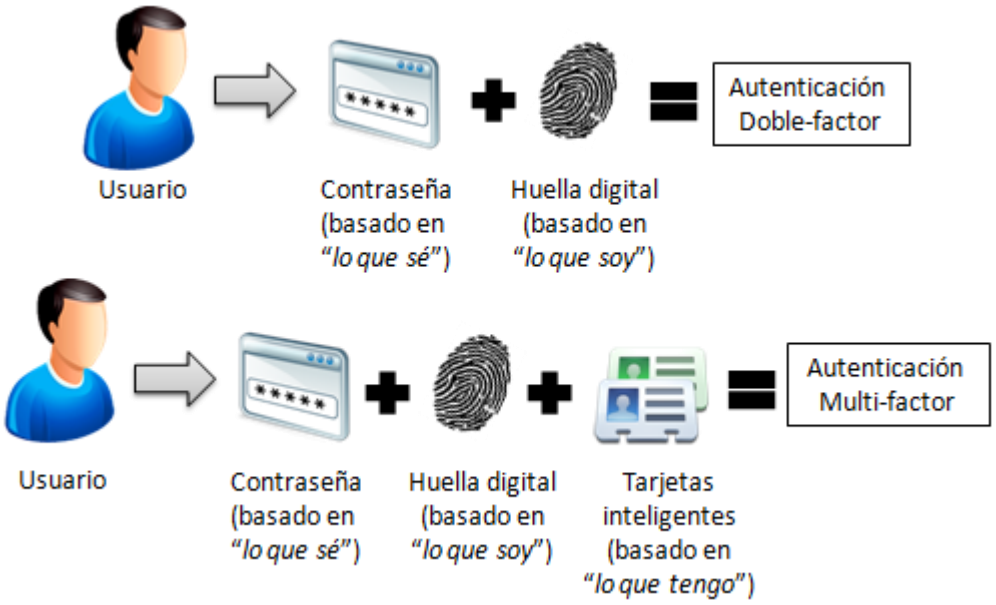

Figura 1.4 Elementos involucrados para una autenticación multi-factor

1.5 Protocolos de autenticación

La comunicación humana se da a través de diferentes maneras, siendo éstas oral, escrita, mímica y de cada una de ellas se desprenden sus diferentes tipos. Cada manera de comunicación posee un procedimiento y un conjunto de reglas para que la comunicación se dé.

¿Qué pasa con la comunicación entre equipos de cómputo? por sí misma no existe, pero mediante la intervención humana es posible esto. Para ello, se han desarrollado conjuntos de reglas y/o estándares, que permiten establecer una comunicación entre los equipos de cómputo a través de la red.

Un protocolo es un conjunto de reglas estandarizadas que se deben seguir, para que la comunicación entre computadoras se lleve a cabo de manera formal y efectiva. Estas primeras reglas, se desarrollaron para la comunicación entre computadoras de un mismo fabricante, propiciando un protocolo monolítico.

Al tener la necesidad de una comunicación más extensa, estas reglas se estandarizan y se clasifican en capas, para lograr una mejor y eficiente comunicación entre cualquier equipo de cómputo.

La autenticación sobre determinados sistemas computacionales es muy importante, y para poder llevarla a cabo se requiere que estos sistemas estén correctamente intercomunicados. Por esa razón, se requiere del empleo de determinados protocolos para establecer la correcta comunicación y realizar el intercambio de mensajes entre los equipos.

El empleo de protocolos no consiste únicamente en establecer una comunicación entre equipos de cómputo, sino que también algunos son empleados para proteger la información que circula a través de su canal de comunicación, mediante métodos de cifrado. Así es posible asegurar que el intercambio de información llegará a su destino y que además estará protegida, evitando con ello el robo de información. Por ejemplo, sobre un sitio web que utiliza usuario y contraseña para acceder a él, ¿cómo percibir si el sistema es seguro para ingresar credenciales de acceso? Detectar este hecho es muy sencillo, ya que hay protocolos que proporcionan características visibles para percatarse de esto.

Se puede mencionar el protocolo HTTPS, que es un protocolo de la capa de aplicación y que va de la mano con el protocolo SSL, en la barra de direcciones del navegador web, muestra como prefijo un "*https*" antes del "*www*", indicando que se trata de la versión segura del protocolo HTTP.

1.5.1 Seguridad de la capa de transporte (TSL) / Capa de conexión segura (SSL) y HTTPS

TSL es un protocolo que permite a las aplicaciones de tipo cliente/servidor comunicarse sobre Internet, de tal manera que la comunicación se mantenga de forma confidencial y la información intercambiada entre ellos se mantenga integra. Se compone de dos mecanismos, el primero consiste en emplear un sistema criptográfico de clave simétrica, en el cual las claves se generan de forma única para cada vez que se realiza este proceso; el segundo se encarga de la autenticación de la información, el intercambio de ésta incluye una comprobación de que estos no han sido alterados y asegurando de este modo la integridad de la misma.[8]

TLS se construye a partir de las especificaciones de SSL 3.0 y son tan semejantes que a veces se le conoce como SSL 3.1.[9]

El protocolo SSL está dividido en 2 capas que son: La capa de enlace y la capa del protocolo registro, la capa de enlace se encarga de la inicialización y sincronización criptográfica entre las entidades que se van a comunicar mientras que la capa de registro provee confidencialidad y autenticación así como protección contra ataques de duplicidad.[10]

Las diferencias entre TLS y SSL no son grandes, pero son suficientes para que ambos no puedan inter-operar. Las diferencias consisten en lo siguiente:

- En los mensajes de solicitud de certificado y verificación del certificado del protocolo de establecimiento, en SSL a menos que el servidor solicite un certificado al usuario para que se autentique, este debe de responder con el certificado o con un mensaje de alerta advirtiendo que no lo posee. Con TLS si el usuario no posee el certificado no responde al servidor de ninguna forma a este requerimiento.
- Para el cálculo de las claves de sesión es ligeramente diferente entre TLS y SSL. En TLS no soporta el algoritmo de cifrado simétrico FORTEZZA caso contrario a SSL que sí lo soporta, TLS busca ser íntegramente público mientras que FORTEZZA es un algoritmo propietario.[11]

TLS utiliza un mecanismo diferente y más seguro en el cálculo del código de autenticación de mensajes (MAC), ya que aplica un algoritmo de hash con claves que produce un valor de comprobación de integridad al igual que MAC, pero con una construcción de funciones hash que hacen que le algoritmo hash sea mucho más difícil de romper.[12]

El protocolo HTTPS es muy simple. Consiste en una combinación del protocolo HTTP con el protocolo SSL/TLS, dicho de otro modo HTTP usa los servicios de SSL para proporcionar el cifrado y la identificación segura del servidor. Comúnmente, se utiliza HTTPS para transacciones de pago en Internet o para transacciones confidenciales en sistemas de información corporativos.[13]

En principio un cliente HTTP debe actuar como cliente TLS/SSL, iniciando una conexión al servidor sobre el puerto apropiado y enviando un "*Client Hello*" para el establecimiento de una conexión SSL/TLS. Una vez que el establecimiento SSL/TLS ha terminado, el cliente entonces puede iniciar la primera petición HTTP.

El primer dato que un servidor HTTP espera recibir del cliente es una petición de línea en la que especifica el tipo de documento solicitado, el método que se aplicara y la versión del protocolo a utilizar. Por otro lado, el primer dato que un servidor TLS/SSL espera recibir es el "*Client Hello*". Por tanto, la práctica común es correr HTTP/SSL sobre un puerto distinto para distinguir el protocolo que se usa. Cuando HTTPS funciona sobre una conexión TCP/IP, el puerto por defecto es el 443.

Algunos sitios web que utilizan HTTPS y requieren autenticación del cliente para acceder al sitio, lo hacen a través del par usuario y contraseña o mediante un certificado digital. El

acceso a un sitio web mediante un certificado digital se verá más a detalle en el tema de autenticación mediante certificados del siguiente capítulo.

1.5.1 Secure Shell (SSH)

Secure Shell por sus siglas en inglés SSH, es un protocolo que permite definir una comunicación segura dentro de una red, incluye la autenticación, cifrado, la integridad de los datos transmitidos[14] y la ejecución de procesos remotos.

El cifrado dentro de SSH consiste en codificar los datos de manera que sean incomprensibles, excepto para el destinatario. Esto protege los datos al pasar sobre una red.

La integridad de la información que se envía mediante *Secure Shell*, garantiza que los datos transmitidos a través de la red lleguen de manera inalterada a su destino, esto quiere decir que si un tercero captura los datos transmitidos y los altera, *Secure Shell* detecta este hecho.

1.5.2 SSH mediante contraseña y llave pública

El método más común de autenticación de usuarios sobre el protocolo SSH, es mediante una contraseña. El cliente solicita al usuario la contraseña, el cliente la envía al servidor y éste la valida mediante los mecanismos instalados en el sistema. Las desventajas de utilizar este método son:

- El usuario debe teclear la contraseña cada vez que requiera conectarse al servidor.
- La contraseña se envía al servidor. Si se accede al servidor mediante un servidor de DNS y éste es manipulado, la contraseña podría enviarse a un servidor distinto al que se está intentando acceder.

El método de autenticación mediante contraseña se ilustra en la figura 1.5.

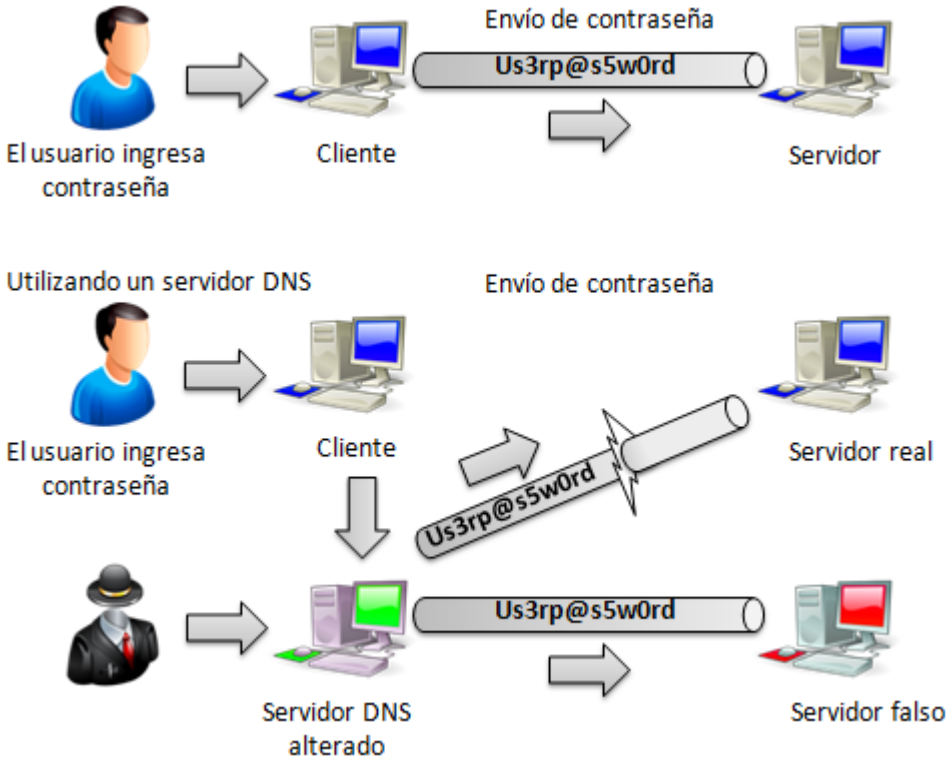

Figura 1.5 Procedimiento de autenticación mediante contraseña

Cuando la autenticación se lleva a cabo mediante llave pública, se requieren dos llaves; una pública y una privada, la primera debe almacenarse en el servidor. Una vez establecida la conexión, el servidor genera un número aleatorio y se cifra con la llave pública, se envía al cliente; el cliente debe descifrarlo con la llave privada correspondiente y devolverlo al servidor, de ésta manera se verifica que el cliente es quien verdaderamente dice ser, a este procedimiento se le conoce como “desafío”, si el desafío falla se niega el acceso. La figura 1.6 muestra un esquema de la autenticación mediante llave pública.

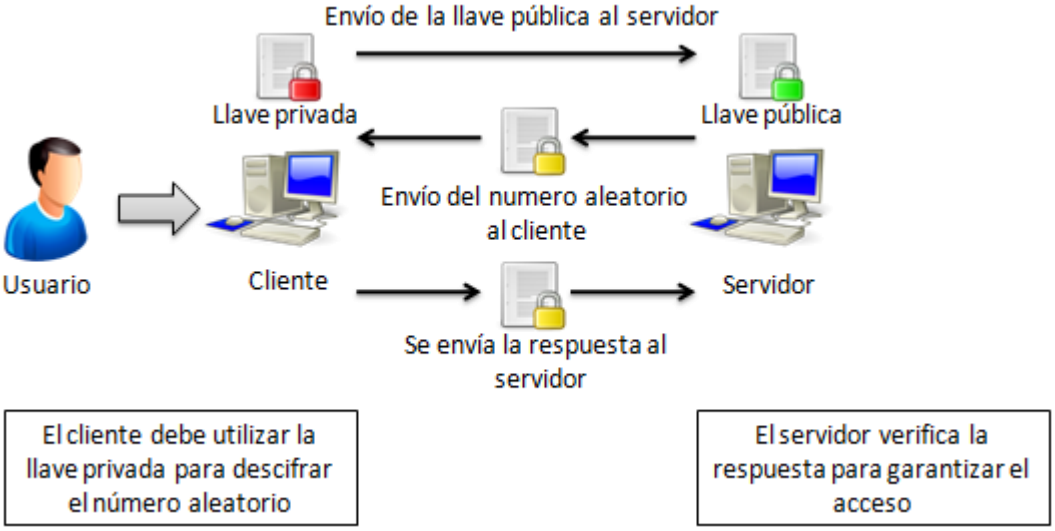

Figura 1.6 Procedimiento de autenticación mediante llave pública

Las ventajas al utilizar autenticación mediante llave pública son:

- El usuario no teclea ni debe recordar una contraseña.
- Ninguna contraseña es enviada al servidor (para este caso la llave privada del usuario hace la función de la contraseña).

1.5.3 Kerberos

Kerberos es un protocolo utilizado para la autenticación de los usuarios en los servicios de una red, está relacionado con el marco de la autenticación, autorización y el control de acceso. Proporciona autenticación y comunicación segura entre clientes y servidores mediante el uso de criptografía para verificar la identidad de los usuarios que acceden a los servidores mediante la red.

Kerberos codifica el intercambio seguro de información entre clientes, el servidor y la aplicación o servicio al que se está intentando acceder. El primer intercambio de información cuando se intenta verificar la identidad de un usuario se conoce como "*ticket*".[15] Posteriormente, para codificar los tickets y las sucesivas comunicaciones se usa una llave o "*key*". Una vez verificada la identidad del cliente y se concede la ficha o "*ticket*", puede ser utilizado para verificar su identidad ante otros servicios que utilicen Kerberos. Estos "*tickets"* se generan a partir de un servidor de autenticación llamado "*Ticket Granting Server (TGS)"* o Servidor de otorgamiento de "*tickets"*.[16] Los "*tickets*" concedidos por el TGS tienen una marca de tiempo, es decir expiran automáticamente a menos que un usuario o servicio los renueve.

Kerberos utiliza el término dominio o "*realm*" para diferenciar entre dominios de autenticación y los de Internet. Un "*realm*" o dominio Kerberos es un conjunto de máquinas que dependen de un servidor Kerberos en específico para autenticación, comúnmente son definidos en letras mayúsculas para diferenciarlos de algún otro dominio DNS similar. La figura 1.7 muestra el proceso de autenticación mediante el protocolo Kerberos para acceder a un servidor de correo.

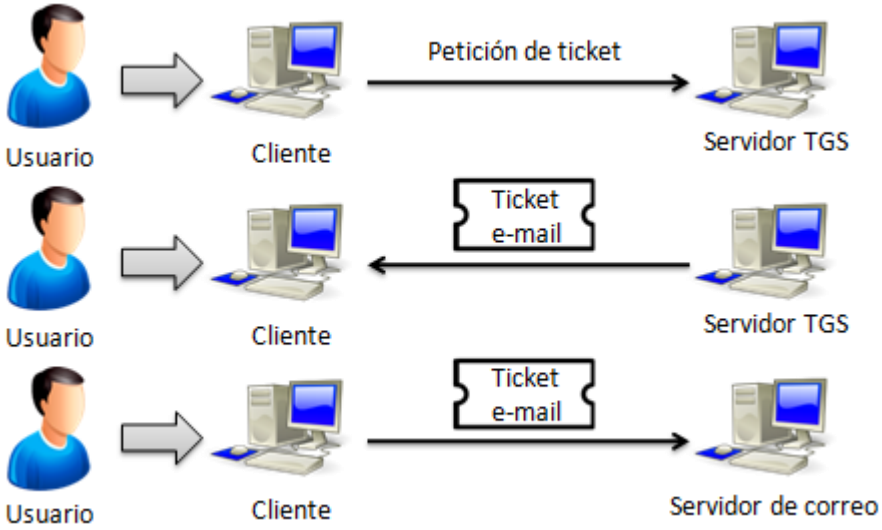

Figura 1.7 Proceso de autenticación mediante Kerberos

1.5.4 Internet Protocol Security (IPSec)

IPSec es un conjunto de protocolos para aseguramiento de comunicaciones en el autenticado y el cifrado, cifrando los paquetes IP en un flujo de datos. Se encarga de asegurar las comunicaciones a través de una LAN, WAN, red privada, red pública e internet. La figura 1.8 muestra un ejemplo de esta descripción.

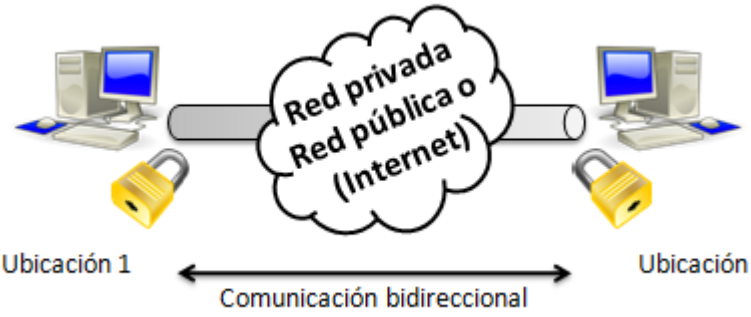

Figura 1.8 Aseguramiento de la comunicación entre dos ubicaciones

El mecanismo de autenticación de IPSec garantiza que un paquete recibido por la parte identificada como origen, fue la que lo transmitió. Además de asegurar que el paquete no fue

alterado durante su transmisión ya que cuenta con protocolos para el establecimiento de claves de cifrado para prevenir la escucha de terceros. IPSec cuenta con dos modos, modo transporte y modo túnel ambos con encabezado de autenticación.[17]

1.5.5 IPSec modo transporte con encabezado de autenticación (AH)

En modo túnel se proporciona autenticación, integridad y protege de la reproducción del paquete, lo que sería el encabezado IP y la carga de los datos. La información es legible, pero está protegida contra modificaciones. Se usan algoritmos hash con claves para firmar el paquete y asegurar su integridad.[18] La figura 1.9 muestra estos elementos y el firmado del paquete.

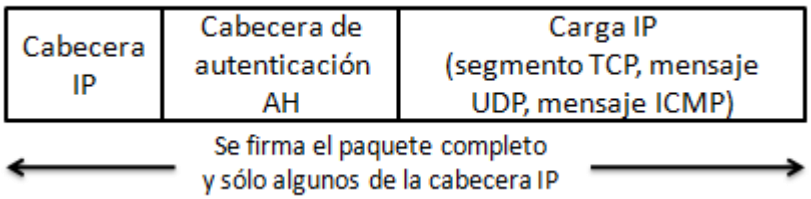

Figura 1.9 Paquete incluyendo la cabecera AH

1.5.6 IPSec modo túnel con encabezado de autenticación (AH)

Cuando se utiliza el modo túnel de IPSec el encabezado IP y la carga se cifran mientras que en el modo transporte sólo se cifra la carga IP. De este modo se protegen los paquetes IP completos, pues son considerados como una carga AH y un encabezado IP adicional. En el esquema de la figura 1.10 se muestran estos elementos.[19]

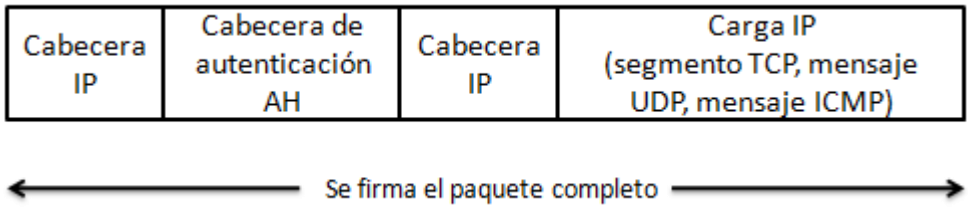

Figura 1.10 Paquete con una cabecera IP adicional

CAPÍTULO 2

PROCEDIMIENTO DE AUTENTICACIÓN

2.1 Mecanismo general de autenticación

Un sistema que posea información valiosa necesitará en primera instancia, un mecanismo de autenticación de usuarios para saber quién solicita el acceso y posteriormente verificar si le es permitido ingresar a ciertas funcionalidades o información del sistema.

Cada sistema cuenta con sus propias reglas o método de autenticación de usuarios, para lo cual, en una organización el acceso a los sistemas se vuelve un tanto independiente, por una parte, los usuarios tienen que re-autenticarse cada vez que necesiten utilizar los sistemas a los que tienen acceso, al igual que utilizar diferentes credenciales para cada sistema; por otro lado la administración de usuarios puede ejercerse por varios administradores y depende del número de sistemas con los que cuente la organización. La figura 2.1 ilustra un escenario de administración de sistemas.

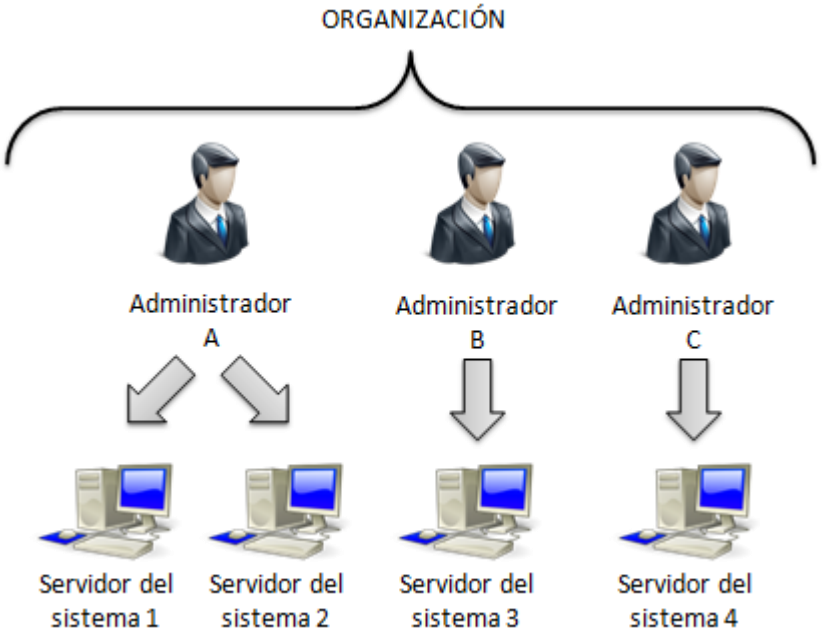

Figura 2.1 Cada sistema con su respectivo administrador

Al tener un escenario como el mostrado en la figura 2.1, cada usuario que tenga acceso a cualquiera de los sistemas, si desea acceder a algún otro, deberá solicitar el acceso y la autorización al administrador de ese sistema. Cada administrador poseerá un listado de usuarios que están registrados en su sistema, con todos los usuarios de esa lista autorizados para acceder, pero con diferente perfil de usuario de acuerdo a la información y funcionalidad del sistema.

Es posible resumir el escenario mostrado en la figura 2.1, de tal modo que la administración de usuarios se convierta en un control de acceso sobre un único listado de usuarios. La figura 2.2 describe este panorama.

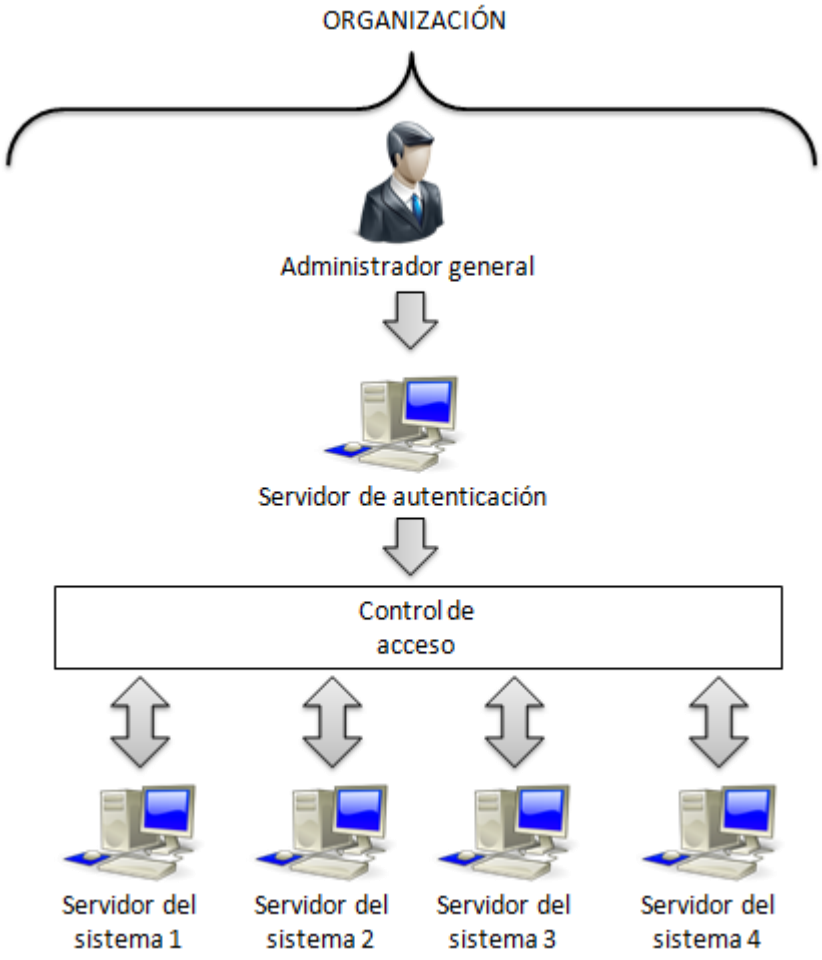

Figura 2.2 Cada sistema cuentan con un único administrador

De este modo, cuando un usuario se autentica, el control de acceso le permite al usuario ingresar únicamente a los sistemas a los cuales le fueron autorizados el acceso.

2.2 Autenticación simple

Una autenticación simple consiste básicamente en solicitar un nombre de usuario y una contraseña, este par de elementos son mejor conocidos como credenciales de acceso.[20] Esto inicia cuando se desea proteger cierta información, la cual esté disponible solamente para determinados usuarios.

Es posible realizar una autenticación de este tipo, mediante programación o utilizando las opciones y directivas que ofrece un servidor web, por ejemplo Apache. Apache ofrece protección de contenido mediante el par usuario y contraseña, estableciendo privilegios para que determinados usuarios puedan acceder al contenido restringido.

Existen dos maneras para realizar una autenticación simple, la primera consiste en proteger el contenido sin cifrar el canal de comunicación, el cual va desde el navegador web del cliente al servidor web, además que esta opción la información viaja casi en texto claro, es decir, las credenciales del usuario viajan a través del canal de comunicación codificadas en hexadecimal,[21] de este modo con un poco de tiempo y esfuerzo es posible decodificar las credenciales de usuario si éstas, fueran interceptadas por un usuario malintencionado. La figura 2.3 ejemplifica esta manera de autenticación.

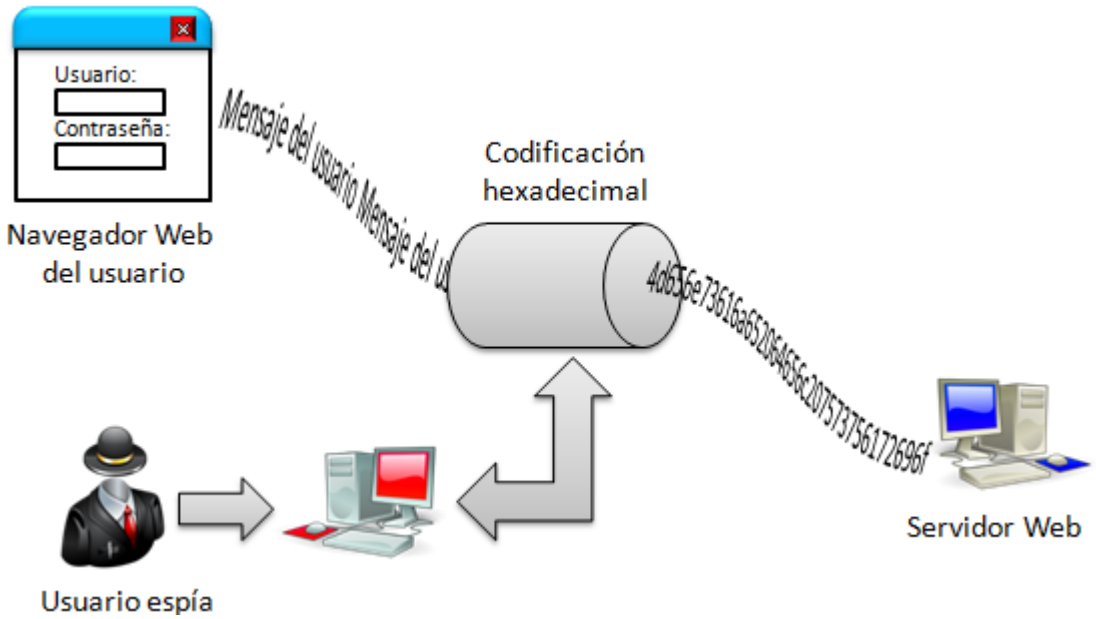

Figura 2.3 Intercepción de la información

Entonces, ¿por qué y para qué utilizar una forma de autenticación débil y fácilmente vulnerable? Muchos administradores que no estén relacionados con la seguridad pueden creer erróneamente que esta autenticación es fuerte, y tienden a creer que cualquier sistema, por el solo hecho de solicitar un usuario y contraseña es seguro, sin evaluar realmente su fortaleza.

Otra respuesta a la pregunta es que, los administradores conocen la debilidad de esta forma de autenticación, pero los datos que se protegen no tienen el nivel de confidencialidad necesario para aumentar el costo de la protección, es decir, mediante una evaluación entre costo y beneficio se permite decidir si se mantiene el método de autenticación o se fortalece.

Por otro lado, la segunda consiste en que la información viaja desde del navegador del usuario, hasta el servidor web, de esta manera los datos que proporcione el usuario en su navegador, por ejemplo sus credenciales de acceso de algún sitio web, utilizarán el protocolo SSL para su transporte y viajará cifrada la información a través de la red.[22] La figura 2.4 ejemplifica este hecho.

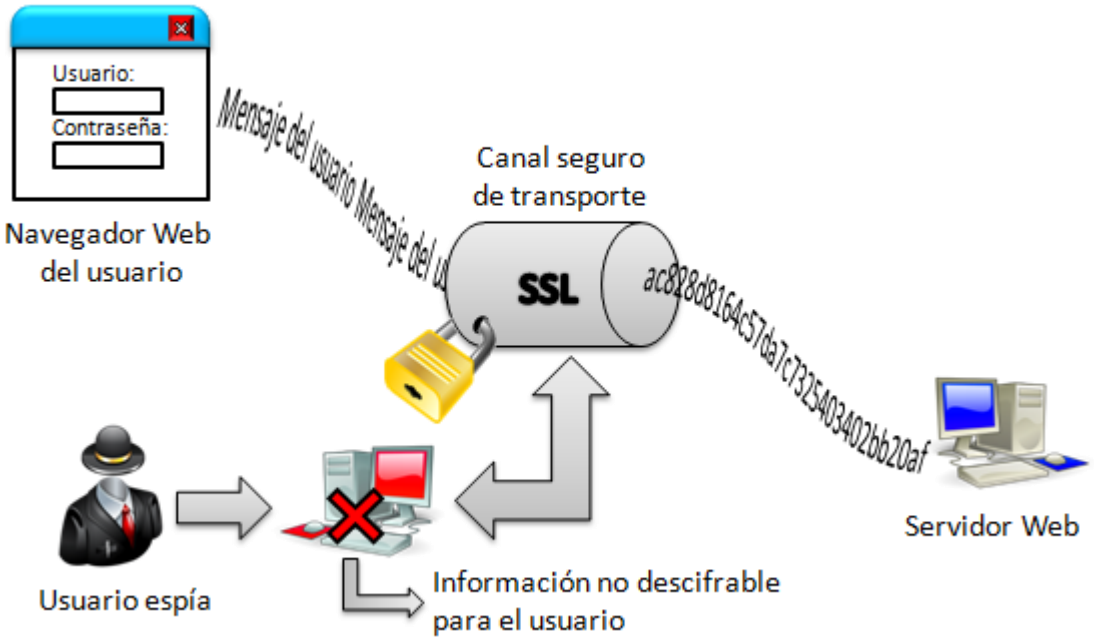

Figura 2.4 Intento fallido al descifrar información interceptada

Sin importar cuánto se esfuerce un usuario para tratar de descifrar la información, ésta no podrá ser revelada, ya que existe un canal de comunicación seguro entre el navegador del usuario y el servidor web.

La diferencia entre las dos formas de autenticación simple, radica en la manera en la que viaja la información en uno lo hace de manera codificada y la otra de manera cifrada. Cabe mencionar que de acuerdo a la información que se quiera proteger mediante autenticación, se debe utilizar un método adecuado para ello.

Por último, el cuadro de dialogo clásico para este tipo de autenticación, se ilustra en la figura 2.5.

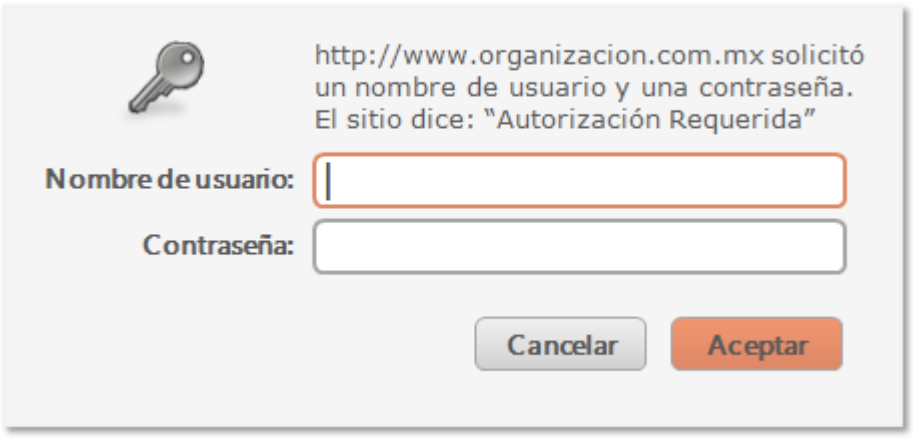

Figura 2.5 Mensaje en un proceso de autenticación simple

El recuadro mostrado en la figura 2.5 es el que comúnmente se muestra en el navegador web, solicitando un usuario y contraseñas válidos para acceder al contenido protegido dentro del sitio.

2.3 Autenticación web

La autenticación web es una manera de autenticar usuarios mediante formularios HTML, en la que se involucran varios términos para que se lleve a cabo de forma segura, y que en el ámbito de la programación web se refiere a ellos de manera muy común. Vale la pena señalar y hablar un poco acerca de ellos, ya que a lo largo del desarrollo del módulo de autenticación se irán haciendo mención de los mismos.

Cabe aclarar que algunos de estos términos no hay una definición concreta, pero es muy común hablar de ellos al momento de implementar controles de acceso a las aplicaciones web. Se explicarán de manera breve para no desviarse del objetivo de esta tesis.

2.3.1 Tráfico HTTP

El protocolo HTTP solo distingue dos tipos de mensajes, solicitudes y respuestas, que se diferencian únicamente en su primera línea. Tanto las solicitudes como las respuestas pueden incluir distintas cabeceras además del cuerpo del mensaje. En el cuerpo del mensaje es donde se transmiten los datos en sí, mientras que las cabeceras permiten especificar información adicional acerca de los datos transmitidos y las cabeceras siempre son de la forma ***clave: valor***.

Cuando se accede a una página web desde algún navegador, la solicitud que se le remite al servidor HTTP es de la siguiente forma:

```
GET http://www.miorganizacion.com/index.html HTTP/1.1
If-Modified-Since: Sat, 18 Ago 2012 19:40:00
Referer: http://www.google.com/search?...
```

La primera línea de la solicitud, aparte de indicar la versión de HTTP utilizada, también determina el método utilizado para acceder al recurso solicitado. Este recurso es identificado mediante la URL. Los métodos de acceso más usados son GET y POST, que solo se diferencian en la forma de pasar los parámetros, GET lo hace a través de la URL y POST lo hace de manera transparente al usuario e indicando al servidor que se prepare para recibir información del cliente, GET y POST suelen usarse para enviar información desde formularios. Existen a su vez otros métodos como HEAD, que solo devuelve la información de cabecera.

Tras la primera línea de la solicitud, pueden o no aparecer líneas adicionales en las cuales se añade información relativa a la solicitud o al propio cliente. En el ejemplo mostrado, `If-Modified-Since` sirve para que el cliente no tenga que descargar una página si ésta no ha cambiado desde la última vez que accedió a ella en el servidor.

Por otro lado, `Referer` indica la URL del sitio desde el que se accede a la página, algo de vital importancia si se quiere analizar el tráfico del servidor web. Incluso existen cabeceras, como `User-Agent`, mediante las cuales se pueden averiguar el sistema operativo y el navegador que usa el cliente al acceder al servidor web.

El final de la solicitud lo marca una línea en blanco. Esta línea le indica al servidor HTTP que el cliente ya ha completado su solicitud, por lo que el servidor puede comenzar a generar la respuesta adecuada a la solicitud recibida.[23]

2.3.2 Ticket de acceso

El término ticket no tiene una definición concreta, para la modalidad de autenticación de usuarios mediante formularios HTML, un ticket es una clave de identificación generada de manera aleatoria por el servidor, para determinar que el usuario se ha autenticado correctamente en la aplicación web.

La generación de un ticket no se lleva a cabo de manera automática sobre el servidor, sino que se tiene que implementar mediante un lenguaje de programación dinámico (por ejemplo PHP). Cada sistema o cada programador es libre de implementar esta generación de tickets, tomando en cuenta parámetros que permitan diferenciar un ticket de otro. Por ejemplo, en un concierto de ópera para que la gente pueda asistir, tienen que comprar una

entrada en ocasiones puede que al comprador le soliciten, o no su nombre. Una vez que efectúa la compra recibe un boleto para asistir al evento de ópera. Al revisar su boleto observa los detalles, que consisten en el número de asiento, número de fila y folio del boleto, a este conjunto de elementos lo diferencian de otro boleto que otro cliente haya comprado. Inclusive para resguardar la seguridad del boleto, se le pueden añadir elementos como un holograma, un sello, marca de agua o un tipo de papel especial.

Algo similar sucede al acceder a una aplicación web, el servidor además de comprobar la autenticidad del usuario, necesita valerse de un elemento (o varios) para identificar que el usuario se ha autenticado correctamente (que haya comprado su boleto). Este elemento es el ticket, el cual se genera al momento de validar las credenciales del usuario (que haya pagado su boleto).

Antes de entrar al contenido protegido el sistema verifica las características del ticket, para determinar que el ticket sea genuino (similar cuando se llega a la entrada del lugar del evento de ópera y la persona encargada de revisar el boleto, verifica ciertas características del boleto, papel, holograma, folio, etc.).

En un sistema web, la implementación de un generador de tickets se efectúa mediante elementos conocidos del usuario que intenta acceder al sistema, por ejemplo se puede emplear la cuenta de acceso (login del usuario), la IP de la máquina del cliente, la fecha, la hora entre otros elementos posibles (similar al folio, número de fila, número de asiento del boleto de ópera).

Generar el ticket con estos elementos en claro, posibilita la falsificación del mismo y por tanto se deben añadir elementos de seguridad (como el holograma, tipo de papel o sello mencionado anteriormente). Para esto, se pueden emplear métodos de cifrado, los cuales incorporan la generación de una cadena de caracteres de longitud fija. Por ejemplo, basta con concatenar el login del usuario, la hora, la fecha y una cadena de 10 caracteres generada al azar, y aplicar el método de cifrado (md5, sha1, etc.) y generar un ticket con una longitud de caracteres fija, de este modo será posible diferenciar un ticket de otro usuario que ingrese al sistema.

De manera metafórica el servidor estaría realizando la verificación del ticket de la siguiente forma:

> *"Buen día, ¿quieres acceder al sistema? Muy bien, proporciona tus credenciales de acceso, ¡bien! Son válidas tus credenciales, aquí tienes tu ticket de acceso dirígete a la segunda puerta a mano derecha y muestra tu ticket al guardia de la entrada. (Al llegar con el guardia) Hola,*

permíteme tu ticket de acceso, muy bien tu ticket es válido, adelante puedes acceder… Bienvenido."

La verificación del ticket también es una tarea importante, la cual consiste en implementar una función que verifique la autenticidad del ticket, es decir, se debe indicar a la función verificadora cuáles fueron los elementos que ayudaron a generar el ticket de acceso, para que el servidor calcule de nueva cuenta el ticket y lleve a cabo la comparación entre el recalculado y el que entrega el usuario al momento de ingresar al contenido protegido. La figura 2.6 muestra el proceso de generación y verificación del ticket en un sistema web.

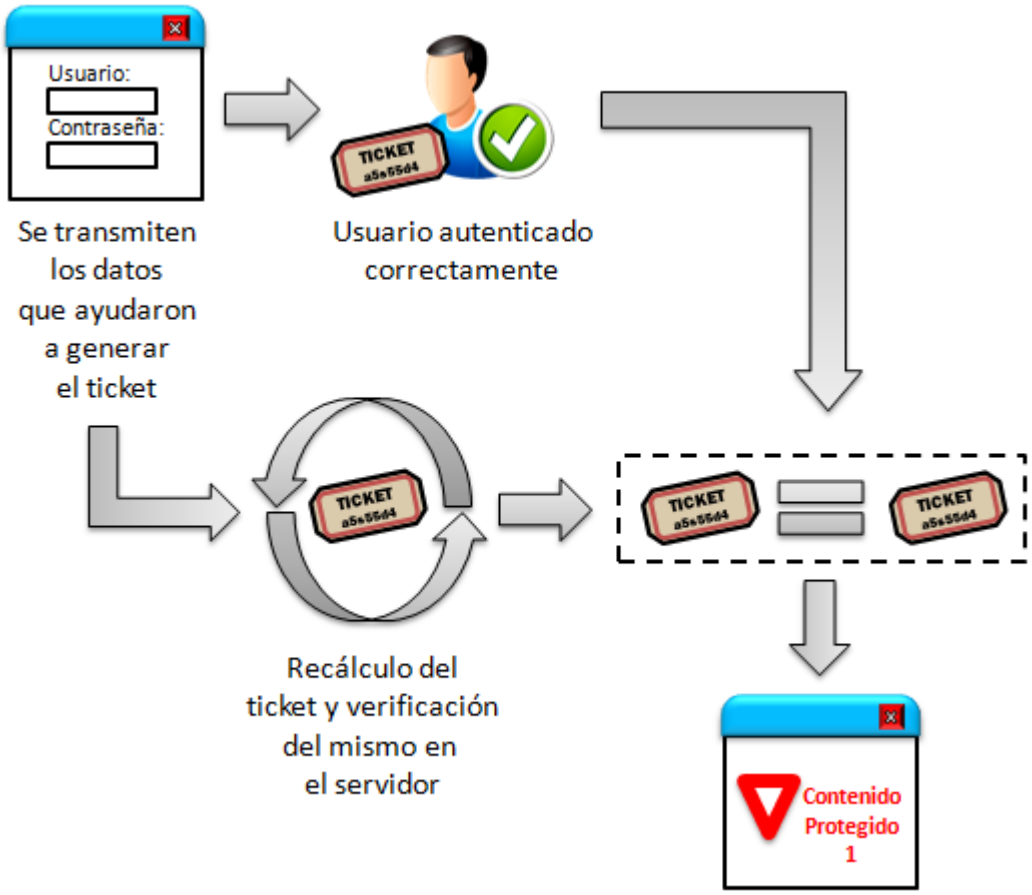

Figura 2.6 Recálculo y verificación del ticket de acceso

En el caso de que la verificación del ticket no sea válida se deniega el acceso a todo contenido protegido.

Del mismo modo que se ha mencionado el término "ticket", se encuentra el término token de acceso. Es un proceso muy similar al descrito con el ticket de acceso, en el que se involucra una generación, una validación y un lapso de tiempo de validez del token, con la

salvedad de que el token de acceso es ampliamente utilizado para acceder de una aplicación web a otra, lo que permite solamente estar registrado en una aplicación que posea la API del token de acceso para poder acceder a otra aplicación web que tenga soporte para el mismo, sin necesidad de estar registrado en esta última aplicación.

De manera general el ticket se implementa con código fuente adicional sobre el propio sistema web para acceder a él, y el token se emplea para el acceso de un sistema web a otro. Su manera de empleo y modo de operar entre aplicaciones web se verá más adelante.

2.3.3 Gestión de una sesión en el proceso de autenticación HTML

Una sesión es un intercambio no permanente de información entre sistemas web,[24] que requiere mantener ciertos datos de estado de un usuario, conforme va accediendo al contenido del sistema desde que ha ingresado, hasta que sale del mismo.

El empleo de las sesiones es muy recurrente en los sistemas web que poseen autenticación de usuarios, y existen muchos usos. Uno de los más utilizados es para controlar el acceso a las páginas que visita el usuario dentro del sistema web, a su vez permiten controlar el acceso no autorizado a las páginas en las que únicamente un usuario autenticado puede acceder.

Para mencionar la manera de gestionar una sesión, se hará énfasis en el lenguaje PHP, ya que el desarrollo de esta tesis se basa en este lenguaje dinámico. De igual forma en cualquier otro lenguaje dinámico se lleva a cabo el mismo proceso, solo cambian los nombres de las funciones y la sintaxis del lenguaje, pero el concepto es el mismo.

La sesión comienza cuando se crea un ID, este ID el lenguaje PHP lo genera de forma predeterminada comúnmente se llama PHPSESSID (es posible cambiarlo dentro de la configuración del lenguaje), PHP permite utilizar ese ID o generar ID’s propios conforme a necesidad.

Este ID consiste en un conjunto de caracteres univoco, es decir, para cada usuario que acceda al sitio web un ID diferente será generado, si se utiliza el valor de sesión PHPSESSID, PHP lo hace de manera automática.

IMPORTANTE: si se utiliza el valor de sesión PHPSESSID, se deberá utilizar una conexión cifrada (mediante SSL) entre el cliente y el servidor, ya que éste valor viaja en claro a través de la red, y puede ser interceptado por un atacante que esté escuchando la red y realizar un robo de sesión.

Ahora si se desea administrar de manera autónoma la sesión y generar ID's para cada usuario que visite el sitio web, es posible hacerlo definiendo funciones que proporcionen la generación de dichos ID's, tomando en cuenta que cada ID debe ser diferente para cada usuario que acceda al sitio.

Una vez que se tiene el ID (ya sea el nativo o los generados mediante código fuente), éste tiene que ser enviado a todas las páginas del sitio web, para indicarle a cada página del sistema que un usuario ha iniciado sesión correctamente y permitirle el acceso. Si un usuario conoce la URL del contenido restringido, y la ingresa directamente en el campo de dirección del navegador, éste no podrá ver el contenido de la página, ya que no ha iniciado una sesión en el sistema.

El envío y la validación del ID a través de las páginas se puede hacer: mediante campos ocultos en el HTML y enviarlo mediante el método POST (este método se verá en un ejemplo más adelante), por medio de la URL del navegador (no se recomienda), mediante el envío de una cookie al navegador (si el usuario tiene deshabilitadas las cookies en el navegador, será imposible crear una sesión), mediante el guardado del ID en una base de datos y finalmente utilizando variables de sesión PHP.

Cuando se utilizan variables de sesión para la gestión de la misma, se deben crear funciones propias que generen los ID's de sesión. La generación de los ID's puede estar basada en datos del usuario como por ejemplo, la cuenta del usuario, la fecha, la dirección IP de donde está accediendo, la contraseña, una cadena llave, etc.

¿Cómo generar estos ID en base a los datos o parámetros de un usuario?, para ello es conveniente utilizar algún método de cifrado como por ejemplo DES, MD5, SHA1, SHA256, entre muchos otros. Una vez que se ha seleccionado un método de cifrado se deberá programar un función hash que genere dichos ID's. El ID resultante de esta función puede ser un hash de sesión o simplemente un ID de sesión, que será enviado a todas las páginas del sistema web para gestionar y mantener la sesión del usuario.

2.3.4 Un ejemplo ilustrativo

La forma de autenticar usuarios mediante un formulario HTML, consiste en generar comúnmente dos campos destinados para el nombre de usuario y una contraseña, en ocasiones si se quiere ser más estrictos y evitar ataques de fuerza bruta, se puede colocar un tercer campo que consista en reescribir una palabra generada por el propio sistema de forma aleatoria, a lo que se le conoce como un captcha. En la figura 2.6 se muestra un formulario solicitando un nombre de usuario (una dirección de correo electrónico) y una contraseña.

Figura 2.7 Formulario HTML con los campos de nombre de usuario y contraseña

Como nota adicional, cuando un usuario ha iniciado sesión en el sistema puede suceder que el sistema permita el guardado del nombre de usuario en una cookie, y una vez que el usuario ha salido del sistema y posteriormente requiera volver acceder el único campo que tendría que proporcionar sería el de su contraseña y si se añade un captcha, adicionalmente resolver el captcha. El código para generar el formulario de la figura 2.7 se muestra en la tabla 2.1

Tabla 2.1 Código para el archivo index.html

CÓDIGO: index.html

```
1. <html>
2. <form method="POST" action="agente.php">
3. Ingresa tu correo: <input type="text" name="usuario">
4. <br>
5. Ingresa tu contraseña: <input type="password" name="contrasena">
6. <br>
7. <input type="submit" value="Enviar" name="btn_enviar">
8. </form>
9. <html>
```

El código de la tabla 2.1 en la línea 2, se pueden apreciar dos atributos que corresponden a "*method*" y a "*action*", el atributo "*method*" consiste en la manera de cómo serán enviados los datos al servidor, este atributo y el valor que toma será tratado más adelante.

El atributo "*action*" corresponde a especificar un nombre de archivo que será el encargado de procesar los datos en el servidor, se observa en el código que se ha proporcionado un nombre de archivo llamado "*agente.php*".

El contenido del archivo "*agente.php"* tendría que ser algo parecido al que se muestra en la tabla 2.2 para captura y manipulación de los datos ingresados en el archivo "*index.html*".

Tabla 2.2 Código en PHP para el procesamiento de datos

CÓDIGO: agente.php

```
1.  <?php
2.  session_start();
3.  if ($_POST['btn_enviar']){
4.      $user = $_POST['usuario'];
5.      $password = $_POST['contrasena'];
6.      $user = validaUsuario($user);
7.      $acceso = validaPass($user,$password);
8.      if ($acceso){
9.          $ticket = creaTicket($user);
10.         $_SESSION['idSesion'] = creaSesion();
11.         $idSesion = $_SESSION['idSesion'];
12.         $_SESSION['ticket'] = $ticket;
13.         guardaInfoSesion($user,$ticket,$idSesion);
14.         header ("http://www.miaposento.com/access.php?usuario=$user");
15.     }
16.     else{
17.         msjError("novalido");
18.     }
19. }
20. ?>
```

Obsérvese con atención la línea 2 del código de la tabla 2.1, se observa que el atributo "*method"* tiene como valor POST, esto significa que los datos serán enviados al servidor de manera transparente sin que el usuario cliente lo note.

El método POST es el recomendado para cualquier forma de envío de información y más aún tratándose de información sensible, especialmente si son credenciales de acceso.

En la línea 2 del código de la tabla 2.2, se crea una sesión para poder rellenar automáticamente la variable super-global $_SESSION utilizada en la línea 10 y 12. En la línea 3, se verifica si el botón enviar de la figura 2.7 se ha pulsado y con ello se captura dicho evento, y se comienzan a recuperar los datos.

Las líneas 4 y 5 recuperan los valores del nombre de usuario y la contraseña mediante el arreglo asociativo $_POST de PHP y se asignan a variables más fáciles de manipular en el código. El arreglo $_POST es una variable super-global, que está disponible en cualquier parte del script, y se encarga de recibir toda la información que provenga de un envío a través del método HTTP POST.

Las líneas 6 y 7 realizan llamadas a funciones para procesar el nombre de usuario y la contraseña, la función *validaUsuario* deberá sanear la entrada del usuario, es decir, quitar todo tipo de carácter que pueda causar problema al momento de utilizar de nueva cuenta la variable $user. La función *validaPass* deberá validar la contraseña almacenada contra la que el usuario ha proporcionado, y deberá regresar un valor que identifique si se llevó a cabo correctamente la validación o no.

La línea 8 verifica si la variable $acceso contiene un valor de aceptación regresado por la función *validaPass*, es decir, si contiene un valor booleano *true* para dar acceso a la siguiente parte del script, de no ser así significa que la validación de la contraseña no fue satisfactoria, y se procede a saltar a la línea 17 re-direccionando a un despliegue de error y saliendo del script.

Al momento de validarse correctamente la contraseña, en la línea 9 se creará un *ticket* de acceso, para poder ser dirigido a la siguiente página; así mismo se generará un identificador de sesión en la línea 10 mediante la función *creaSesion*, para poder acceder a la página *access.php*, la cual está restringida y sólo se puede acceder a ella con un número de *ticket* válido y que irá implícito en el cálculo de un identificador de sesión, sólo el servidor dentro de la función *creaTicket* lo puede generar utilizando el nombre de usuario como parámetro.

Las líneas 11 y 12 corresponden a la manipulación del identificador de sesión creado, y los valores de las variables $user, $ticket y $idSesion, necesitarían ser almacenados en una base de datos para poderlos verificar en un proceso posterior e impedir que sean modificados si estos llegaran a ser interceptados, para ello en la línea 13 mediante la función *guardaInfoSesion,* se guardan estos valores en la base de datos.

Finalmente en la línea 14, se observa un re-direccionamiento a la página "*access.php*" y en seguida se ve el carácter "?", y una asignación de un valor. ¿Qué significa? Significa que cuando se hace una asignación de este tipo, se está utilizando el método GET para el envío de datos por la URL y recuperarlos en páginas más adelante.

Para este caso se envía el valor que contiene la variable $user y se asigna a la variable *usuario* dentro de la URL. Una vez que se ha accedido a la página "*access.php*", dentro de ella se puede recuperar el valor del usuario mediante lo siguiente:

```
$userTemp=$_GET['usuario'];
```

De esta forma la variable $userTemp recibiría el valor de la variable usuario que se muestra en la URL. El método GET envía los datos a través de la URL, y estos pueden ser fácilmente visualizados e interceptados, por lo que no se recomienda enviar datos sensibles mediante el método GET.

Para el manejo de sesiones si se observa en las líneas 2, 10 y 12 del código de la tabla 2.2, se tienen indicios acerca de algo llamado sesión.

Las sesiones, como se mencionó en el subtema anterior, permiten tener un control sobre las páginas que se visitan, ya que son un método para hacer que determinadas variables estén disponibles en múltiples páginas, sin necesidad de pasarlas como parámetro, y poder ejercer un control mediante decisiones, en el que se pueda verificar si una variable de sesión tiene un determinado valor, con el cual se determine si se concede el acceso o no.

Para hacer uso de variables de sesión, es necesario en primera instancia hacer una llamada a la función PHP *sesion_start*, de no hacerlo así será imposible obtener algún valor del arreglo $_SESSION.

Para ejemplificar la gestión de la sesión, se harán referencias al código de la tabla 2.2 y se empleará el código siguiente de la tabla 2.3 que corresponde al contenido del archivo "*access.php*".

Tabla 2.3 Código del archivo access.php

CÓDIGO: access.php

```
1. <?php
2. session_start();
3. $usuario = $_GET['usuario'];
```

```
4.  $genuinoIdSesion = recuperaIdSesion($usuario);
5.  $tmpIdSesion = recalculaIdSesion($usuario,$_SESSION['ticket'],$_SESSION['idSesion']);
6.  if ($genuinoIdSesion == $tmpIdSesion){
7.      echo "Estás autorizado a ver esta página y su contenido";
8.      }
9.      else{
10.         msjError("nosession");
11.     }
12. }
13. ?>
```

En el código de la tabla 2.3 en la línea 2 se tiene la llamada a la función *session_start*, para poder recuperar correctamente los valores de las variables, que se generen dentro del código de "*agente.php*", y que corresponden a $_SESSION['ticket'] y $_SESSION['idSesion'].

En la línea 3 mediante el método GET se obtiene el valor del nombre de usuario y se asigna a una variable más fácil de manipular en el código. Previamente en el código de la tabla 2.2 línea 13 se guardó en la base de datos la información de las variables $usuario, $ticket e $idSesion; supóngase que lo que se guardó fue el cálculo de un md5 con la concatenación de esos tres valores, por lo tanto, en la base de datos se encuentra un hash md5 de 32 caracteres asignada al usuario que se autenticó.

Con base en el md5 guardado en la línea 13 de la tabla 2.2, en la línea 4 de la tabla 2.3 se recupera ese md5 con la función *recuperaIdSesion* y el parámetro $usuario. En la línea 5, la función *recalculaIdsesion* lo que hará, será utilizar los parámetros $usuario, $_SESSION['ticket'] y $_SESSION['idSesion'], para volver a generar el hash md5 y asignarla a la variable $tmpIdSesion.

Si las variables de sesión no fueron interceptadas y modificadas por un usuario malicioso, el recálculo del hash md5 deberá ser satisfactorio, es decir, al llevarse a cabo la comparación de éste en la línea 6, contra el guardado en la base de datos, deberán coincidir.

Al ser verificada correctamente la sesión, será posible ver el mensaje desplegado en la línea 7. De cualquier otro modo si la comparación no coincide se redirigirá al usuario a otra página con un mensaje de error alusivo. En la figura 2.8 se puede ver gráficamente el proceso descrito de los códigos fuente.

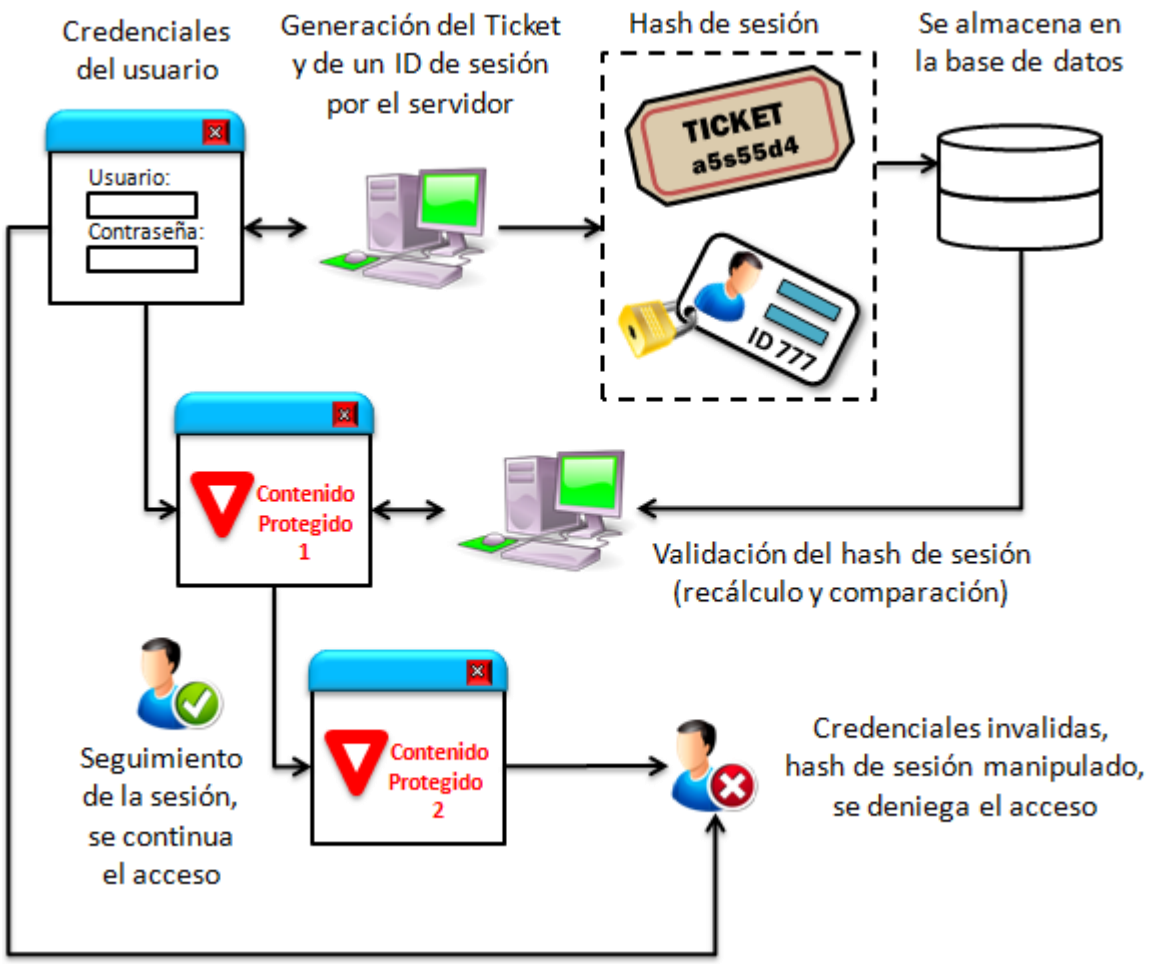

Figura 2.8 Esquema gráfico del manejo y seguimiento de una sesión.

El recálculo de la sesión, se puede hacer cada vez que el usuario navega entre las páginas, para verificar que la sesión de ese usuario todavía sigue siendo válida, y verificar que las variables de sesión generadas inicialmente cuando el usuario se autenticó, no hayan sido manipuladas por un tercero, e impedir un robo de sesión.

2.4 Autenticación mediante certificados

Los certificados digitales son emitidos por terceros conocidos como Autoridades Certificadoras (CAs), usando el formato X.509, estándar de la industria. La CA firma digitalmente el certificado usando su propia llave privada, de modo que da fe de la identidad del portador protegiendo el certificado de falsificaciones.

Los certificados digitales enlazan la información del propietario del certificado con el propietario de la llave pública, que es usada para cifrar los datos. Normalmente, un servidor

Web que maneja información sensible, cifra el tráfico entre el usuario y el servidor, especialmente cuando los datos viajan a través de redes tal como lo es Internet.

El servidor Web y el cliente presentan un certificado digital con dos propósitos:

- Autenticación:
  - Probar la identidad del servidor Web: Esto ayuda a prevenir que los usuarios sean redirigidos a sitios maliciosos.
  - Probar la identidad del cliente: Conceder el acceso al cliente que posea un certificado de cliente válido.
- Confidencialidad:
  - La llave pública del servidor Web se incluye en el certificado digital y facilita la encriptación.

Los certificados de cliente y de servidor no están relacionados entre sí. Los certificados de SSL proporcionan funcionalidad de cifrado y seguridad. Los de cliente proporcionan funcionalidad de autenticación de usuario.

Una CA (Autoridad Certificadora) es la encargada de emitir certificados a usuarios o entidades que los soliciten, esto se lleva a cabo mediante una petición de certificado generada por la entidad o el usuario, ésta se envía a la CA, la CA verifica la su autenticidad, ya que se ha comprobado y verificado la autenticidad, la CA devuelve un certificado firmado mediante el certificado raíz de la propia CA, conformado por una clave privada que se mantiene sólo por el usuario o la entidad a la que se emitió el certificado, además de que esto tiene un costo y una vigencia.

La CA puede ser una organización pública conocida, que proporciona este servicio como parte de su negocio o puede ser un servidor interno que sea utilizado solamente para la compañía, empresa o área de trabajo. En cualquier caso, el certificado de cliente tendrá cierta información que identifica al usuario de manera única. La figura 2.9 ilustra el procedimiento de solicitud de un certificado a una autoridad certificadora externa.

Figura 2.9 Petición de certificado a una autoridad certificadora externa y reconocida

Cuando no se cuenta con un certificado firmado por una CA externa, se puede crear una y generar un certificado raíz con el cual se pueda firmar cualquier certificado de servidor para la organización.

Para entender un poco más acerca de los certificados, se puede traducir esto a lenguaje coloquial humano de la comunicación que existe entre el cliente y el servidor, por ejemplo para el caso de que una CA emita el certificado firmado por ella, se tiene la siguiente comunicación:

"*Hola navegador, soy el servidor www.miaposento.com, te envío mi certificado con mi clave pública que te permitirá autenticarte con mi clave privada en mi servidor, esto te lo mando en un certificado firmado nada más y nada menos por la CA Verisign. Verisign ya verificó que sí soy la empresa Social Networking Co., me cobró sus honorarios por esto, así que mi certificado debe ser autorizado sin mayor problema por los certificados raíz que se encuentran en tu lista de certificados confiables.*" Y la conexión se establece sin ningún problema.

Ahora para el caso de un certificado auto-firmado por una CA propia:

"Hola navegador, soy el servidor www.miaposento.com tienes que confiar que soy de la empresa Social Networking Co. y nadie más, para demostrártelo te envió mi clave pública que te permitirá autenticarte con mi clave privada en mi servidor, todo esto te lo mando en este certificado que espero aceptes, ya que yo mismo me convertí en mi propia CA. Y de veras yo el

firmante Social Networking Co. te juro que soy yo, créeme, acéptame, sí soy yo, por favor aprieta Aceptar para poder establecer la conexión segura.". Si el certificado no es aceptado, no se realiza ninguna comunicación entre el cliente y el servidor.

Cuando se trata de una autenticación mediante certificados, se necesita contar con un certificado de cliente válido y vigente ya que el servidor Web será lo primero que validará, deberá de contar con información básica como el nombre común (CN), la unidad organizacional (OU), correo electrónico (E),y la palabra clave del certificado (*passphrase*).

Además se requiere que el servidor Web se configure a modo de que solicite el certificado de cliente, y se utilice de manera forzosa el protocolo SSL. También se debe de configurar dentro de éste, las variables de ambiente que contienen la información del certificado, para que sean accesibles por el lenguaje de programación Web.

Al momento de acceder al contenido restringido, se verifica la validez del certificado y su vigencia. Una vez hecho esto se acceden a las variables de entorno que contienen los datos del certificado, esa información se manipula en el lenguaje de programación para conceder el acceso al contenido según sea el caso. La figura 2.10 muestra la manera en la que se lleva a cabo la autenticación de un usuario mediante un certificado de cliente.

Figura 2.10 Autenticación mediante un certificado de cliente

Un certificado inválido, con fecha expirada o con información de usuario no autorizado, el acceso al contenido será denegado.

2.5 OAuth y token de acceso

El estándar OAuth fue desarrollado para dar a los proveedores de un servicio una manera de poder implementar el modelo de autorización de usuarios, en el que un sitio web A pueda acceder de manera segura, previa autorización del usuario, a datos de acceso restringido de dicho usuario contenidos en otro sitio web B, es decir que el sitio web B pone la información del usuario a disposición del sitio web A.

La creciente de servicios en Internet es inminente, y cada vez que se desee acceder a uno de ellos es necesario registrar datos propios y crear una cuenta de acceso, pero ¿qué sucede si se tiene acceso a más de 5 servicios?, la situación se complica al tener 5 contraseñas diferentes para estos servicios, y utilizar una misma cuenta de usuario y contraseña para acceder a ellos implica un riesgo de seguridad muy grande, más aún con una contraseña débil.

Pensando en la problemática a la que se enfrentan los usuarios en cuanto a recordar varias contraseñas, resultan soluciones como OpenId y OAuth. La mayoría de los grandes servicios soportan OpenId (WordPress, Blogger), OAuth (Twitter, Vimeo, Digg, Foursquare, Hi5) o ambos (Google, Yahoo, MySpace, Facebook). Estas dos soluciones coinciden en tomar en cuenta un modo de autenticación universal, aunque OAuth es más extenso.

Para llevar a cabo una autenticación y concesión de privilegios al contenido protegido por un usuario, se requieren de tres partes, un proveedor de servicios, un consumidor y por supuesto un usuario. A continuación se describen estas partes:

- **Proveedor de servicios (*Service Provider*)**: son los sitios web que contienen información de usuario de acceso restringido. Entre los que destacan Facebook, Twitter o Google. Estos proveedores son los que ofrecen al desarrollador una API que soporta el protocolo de autenticación OAuth.
- **Usuario, propietario/dueño (Resource Owner)**: También conocido como recurso protegido, propietario o dueño *(Resource Owner*). Es el propietario del recurso que se intenta acceder, y es necesario que realice un inicio de sesión con alguno de estos proveedores de servicio, para que OAuth se lleve a cabo.
- **Consumidor o Cliente (*Consumer*)**: Es cualquier sitio o aplicación web, móvil o de escritorio que solicita permiso para acceder a información restringida que almacena un proveedor de servicios. El usuario es el único que puede conceder o denegar la autorización al consumidor.

OAuth ofrece algunas grandes mejoras sobre los modelos tradicionales de autenticación, como lo es la autenticación básica y consisten en lo siguiente:

- En vez de tener que teclear las credenciales del usuario en el proveedor de servicio con cada solicitud de autenticación, se trabaja con un acceso abstracto de tokens en el que no se comparte ninguna contraseña del usuario.
- Los tokens son emitidos del sitio proveedor, estos pueden ser revocados en cualquier momento, estableciendo más control en las manos del usuario. Varios proveedores también implementan un mecanismo de expiración de token, que solicita a la aplicación que periódicamente renueve el token de acceso para seguir haciendo peticiones a los datos del usuario.

Los usuarios pueden ver los tokens que han activado en el proveedor, por ejemplo las aplicaciones que pueden acceder a sus datos, significa que pueden manualmente revocar el acceso a una aplicación. Puesto que la aplicación no tiene las credenciales de inicio de sesión (*Login credencials*) del usuario, no puede hacer más peticiones a sus datos una vez que se ha revocado la autorización.

La manera en la que opera OAuth es la siguiente:

1. La aplicación redirige al usuario hacia el proveedor con el fin de hacer que acepte las condiciones de la aplicación, de este modo le permite el acceso a sus datos protegidos.
2. Una vez que el usuario acepta las condiciones y delega permisos a la aplicación, el proveedor redirige al usuario de vuelta a donde surgió la petición original. El proveedor también envía un parámetro con un código de verificación, que indica que el usuario ha aceptado las condiciones.
3. Una vez que se ha regresado a la URL original, el cliente envía una petición al servidor de autorización para obtener un token de acceso para el usuario, enviando el código de verificación con la solicitud.

El servidor de autorización envía de regreso un token de acceso junto con un token de actualización opcional, que depende si el servidor especifica un límite de tiempo para el token de acceso. La figura 2.11 ilustra el proceso de funcionamiento.

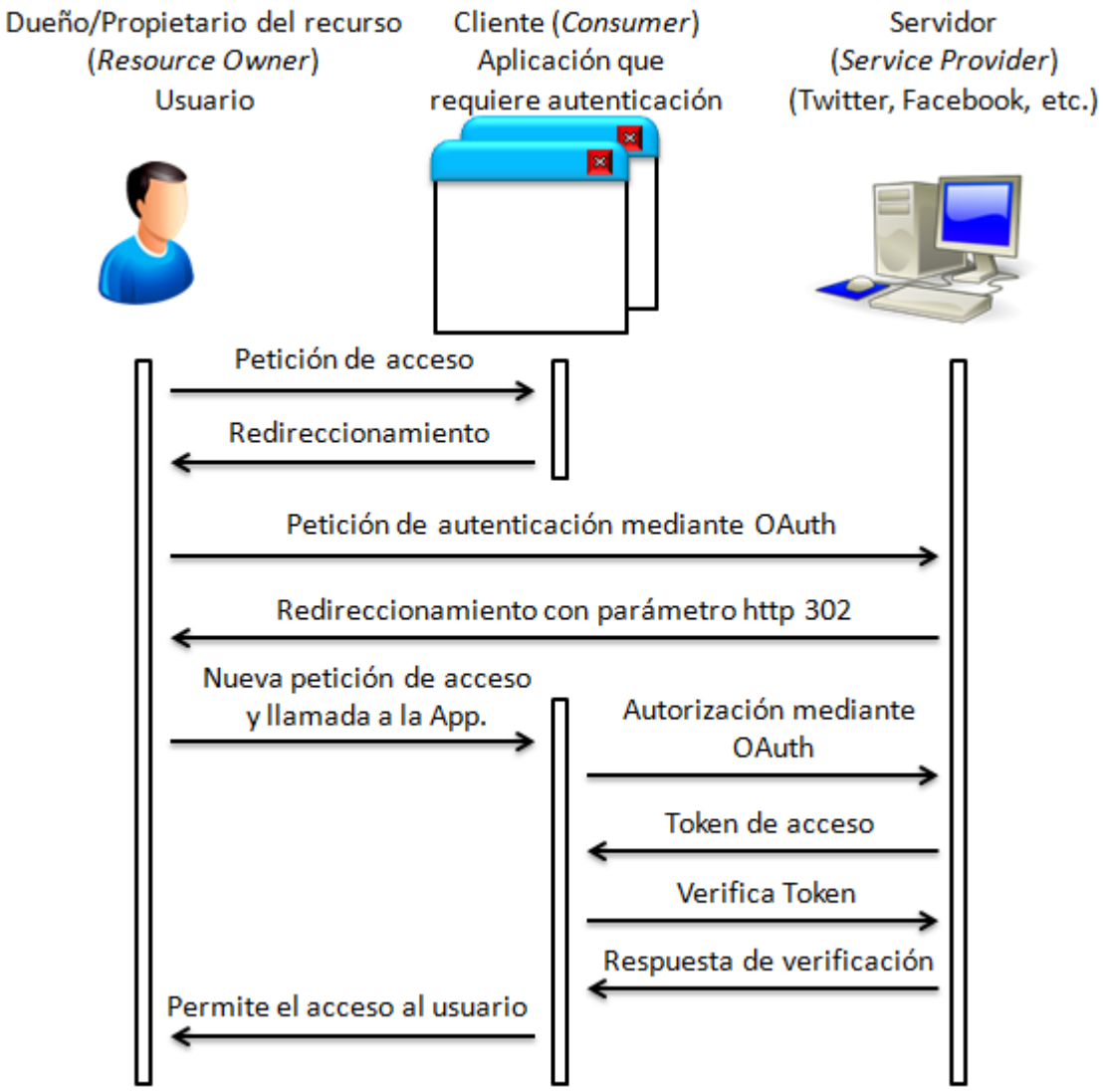

Figura 2.11 Proceso de autenticación mediante OAuth.

Aplicando esta teoría del funcionamiento de OAuth, se tiene el siguiente ejemplo:

Se desea tener acceso al servicio de redes sociales Hi5, para comunicarse con amigos y familiares, al acceder a Hi5 el sistema presenta un login de acceso con varias opciones, ver figura 2.12.

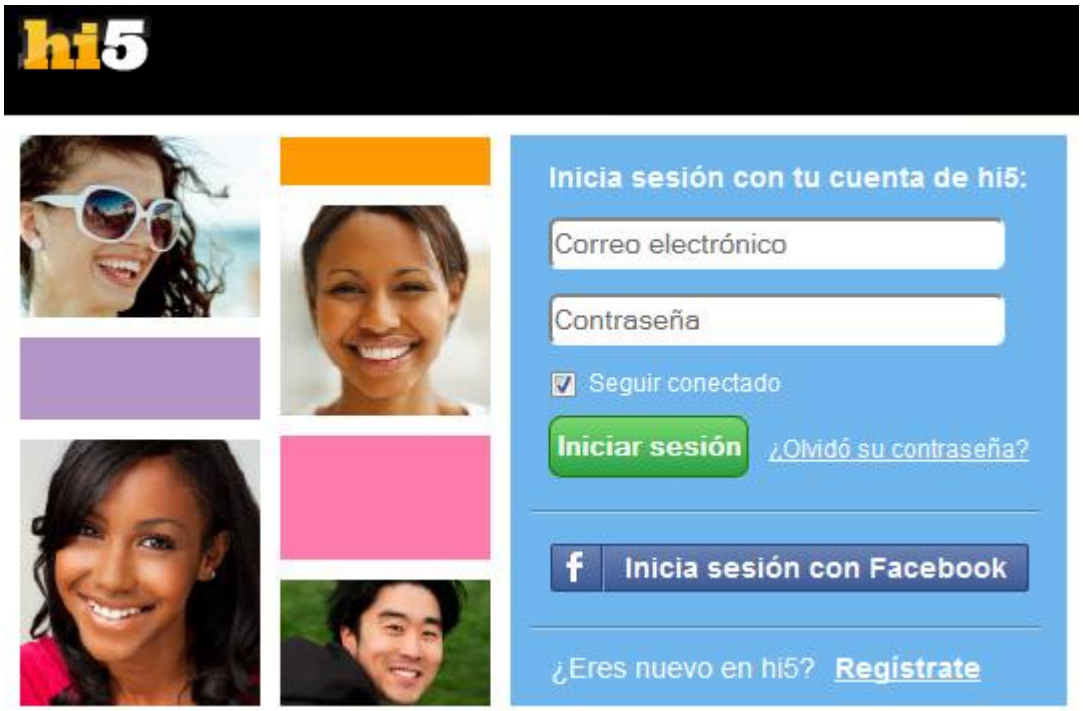

Figura 2.12 Login de acceso a Hi5

Para acceder se tienen 3 opciones, en primera instancia sería tener una cuenta e iniciar sesión, la segunda, se es nuevo en el sitio y por lo tanto se tendría que crear una cuenta, y por último se observa que existe un botón con la leyenda “Inicia sesión con Facebook”.

Afortunadamente se posee una cuenta en Facebook, entonces se puede utilizar la cuenta para acceder a Hi5 sin necesidad de registrarse. Al pulsar el botón de “Inicia sesión con Facebook” se obtiene una pantalla como la siguiente, ver figura 2.13

Figura 2.13 Login de acceso a Facebook para acceder a Hi5

En ella se introducen las credenciales de acceso a Facebook y se pulsa Entrar. Después Facebook realiza un control de acceso, preguntando si se desea acceder a Hi5, y a su vez indica qué información personal almacenada en Facebook será compartida con Hi5, ver figura 2.14.

Figura 2.14 Control de acceso de la aplicación en Facebook

Inmediatamente después se solicita la aprobación para publicar cierta información en el perfil de Hi5 y permitirle acceso a determinados datos de usuario cuando el sistema Hi5 así lo requiera, ver figura 2.15.

Figura 2.15 Control de acceso a información personal del usuario

Finalmente, se ha tenido acceso al perfil de usuario en Hi5, ver imagen figura 2.16.

Figura 2.16 Perfil de usuario en Hi5

El acceso a otras aplicaciones mediante OAuth es muy sencillo y transparente al usuario, simplemente basta tener una cuenta de usuario en uno de los tantos servicios que soportan OAuth, para acceder a otro servicio web sin la necesidad de registrarse y crear una nueva cuenta de usuario, por supuesto, siempre y cuando que también soporte la autenticación mediante OAuth.

CAPÍTULO 3

INFRAESTRUCTURA Y MÓDULO DE AUTENTICACIÓN

3.1 Servicio de directorio y LDAP

A menudo un servicio de directorio es confundido con una base de datos,[25] debido a que comparten ciertas características importantes, como la velocidad de búsqueda, el guardado y la lectura de los datos. Una de las características que hace diferente un directorio de una base de datos, es que está diseñado para leer más de lo que se escribe en él. Una base de datos además de realizar la lectura de datos, efectúa la operación de escritura en promedio, en una misma frecuencia.

Un directorio en esencia, se utiliza frecuentemente para lectura y muy poco para escritura de datos, características intrínsecas de una base de datos y que no lo son para un servicio de directorio como lo es LDAP. LDAP (*Light Directory Access Protocol* por sus siglas en inglés), es un protocolo estándar extensible para Internet usado para acceder a servicios de directorios. Hablar de un servicio de directorio hace que sea más fácil olvidar que LDAP es un protocolo, y es muy común referirse a él como servidor o árbol LDAP.

Es importante entender que LDAP es un protocolo orientado a mensaje,[26] es decir un cliente construye un mensaje que contiene una petición y la envía al servidor. El servidor procesa la petición y envía el resultado de vuelta al cliente con uno o más mensajes LDAP. Por ejemplo, cuando un cliente LDAP busca en el directorio una entrada específica, le envía una petición de búsqueda LDAP al servidor. Este mensaje contiene un identificador de mensaje único (ID), generado por el cliente. El servidor recupera la entrada desde la base de datos y la envía al cliente en un mensaje LDAP. Todas las respuestas desde el servidor para el cliente son identificadas por el ID de mensaje proporcionado en la petición original del cliente.

Si el cliente busca en el directorio y encuentra múltiples entradas, éstas son enviadas al cliente en una serie de mensajes LDAP, una por cada entrada. La figura 3.1 muestra la interacción entre un cliente y un servidor LDAP.

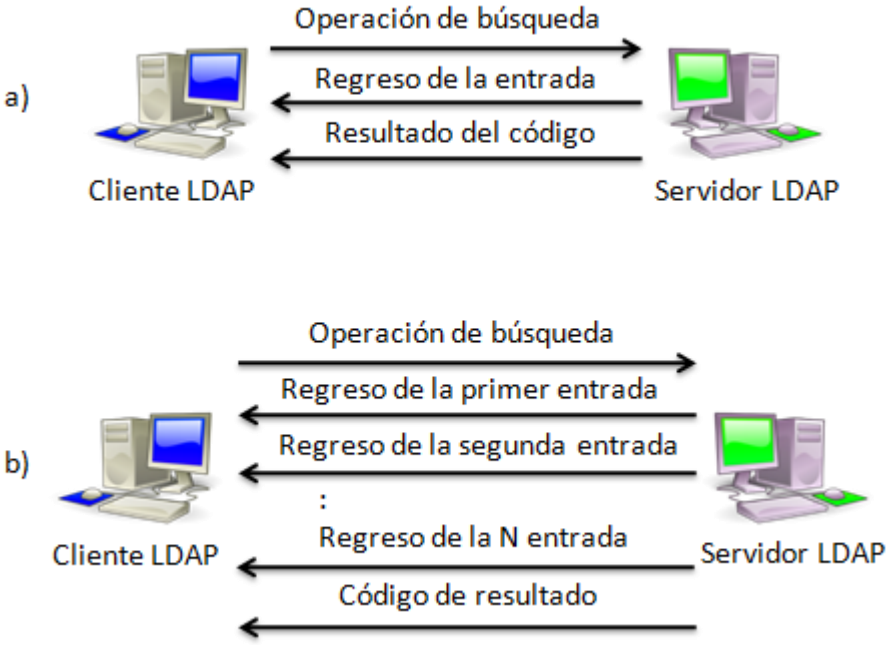

Figura 3.1 a) Interacción cliente-servidor LDAP para una entrada específica, b) Interacción cliente-servidor LDAP para entradas múltiples

La interacción que se lleva a cabo entre el cliente y el servidor LDAP, se realiza mediante un conjunto de funciones y procedimientos o APIs, las cuales permiten de manera general y estándar la correcta comunicación entre el cliente y el servidor.

3.1.1 APIs LDAP

Una API es un conjunto general de procedimientos o métodos que ofrecen la capacidad de comunicación entre componentes de software. Generalmente son usadas como bibliotecas o librerías.

Entre las funciones más útiles para llevar a cabo una petición LDAP mediante una interfaz de programación o ejecutada directamente sobre línea de comandos, son las que se muestran en la tabla 3.1:

Tabla 3.1 Principales funciones LDAP más utilizadas para comunicación entre cliente y servidor LDAP

FUNCION	DESCRIPCIÓN
ldap_search	Busca una entrada en el directorio
ldap_compare	Verifica si una entrada coincide con el valor del atributo proporcionado
ldap_bind	Autentica sobre el servicio de directorio (prueba de credenciales)

ldap_unbind	Finaliza una sesión LDAP
ldap_modify	Efectúa cambios en una entrada de directorio existente
ldap_add	Agrega una nueva entrada al directorio
ldap_delete	Elimina una entrada de directorio existente
ldap_rename	Renombra una entrada de directorio existente
ldap_result	Recupera los resultados de una operación previa

Independientemente del sistema operativo o lenguaje de programación utilizado, las funciones que se describen en la tabla T.1 se encuentran otras APIs disponibles por ejemplo Net::LDAP o PerLDAP (para perl), python-ldap (para python), JNDI (para java), ADSI (para Visual Basic, C y C++) y php5-ldap (para php). También por ejemplo, para el sistema operativo Linux se encuentra un paquete (ldap_utils) que contiene todas las utilidades necesarias, incluyendo las funciones mostradas en la tabla anterior para acceder mediante línea de comandos a un servidor LDAP.

3.2 Árbol de directorio LDAP

LDAP define cuatro modelos básicos que describen completamente cómo opera, qué datos pueden ser almacenados en el directorio LDAP y lo que se puede hacer con esos datos. [27]

3.2.1 Modelo de información

Este modelo define el tipo de dato y la unidad básica de información que se almacena en el directorio, es decir, la construcción de bloques que se pueden usar para crear el directorio.

La unidad básica de información en el directorio es la entrada (*entry*), que es una colección de información sobre un objeto. A menudo la información de estas entradas describen un objeto del mundo real, tal es el caso de una persona. En un directorio común se encontrarían miles de entradas que corresponden a gente, departamentos, servidores, impresoras y otro tipo de objetos cotidianos dentro de una organización. La figura 3.2 muestra una parte común del árbol de un directorio.

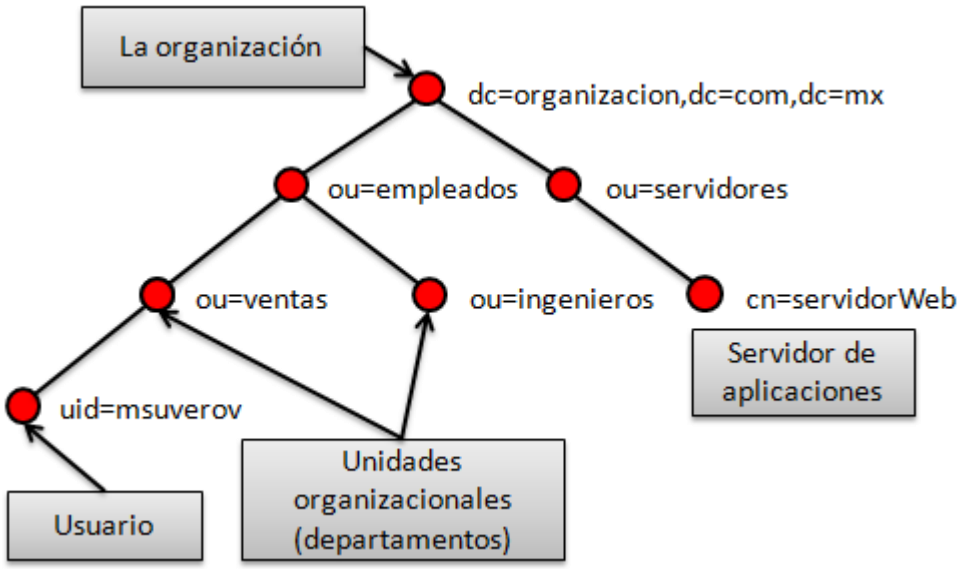

Figura 3.2 Porción típica de un directorio con objetos comunes dentro de una organización

Cada entrada de directorio tiene un único DN o *distinguished name,* que es la forma en la que se hace referencia a las entradas de manera completa. Hasta este punto, cabe señalar que el DN es un atributo propio de LDAP y el más importante entre muchos otros, algunos de ellos se enlistan en la tabla 3.2, ya que serán de ayuda para conocer el significado y la referencia a la que se hace.

Tabla 3.2 Atributos más utilizados por LDAP para la generación y recepción de entradas

ATRIBUTO	DESCRIPCIÓN
DN	Nombre completo o distinguido (**D**istinguised **N**ame), hace referencia a una entrada LDAP completa
dc	Componente de dominio (**d**omain **c**omponent), indica la raíz del árbol LDAP
uid	Id de usuario (**u**ser **id**), éste es el identificador único y obligatorio
cn	Nombre común (**c**ommon **n**ame), éste es el nombre de la persona
givenName	Éste es el nombre de pila de la persona
sn	Apellido (**s**ur**n**ame),éste es el apellido de la persona
o	Organización (**o**rganization), éste es el nombre de la compañía
ou	Unidad organizacional (**o**rganizational **u**nit), éste es la unidad organizacional, grupo o departamento
mail	Éste es el correo electrónico de la persona

Para referirse a un objeto de un directorio, se requiere de una entrada que se forma concatenando una serie de atributos individuales, emparentados a un determinado origen, es decir la ruta desde donde se encuentra el elemento deseado y yendo en retroceso hasta el origen. Por ejemplo basándose en el árbol de la figura 3.3 se obtendrá la entrada LDAP para el nombre de usuario “msuverov”.

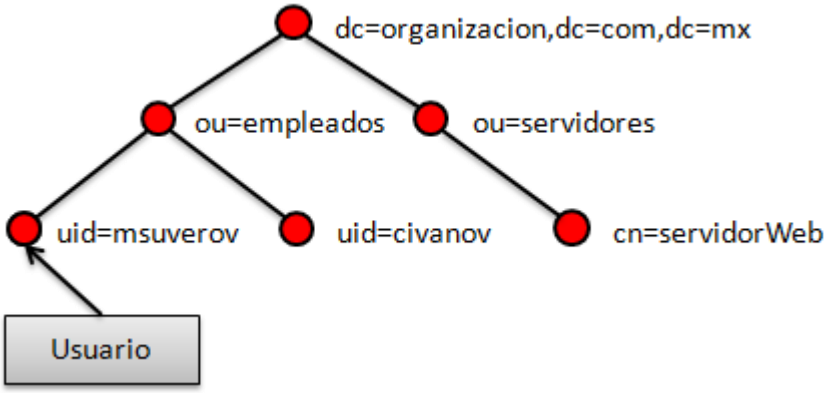

Figura 3.3 Pequeña porción de un árbol LDAP dentro de una organización

La entrada queda de la siguiente forma:

uid=msuverov,ou=empleados,dc=organizacion,dc=com,dc=mx

Esta manera de expresar las entradas en LDAP, es una forma estándar de representar un dato del directorio en un formato textual, y se utiliza cuando se requiere efectuar una operación sobre el directorio, los datos son enviados al servidor mediante línea de comandos o una API.

Cualquier entrada en el directorio, contiene un conjunto de atributos que son requeridos, y otro conjunto de atributos que son permitidos. Por ejemplo, una entrada para describir a una persona, se requiere tener el atributo “cn” y el atributo “sn”, pero para el caso de querer obtener un dato más específico de esa persona, por ejemplo el atributo “mobile” (que hace referencia al teléfono móvil), puede que este atributo esté restringido y no se permita la consulta, de igual forma pudiera ser que el atributo no esté definido.

Algunos atributos no se definen de manera predeterminada sólo los requeridos para poder describir a un objeto LDAP. Los atributos adicionales se tienen que definir manualmente en la configuración del servicio de directorio.

3.2.2 Modelo de nombrado

El modelo de nombrado define cómo se organizará y se hará referencia a los datos, describirá el tipo de estructura que se puede construir con los atributos individuales. También indica cómo referirse a cualquier entrada particular del directorio dentro de la estructura.

La flexibilidad que ofrece LDAP en este modelo, permite colocar los datos en una manera que sea mucho más fácil administrar, de este modo dentro del árbol LDAP se puede establecer una rama que describa a los empleados, y otra que describa a los grupos o departamentos dentro de la organización, de tal modo que se vea reflejada la administración geográfica y jerárquica del lugar donde se encuentra establecido el lugar de trabajo.

De manera análoga, esta estructura se puede comparar con la que tiene el sistema de archivos en un sistema operativo. El esquema de archivos tiene un conjunto de archivos y carpetas referidos a una estructura jerárquica. La figura 3.4 muestra la similitud de la estructura que sigue el sistema de archivos de un sistema operativo y la de un árbol LDAP.

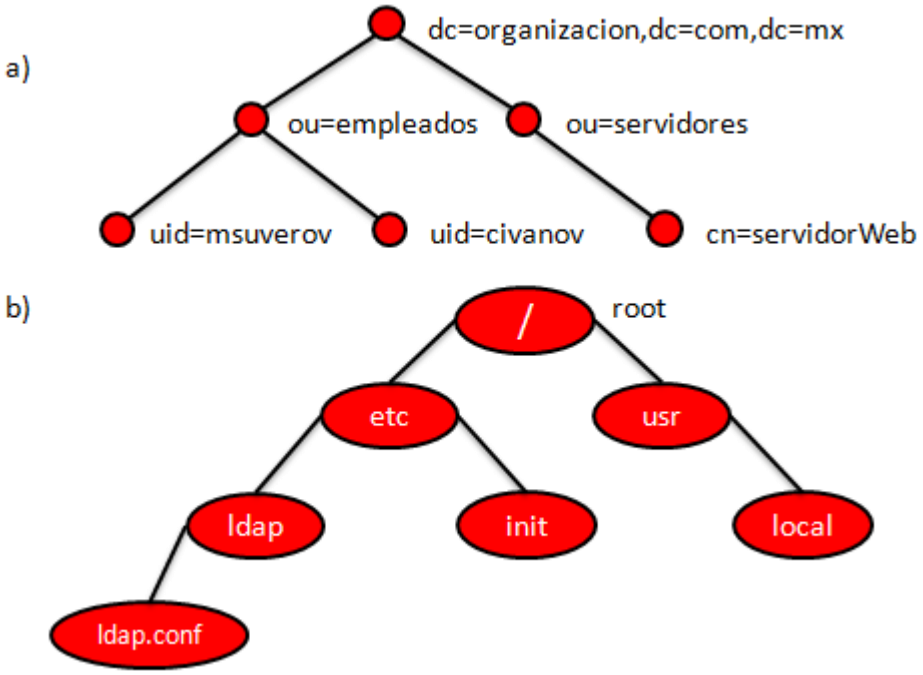

Figura 3.4 a) Estructura básica de un árbol LDAP, b) Pequeña porción de la Estructura de un sistema de archivos Linux

Hay tres diferencias principales entre los esquemas mostrados en la figura 3.4. La primera de ellas es que no hay una raíz (root) en el modelo LDAP, ya que en el sistema de archivos sí lo tiene, y por supuesto, es la carpeta raíz (la carpeta padre) común de todos los archivos y carpetas en la jerarquía del sistema de archivos. Por otro lado en la jerarquía del

directorio LDAP la entrada raíz es una entrada especial, que contiene la información de configuración del servicio de directorio, y normalmente no se usa para almacenar información.

La segunda diferencia es que en el directorio LDAP, cada nodo contiene datos, y cada nodo puede ser un contenedor. Esto es que cualquier entrada LDAP puede tener nodos hijos dentro de ésta y contener información al mismo tiempo. En contraparte en el sistema de archivos un nodo puede ser un archivo o una carpeta más no ambos. Sólo una carpeta puede tener hijos y ser un contenedor, y sólo los archivos pueden tener información. La figura 3.5 muestra un pequeño ejemplo jerárquico de un directorio LDAP con datos más a detalle.

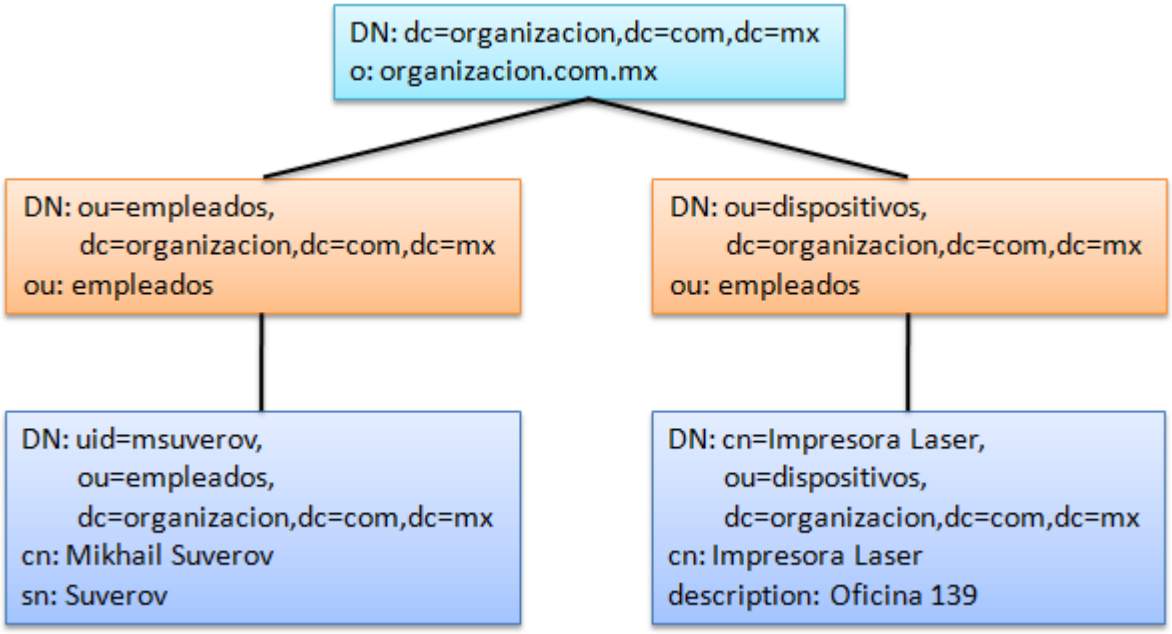

Figura 3.5 Porción de árbol LDAP, cada nodo puede tener datos y puede ser contenedor

La tercer diferencia, es la manera en cómo se ordenan de manera jerárquica un sistema de archivos y un sistema de directorio y el nombramiento de cada uno de los nodos. En base al árbol del sistema de archivos de la figura 3.4b y a la figura 3.5, para hacer referencia al archivo de configuración “ldap.conf” se hace de la siguiente manera:

/etc/ldap/ldap.conf

Y para referir al elemento “msuverov” se nombra de esta otra forma:

uid=msuverov,ou=empleados,dc=organizacion,dc=com,dc=mx

Cabe recalcar que para nombrar al archivo “ldap.conf” y leyendo de izquierda a derecha, corresponde a leer el árbol de la figura 3.4b de arriba abajo; para nombrar al elemento “msuverov” y leyendo de la misma forma, se recorre el árbol de la figura 3.4a de abajo hacia arriba.

LDAP soporta ordenamiento jerárquico de entradas de directorio, y no se establece ningún tipo de jerarquía en particular. Así como se tiene la libertad de ordenar un conjunto de archivos en un SO de modo que se facilite la búsqueda de un archivo determinado y sea más fácil de administrar, también se tiene la libertad de construir cualquier tipo de jerarquía de directorio que se desee. Claro está, que algunas estructuras de directorio pueden ser mejores unas que otras, dependiendo de la situación.

Cualquier entrada DN, el componente de más a la izquierda es conocido como el RDN o *relative distinguished name*, que por ejemplo, en un intento de agregar un par de entradas al directorio, éstas pueden quererse colocar sobre el mismo nodo padre, estos RDN deben ser únicos para evitar ambigüedades, ya que si al ingresar una segunda entrada con el mismo nombre que la primera, el servicio de directorio busca recursivamente y encontrará que ya se ha ingresado anteriormente la esa entrada, por tanto será rechazada por el servidor.

3.2.3 Modelo funcional

Existen dos operaciones interrogantes que permiten a los clientes LDAP buscar en el directorio y recuperar un dato. La operación de búsqueda permite a un cliente encontrar entradas en el directorio, y la operación de comparación permite al cliente probar si una entrada contiene un valor de entrada en particular.

La operación de búsqueda se utiliza para buscar entradas en el directorio y recuperar de manera individual un dato. La operación de lectura no existe en LDAP.[28] Cuando se requiere leer una entrada particular en el directorio, se debe llevar a cabo mediante una operación de búsqueda en la que se restrinja solo para esa entrada.

Al efectuar la búsqueda se debe de definir el alcance de la misma. Hay tres maneras de definir el alcance, la primera que es “sub” (subárbol), es indicando que se desea buscar sobre el subárbol desde el objeto base incluyendo las hojas, la segunda forma “onelevel” (un nivel), es indicando que se desea buscar sólo en el nivel inmediato de la entrada especificada. Por último la tercera forma “base”, se indica que se quiere buscar solo en el objeto base especificado. La figura 3.6 ilustra estos tres tipos de búsqueda.

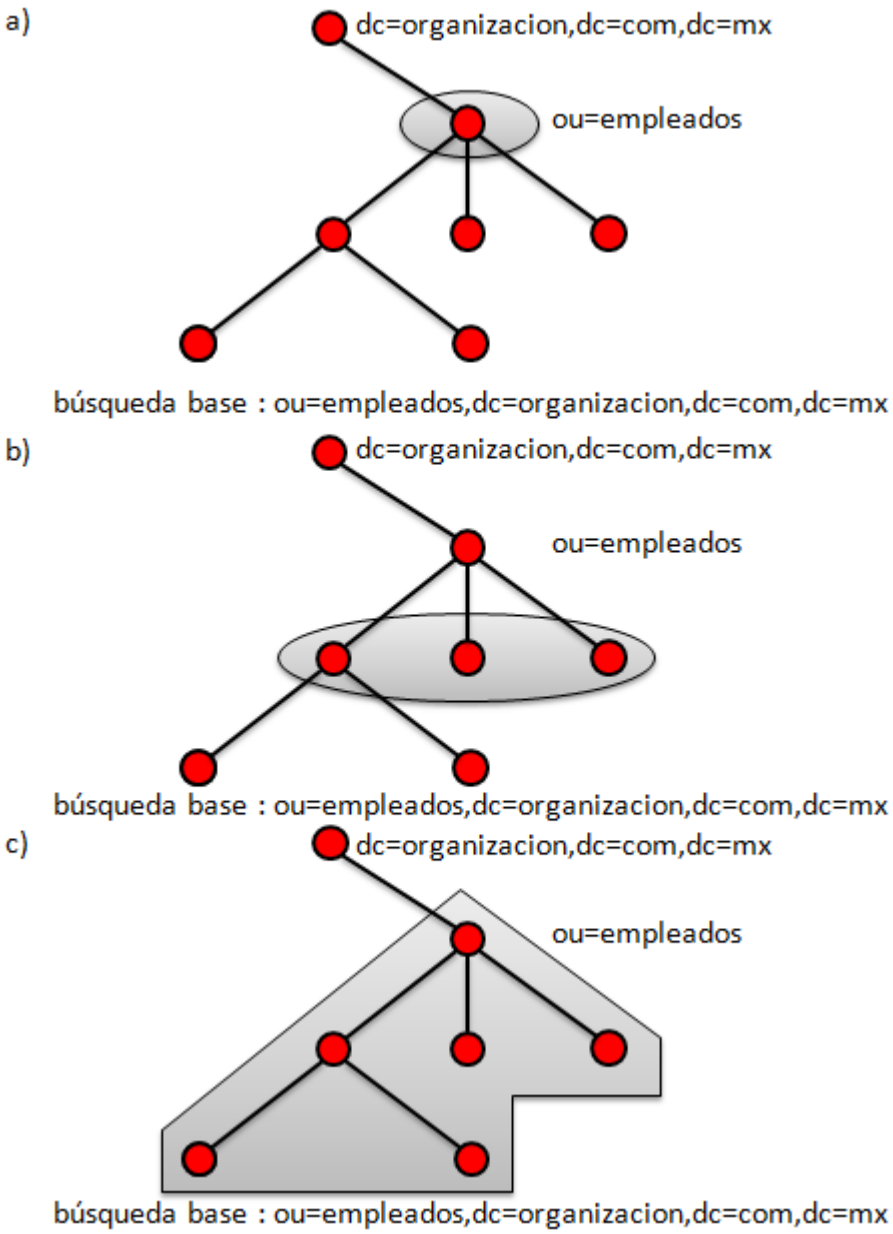

Figura 3.6 a) búsqueda base, b) búsqueda un nivel, c) búsqueda subárbol

Al final de cada búsqueda una lista de atributos pueden ser regresados por cada correspondencia encontrada.

NOTA: *Si se efectúa una búsqueda desde línea de comandos, mediante el comando "ldapsearch" solo es necesario proporcionar los atributos uid, cn y dc para llevar a cabo la búsqueda, ya que el comando utiliza estos atributos de manera predeterminada y obligatoria.*

3.2.4 Modelo de seguridad

El propósito del modelo de seguridad, consiste en proteger lo que hasta ahora se ha mencionado en los modelos anteriores lo que concierne a entradas de directorio, búsquedas y actualizaciones sobre el directorio.

El modelo de seguridad, depende del hecho de que LDAP es un protocolo orientado a conexión, esto es que un cliente LDAP abre una conexión hacia un servidor LDAP y lleva a cabo diversas operaciones de protocolo sobre la misma conexión. Todo lo que resta es proporcionar un marco de seguridad, para proteger la información en el directorio del acceso no autorizado.

Un cliente LDAP se puede autenticar para ingresar al servicio de directorio (o servidor LDAP) en algún momento durante la conexión, momento en el cual se podrán conceder privilegios adicionales o sólo permitir determinados privilegios. De esta manera se inicia un marco de seguridad sobre un servicio de directorio mediante la autenticación.

3.3 LDAP como sistema de autenticación

El proceso de autenticación sobre un directorio LDAP se conoce como enlazado o ligado (mejor conocido como *binding)*. Hay varios métodos de enlazado.

En un enlace simple (*simple bind*) de autenticación, un cliente LDAP proporciona al servidor un DN y una contraseña, los cuales se envían en texto claro al servidor (sin ninguna clase de cifrado). El servidor busca la entrada en el directorio, que corresponda al DN proporcionado por el cliente y lo verifica, si la contraseña coincide con el valor almacenado en el atributo *userPassword* el cliente se ha autenticado correctamente; si no, la operación de autenticación ha fallado y se notifica mediante un código de error al cliente.

En la forma *simple bind*, la contraseña se envía a través de la red en claro. Sin embargo, se puede proteger en contra de un ataque de "*Man-in-the middle*", para evitar que la contraseña sea interceptada y leída fácilmente.

Una manera inmediata de probar si una entrada existe dentro del directorio LDAP es mediante línea de comandos. Al utilizar el comando "ldapsearch" que viene integrado en el paquete ldap_utils para el sistema operativo Linux, permite realizar una búsqueda sobre el directorio y obtener determinados parámetros, claro que para ello se necesita autenticarse mediante contraseña del usuario que se desea consultar. A continuación se muestra un ejemplo de la ejecución del comando "ldapsearch" y los atributos que son regresados:

```
1. ldapsearch -W -D
   "uid=msuverov,ou=empleados,dc=organizacion,dc=com,dc=mx" -s sub
   "(cn=Mikhail Suverov)" -h localhost
2. Enter LDAP Password:
3. # extended LDIF
4. # LDAPv3
5. # base <dc=organizacion,dc=com,dc=mx> (default) with scope
   subtree
6. # filter: (cn=Mikhail Suverov)
7. # requesting: ALL
8. # msuverov, empleados, organizacion.com.mx
9. dn: uid=msuverov,ou=empleados,dc=organizacion,dc=com,dc=mx
10.    objectClass: posixAccount
11.    objectClass: inetOrgPerson
12.    objectClass: organizationalPerson
13.    objectClass: person
14.    homeDirectory: /home/msuverov
15.    loginShell: /bin/bash
16.    uid: msuverov
17.    cn: Mikhail Suverov
18.    uidNumber: 10009
19.    gidNumber: 10001
20.    userPassword:: e01ENX1oeVpVSEQyTU1CZkR2MXczRVpqZjRRPT0=
21.    description: Honorarios
22.    sn: Suverov
23.    givenName: Mikhail
24.    mail: msuverov@organizacion.com.mx
25.    physicalDeliveryOfficeName: Ventas
26.    # search result
27.    search: 2
28.    result: 0 Success
29.    # numResponses: 2
30.    # numEntries: 1
```

En este ejemplo se ha realizado una búsqueda de tipo sub (subárbol) utilizando únicamente la entrada de usuario “uid=msuverov”, estableciendo a su vez un filtro de búsqueda con “cn=Mikhail Suverov”, al ejecutar el comando se obtiene esta salida.

En la línea 2 del resultado obtenido, se observa que el servidor está solicitando una contraseña para poder realizar la búsqueda, al ser válida la contraseña significa que la autenticación tuvo éxito, y devuelve como respuesta la búsqueda solicitada. De otro modo si hubiese fallado se devuelve el siguiente código de error:

```
ldap_bind: Invalid credentials (49)
```

Los atributos resaltados en rojo, pueden ser de importancia al momento de querer obtener información detallada acerca de la persona. Nótese que algunos de esos atributos, corresponden a los que se encuentran en la tabla T2.

3.3.1 LDAP en modo seguro

LDAP puede llevar a cabo una negociación entre el cliente y el servidor LDAP, sobre una capa de transporte cifrada previa a cualquier operación de ligado (*bind*). De este modo toda la información se mantiene segura así como cualquier otra información transmitida durante la sesión.

Mediante un *simple bind* LDAP utiliza de manera predeterminada el puerto tcp/389 para una negociación no cifrada entre el cliente y el servicio de directorio (servidor LDAP). Para una negociación cifrada utilizando el mismo puerto, se utiliza una extensión del protocolo TLS llamado StartTLS. StartTLS trabaja sobre la versión más reciente del protocolo LDAP (LDAPv3).

Otra forma para establecer una negociación segura entre un cliente y el servidor LDAP, es mediante un túnel SSL, para ello se requiere utilizar el puerto 636 (LDAPS – tcp/636). Actualmente su uso es muy frecuente, pero esta forma se encuentra obsoleta junto con la versión 2 del protocolo LDAP (LDAPv2), nunca se estandarizó; quedando a favor la extensión StartTLS dentro del estándar RFC2830 de la versión 3 del protocolo LDAP.[29] El modelo de una negociación segura entre un cliente y el servidor LDAP se ilustra en la figura 3.7.

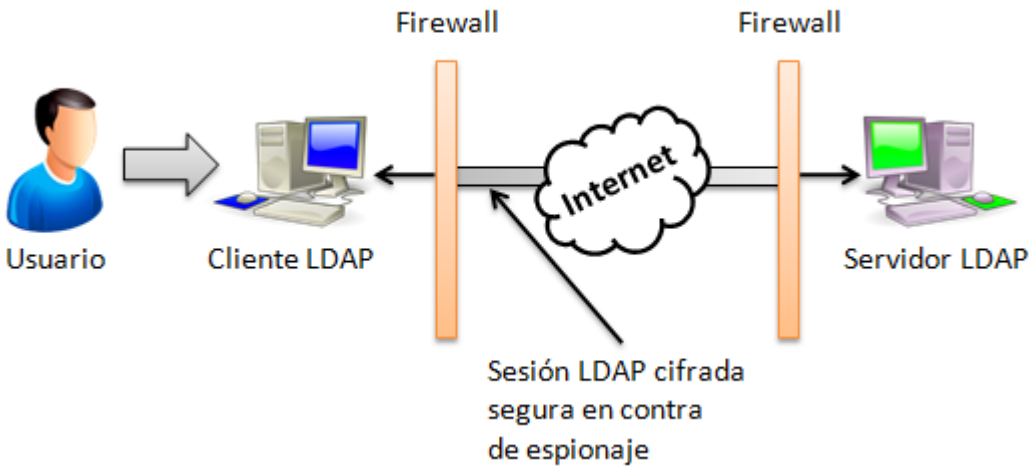

Figura 3.7 TLS permite transmisión segura de los datos del directorio sobre Internet

El inicio de la conexión entre el cliente y el servidor LDAP se efectúa en tres pasos:

Paso 1. Se abre una conexión TCP hacia el servidor.

Paso 2. Se envía la operación de StartTLS extendido. Bajo la capa de estos protocolos se preparan el cifrado y la autenticación de acuerdo a la especificación TLS.

Paso 3. Se enlaza utilizando el mecanismo externo *Secure Autentication and Secure Layer* (SASL) si un certificado fue proporcionado durante la negociación TLS, o usando otro mecanismo SASL, por ejemplo DIGEST-MD5.

Después de que TLS se ha establecido y la operación de ligado (*bind*) se ha realizado, el cliente dispone de una conexión segura y autenticada en el directorio.

3.4 Compatibilidad de los servicios Web con LDAP

Los métodos de autenticación por sí solos requieren de algún otro servicio o software adicional, para lograr el efecto de autenticación centralizada. Al combinar estas herramientas se involucra una situación de compatibilidad, que conlleva a una evaluación de costo-beneficio, y a la vez de un diseño y desarrollo.

La subdirección de seguridad de la información (SSI) UNAM-CERT, que pertenece a la Dirección General de Cómputo y de Tecnologías de Información y Comunicación (DGTIC), cuenta ya con un servicio de directorio activo que es OpenLDAP.

OpenLDAP es una implementación del protocolo LDAP de código abierto que permite utilizar diferentes esquemas o repositorios de datos.

Comúnmente OpenLDAP se utiliza para almacenar la información de usuarios, grupos y recursos de una organización mediante atributos los cuales están definidos en clases de objeto que vienen junto con el software.

Resulta muy sencillo añadir o quitar atributos a los objetos de una clase ya que la arquitectura de OpenLDAP es altamente escalable, soportando gran cantidad de registros en el directorio, además de que posee un gran rendimiento en la búsqueda y lectura de entradas.

Tomando en cuenta la implementación del protocolo LDAP con información de usuarios, grupos y contraseñas, un sistema Web por sí mismo se encuentra aislado quedando fuera del alcance de LDAP, ya que cuenta con su sistema operativo y base de datos configurados para funcionar con su propio agente o mecanismo de autenticación.

Para enlazar una aplicación Web existente en la SSI con un directorio LDAP, es necesario hacer uso de librerías o APIs, que en la actualidad existen varias soluciones gratuitas de código

abierto, para casi la mayor parte de los lenguajes de programación y compatibles con el protocolo LDAP, de esta manera no es necesario reescribir la aplicación Web, ya que sólo lo que se requiere es conectarla a un directorio LDAP para centralizar el acceso.

3.5 Agente de autenticación

En computación un agente describe una idea o concepto de una abstracción de software, similar a las funciones o métodos que se utilizan al momento de programar. El concepto de agente provee una forma conveniente para describir una compleja entidad de software, que es capaz de actuar con cierto grado de autonomía, para cumplir tareas en representación de personas.

Llevando este concepto a la vida cotidiana, supóngase un agente de tránsito (un policía) ubicado en la esquina de una avenida, éste tiene órdenes de detener y multar a todo vehículo que infrinja una de las normas establecidas en el reglamento de tránsito. El agente actúa de manera autónoma, y es capaz de observar, detectar y actuar en cualquier infracción al reglamento de tránsito, por decirlo de alguna manera el agente está llevando a cabo sus labores de manera persistente.

Pero ¿qué pasa cuando se le dan órdenes más estrictas?, es decir, que su superior le indica que “por cada automóvil que pase por la avenida en la que se encuentra, tiene la obligación de detenerlo y revisar que su documentación de circulación esté en orden y en regla, de no ser así, deberá infraccionarlo y arrestarlo por 24 horas”. De manera que ahora el agente de tránsito está actuando de forma más específica y estricta en un escenario determinado, ignorando en cierto modo lo que pase fuera de este entorno, ya que tiene explícitas instrucciones para actuar de determinada manera, sin importar por ejemplo que 50 metros más adelante se haya estacionado un vehículo en zona de no estacionarse, otro agente de tránsito podría atender esta acción.

Regresando al área de computación, los agentes de autenticación que se encuentran originalmente en los sistemas web de la subdirección, estrictamente hablando no son agentes, ya que no son autónomos, ni inteligentes, ni están siendo ejecutados de manera persistente, solo son invocados cada vez que un usuario intenta acceder al sistema, pero tienen una característica que permite solamente llamarlos agentes, la cual es realizar una tarea en sustitución de una persona, además que para su identificación dentro del sistema fueron nombrados aludiendo al nombre de agente.

Por esas razones, de aquí en adelante se hará referencia a estas funciones y métodos como agentes. Éstos operan de la siguiente manera:

- Al ingresar un cliente al sistema web, éste le solicita sus credenciales de acceso. Las credenciales son enviadas a un agente de filtrado llamado Authwrap (ejecutable escrito en C).
- El agente Authwrap se encarga de revisar las credenciales de acceso que estén libres de caracteres dañinos, por ejemplo asteriscos, comillas dobles o simples, punto y coma, etc.
- Una vez que las credenciales de acceso fueron correctamente revisadas, se envían a otro agente de concesión de acceso llamado Authagent (escrito en Perl), que se encarga de generar un identificador de sesión y un ticket de acceso, además de registrar el acceso del usuario en la base de datos.
- El ticket de acceso es enviado al servidor y utilizado para conceder el acceso al usuario.
- Una vez autenticado el usuario, cada vez que éste acceda a funciones dentro del sistema, el agente de gestionamiento Authagent (escrito en PHP), está encargado de gestionar la sesión, además de verificar el estado y la validez del ticket de acceso, para impedir un robo de sesión o un acceso no autorizado. En el transcurso de este proceso, donde no se cumplan las condiciones de las reglas implementadas en los agentes, se desiste del proceso de autenticación. La figura 3.8 ilustra la interacción entre los agentes.

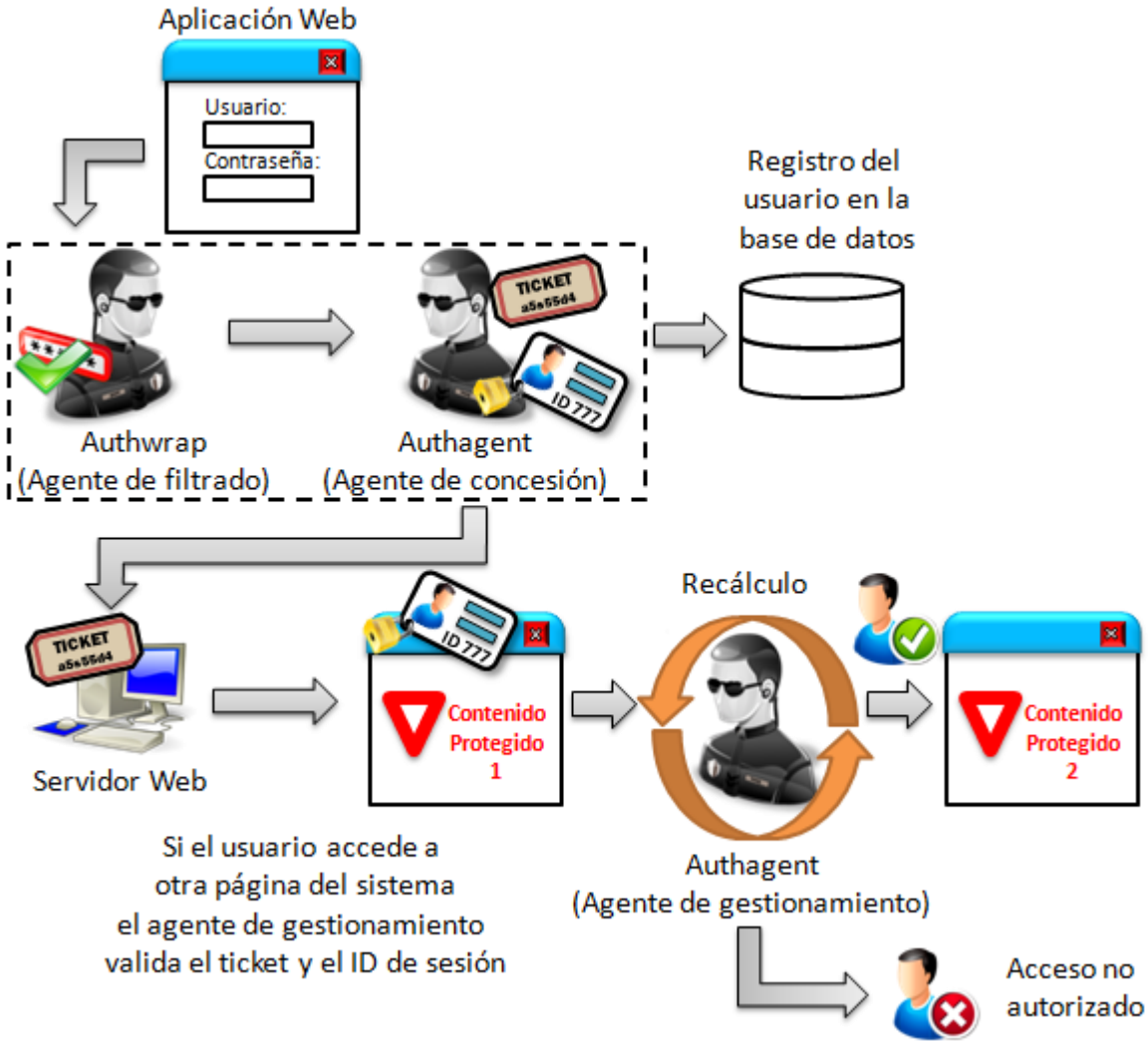

Figura 3.8 Interacción de los agentes de autenticación en los servicios web de la SSI

El recuadro punteado que se encuentra en la figura 3.8, corresponde a lo que se sustituyó en el sistema de autenticación, que consiste en el remplazo de dos agentes por un módulo de autenticación centralizada.

3.6 Diseño y desarrollo del módulo de autenticación

El desarrollo del módulo de autenticación, se centró inicialmente sobre el sistema web del portal de la SSI UNAM-CERT, basándose principalmente en las funciones que realizan los agentes Authwrap (agente de filtrado) y Authagent (agente de concesión), agregando el factor principal que es poder autenticar usuarios de manera centralizada, complementar la funcionalidad de los agentes antes mencionados, y ajustar este módulo de autenticación a las políticas de seguridad que establece la SSI UNAM-CERT.

¿Qué es un módulo?, un módulo es un componente funcional del software, el cual se define siguiendo ciertos criterios, de tal modo que se reduzca la complejidad del programa y facilite su mantenimiento.[30]

Partiendo de las características que debe tener un módulo de software, el diseño del módulo de autenticación se basa en el reciclaje de código fuente de los agentes que utilizaba el sistema, y una petición de autenticación a un servidor LDAP mediante la API PHP-LDAP, esta petición se realiza mediante una función programada en PHP. Esta función es el tronco principal del módulo. La figura 3.9 muestra un esquema general de las fases del diseño del módulo de autenticación.

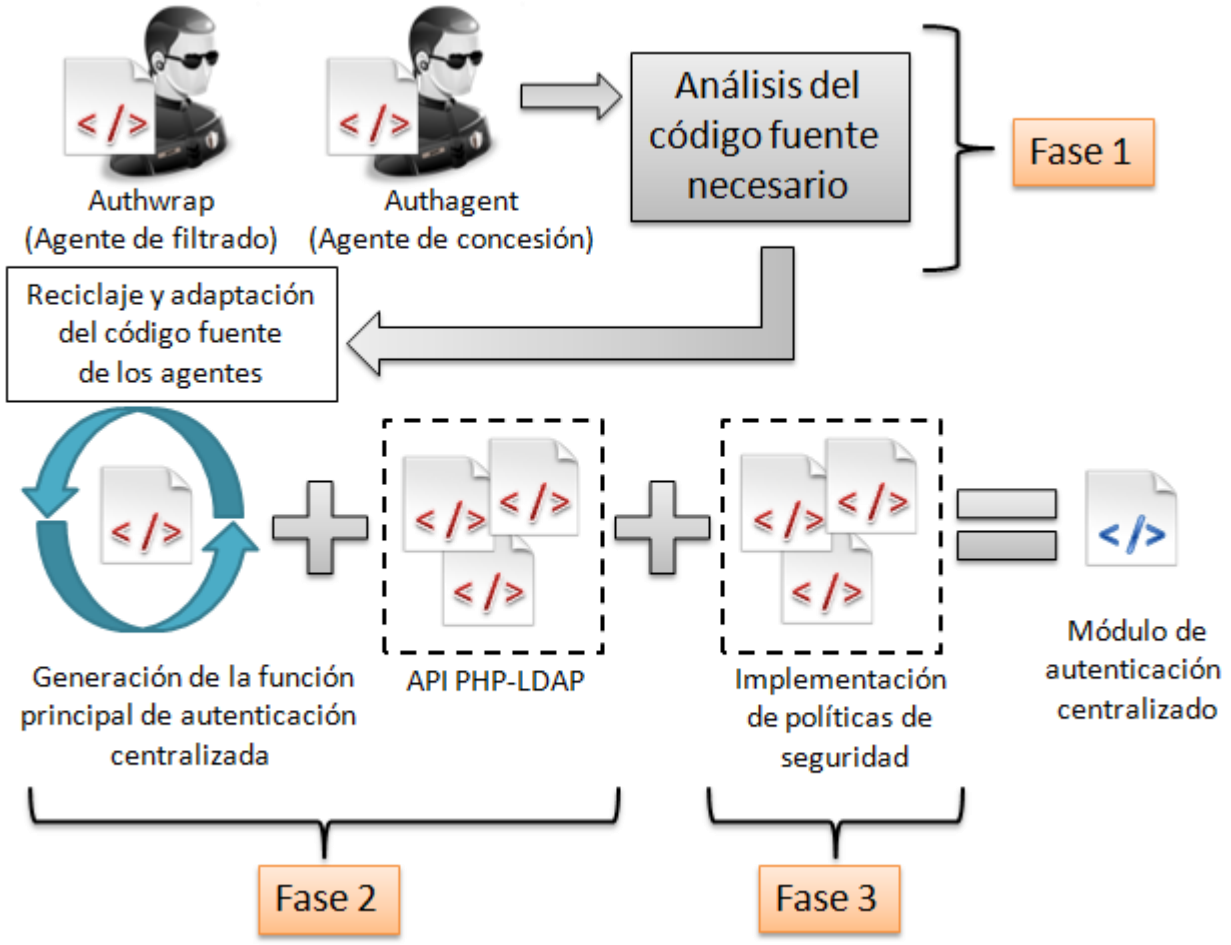

Figura 3.9 Fases de diseño del módulo de autenticación

En la fase 1, consistió en revisar el código fuente de los agentes de autenticación, conocer y entender su funcionamiento, una vez cumplido esto se procedió a realizar un reciclaje de código fuente, extrayendo las reglas fundamentales en el proceso de autenticación. El primer

problema a resolver era en que los agentes de filtrado y de concesión están escritos en dos lenguajes de programación diferente, en C y Perl, respectivamente. Se tenía que desarrollar un sólo código fuente escrito en PHP, unificando en él las tareas que efectuaban ambos agentes.

Otra de las problemáticas a solucionar, era que si se deseaba migrar el sistema a otro servidor, se tenía que volver a compilar el agente de filtrado en el nuevo servidor y para que funcionara el agente de concesión, se tenían que instalar Perl y varias librerías dependientes de Perl, de este modo se opta completamente depender de PHP y la API PHP-LDAP y un sólo módulo de autenticación, y así remplazar la labor de estos dos agentes.

Cabe aclarar que las acciones que realizaban los agentes ahora las hace uno sólo, pero este nuevo "agente" va inmerso o embebido dentro del módulo de autenticación centralizada. Por las razones antes ya mencionadas de que los estos agentes no son estrictamente agentes, por ello es más apropiado referirlo como módulo, ya que posee varias características correspondientes a un componente de software (módulo), como son la independencia funcional, la cohesión, y el acoplamiento.

NOTA: Por razones de políticas de seguridad de la SSI, no es posible mostrar el nombre de los archivos, ni el código fuente de los agentes o del módulo de autenticación en su totalidad, sólo partes que no representen riesgos de seguridad serán expuestas o en su defecto sólo mencionadas.

La fase 2 consistió, en reescribir el código fuente extraído de los agentes Authwrap y Authagent (agente de concesión), para su adaptación posterior sobre la función principal del módulo de autenticación.

La función principal del módulo consiste en la obtención de las credenciales de acceso del usuario y compararlas con las almacenadas en el servidor LDAP, este procedimiento básico se muestra en el código fuente de la tabla 3.3.

Tabla 3.3 Código fuente para una autenticación esencial sobre un servidor LDAP

CÓDIGO

```
1. <?php
2. $usuario = $_POST["usuario"];
3. $password = $_POST["pass"];
4. $usuario = filtraEntrada($usuario);
5. $host = '127.0.0.1';
6. $puerto = 389;
```

```
7.  $uid = 'uid='.$usuario.',';
8.  $ou = 'ou=grupo,';
9.  $dc = 'dc=organizacion,dc=com,dc=mx';
10. $dn = $uid.$ou.$dc;
11. $conexion = ldap_connect($host, $puerto);
12. ldap_set_option($conexion, LDAP_OPT_PROTOCOL_VERSION, 3);
13. $bind = ldap_bind($conexion, $dn, $password);
14. if ($bind){
15.     header ("Location: https://www.sistema.com.mx/contenidoProtegido1.php");
16. }
17. else{
18.     header ("Location: https://www.sistema.com.mx/index.php?error=denegado");
19.     exit();
20. }
21. ?>
```

El código fuente de la tabla 3.3 en las líneas 2 y 3 se obtienen las credenciales de acceso del usuario mediante el método post.

La línea 4 la función filtraEntrada() sustituye la función del agente de filtrado Authwrap, este agente era un ejecutable generado al compilar un código en C muy extenso, demasiado código para una función de filtrado de caracteres sobre una entrada de datos, el desglose de la nueva función que suple al agente Authwrap se expone en la tabla 3.4

Tabla 3.4 Función de filtrado de caracteres dañinos

CÓDIGO

```
1.  <?php
2.  function filtraEntrada($valor){
3.      $listanegra = array ("|","%","&","=","'","*","<",">","\\",";","\"","/","(",")");
4.      $valor = strip_tags ($valor);
5.      for ($i = 0; $i<count ($listanegra); $i++)
6.          $valor = str_replace ($listanegra[$i],"",$valor);
7.          return $valor;
8.  }
9.  ?>
```

Mediante una lista negra de caracteres se hace un barrido para detectar y remplazar aquellos caracteres que pueden ocasionar problema al momento de que el servidor procesa el login de usuario, con esto se evitan ataques de inyección SQL y de LDAP.

Continuando con el código de la tabla 3.3, en la línea 5 se asigna la dirección IP del servidor LDAP, en la línea 6 se asigna el número de puerto con el que trabaja el protocolo LDAP. Como ya se sabe que LDAP trabaja por los puertos 389 y 636, para una prueba de funcionamiento de este código es recomendable utilizar el puerto 389, y una vez que ya se haya podido establecer la comunicación entre el cliente y el servidor LDAP, al poner esto en un ambiente de producción, se debe usar el puerto 636 para cifrar toda la comunicación entre el cliente y el servidor LDAP.

Cabe destacar un detalle importante en estas líneas del programa, al utilizar el puerto 636 se debe realizar de la siguiente manera:

```
$host = 'ldaps://miservidorldap.com.mx';
$puerto = 636;
```

Con forme a la documentación de OpenLDAP[31] respecto a los parámetros que debe de llevar el archivo de configuración del cliente ldap (archivo ldap.conf), en la variable $host se debe asignar el valor de esa manera ya sea utilizando el nombre del servidor ldap o la dirección IP del mismo, pero antecediendo el prefijo “ldaps://”, ya que de otro modo no será posible la conexión al servidor ldap mediante el puerto 636.

La línea 7 consiste en crear el formato del uid (identificador único) de usuario y se asigna a la variable $uid, del mismo modo en la línea 8 y 9 para los parámetros ou (unidad organizacional) y dc (componente de dominio) respectivamente.

El parámetro ou comúnmente hace referencia a un nombre de grupo dentro del árbol LDAP, que de manera predeterminada se utiliza el nombre de “*users*”, el parámetro dc consiste en establecer el nombre del subárbol que se asignó al servidor LDAP, en este caso se estableció como **'dc=organizacion,dc=com,dc=mx'** esto debe ser ajustado según a las necesidades.

En la línea 10 se concatenan todos los parámetros, para generar la cadena completa de petición al servidor LDAP. La línea 11 se asigna una variable de conexión al servidor LDAP, en este punto es cuando se hace uso de una de las funciones contenidas en la API PHP-LDAP.

La función ldap_connect consiste en generar un identificador de conexión para ser utilizado más adelante en el enlazado al servidor LDAP, utiliza los valores del nombre del host y el número de puerto para realizar la conexión.

En la línea 12, se utiliza la función ldap_set_option para establecer la versión del protocolo LDAP a usar, que en este caso se hace uso de la versión 3 del protocolo.

La línea 13, establece el enlazado con el servidor LDAP, se utiliza la función ldap_bind y todos los parámetros asignados anteriormente, que son el identificador de conexión, el dn, y la contraseña del usuario. La función ldap_bind autentica contra el servidor LDAP tomando en cuenta los parámetros ya mencionados, y devuelve un valor booleano *true* en caso de éxito o *false* en caso de error.

En la línea 14 se establece la sentencia de control if utilizando para su decisión el valor devuelto por la función ldap_bind, que en caso de ser el valor booleano *true*, redirigirá a la página "*contenidoProtegido1.php*" (línea 15), en caso de ser el valor *false*, denegará el acceso redirigiendo a la página principal del sistema (al "*index.php*", línea 18), desplegando por supuesto un mensaje de error alusivo.

Un detalle importante sobre la línea 15 de la tabla 3.1, antes de la implementación del módulo de autenticación, se utilizaba un ID de sesión que viajaba a través de la URL, esta línea estaba de la siguiente manera:

```
header ("Location: https://www.sistema.com.mx/contenidoProtegido1.php?md5cookie=55e7dd3016ce4ac57b9a0f56af12f7c2");
```

Para evitar un posible robo de sesión, ha quedado este direccionamiento con la sintaxis mostrada en la línea 15 de la tabla 3.3, de este modo ya no se exhibe el ID de sesión en la URL del navegador.

Teniendo un script base de autenticación centralizada, se adaptaron las funciones que realizaban el agente de filtrado y el agente de concesión, consiguiendo el script de la tabla 3.5 con esa adaptación.

Tabla 3.5 Script con las funciones de la generación del ticket y el ID de sesión

CÓDIGO

```
1. <?php
2. $usuario = $_POST["usuario"];
3. $password = $_POST["pass"];
4. $usuario = filtraEntrada($usuario);
5. $host = '127.0.0.1';
6. $puerto = 389;
7. $uid = 'uid='.$usuario.',';
8. $ou = 'ou=grupo,';
```

```
9.  $dc = 'dc=organizacion,dc=com,dc=mx';
10. $dn = $uid.$ou.$dc;
11. $conexion = ldap_connect($host, $puerto);
12. ldap_set_option($conexion, LDAP_OPT_PROTOCOL_VERSION, 3);
13. $bind = ldap_bind($conexion, $dn, $password);
14. if ($bind){
15.     $_SESSION["key"] = creaLlave(8);
16.     $_SESSION["usuario"] = $usuario;
17.     $_SESSION["clavemd5"] = md5($password);
18.     $key = $_SESSION["key"];
19.     $_SESSION["ticket"] = generaTicket($usuario, $key);
20.     $_SESSION["sesion"] = generaSesion(md5($password), $key);
21.     header ("Location: https://www.sistema.com.mx/contenidoProtegido1.php");
22. }
23. else{
24.     header ("Location: https://www.sistema.com.mx/index.php?error=denegado");
25.     exit();
26. }
27. ?>
```

Las variables que se encuentran en las líneas 15, 16, 17, 18, 19 y 20 de la Tabla 3.5, corresponden a los valores necesarios para el recálculo del ticket y el ID de sesión que tomará el agente de gestionamiento para verificar estos parámetros.

Cabe recalcar que por cuestiones de seguridad las funciones expuestas en las líneas 19 y 20 de la tabla 3.5, no son las verdaderas que funcionan en el sistema de producción, y sirven únicamente para ilustrar la generación del ticket y el ID de sesión, del mismo modo los parámetros utilizados en dichas funciones. Las funciones creaLlave(), generaTicket() y generaSesion() sustituyen la labor del agente de concesión Authagent (escrito en Perl), dichas funciones se exponen en la tabla 3.6.

Tabla 3.6 Código de las funciones generaTicket() y generaSesion()

CÓDIGO

```
1.  <?php
2.  function generaTicket($usuario, $key){
3.      $ticket=md5($key.$usuario);
4.      return $ticket;
5.  }
6.  function generaSesion($password, $key){
7.      $IDsesion=md5($key.$password);
8.      return $IDsesion;
9.  }
```

```
10. function creaLlave($longitud){
11.     $list="abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789";
12.     $key="";
13.     $max=strlen($list)-1;
14.     for ($i=0; $i<$longitud; $i++)
15.         $key.=substr($list, rand(0, $max), 1);
16.     return $key;
17. }
18. ?>
```

La función generaTicket(), recibe 2 parámetros el login de usuario y la llave generada por la función creaLlave(), el parámetro $key se concatena con el parámetro $usuario. A la concatenación de estos parámetros, se les aplica una función hash (MD5) para generar una cadena cifrada y finita de 32 caracteres. Para todos y cada uno de los usuarios que ingresen al sistema se generará un ticket distinto.

La función generaSesion(), de igual forma recibe 2 parámetros. Para este caso se utiliza la contraseña del usuario y la misma llave generada de 8 caracteres (línea 15, tabla 3.5), el orden de concatenación es primero el parámetro $key seguido del parámetro $password. Al resultado de la concatenación se le aplica una función hash (MD5). Finalmente la cadena regresa a donde fue llamada la función.

La función creaLlave(), genera una cadena aleatoria de caracteres de acuerdo a lo que valga $longitud, que para ejemplificar este proceso de generación del ticket y el ID de sesión, se ha usado una longitud de 8 caracteres.

Por razones obvias de seguridad, no se exponen los verdaderos parámetros utilizados en estas funciones, pero las verdaderas siguen un algoritmo muy similar a las aquí expuestas, utilizando otros parámetros, concatenación y cifrado diferente.

Finalmente la fase 3, corresponde a la implementación de las políticas de seguridad. Estas políticas establecen parámetros que deben ser tomados en cuenta, al momento de acceder a todas y cada una de las aplicaciones web de la SSI.

Principalmente lo que se implementó en esta parte del módulo consiste en el bloqueo de la cuenta en un lapso de 3 intentos fallidos, un tiempo de bloqueo y registro en bitácoras de todos los accesos.

A partir del código de la tabla 3.5 se añaden las funciones que implementan las políticas de seguridad, todo el conjunto se muestra en el código de la tabla 3.7.

Tabla 3.7 Implementación de las políticas de seguridad y estructura final del módulo de autenticación centralizada

CÓDIGO

```
1.  <?php
2.  $usuario = $_POST["usuario"];
3.  $password = $_POST["pass"];
4.  $usuario = filtraEntrada($usuario);
5.  $host = '127.0.0.1';
6.  $puerto = 389;
7.  $uid = 'uid='.$usuario.',';
8.  $ou = 'ou=grupo,';
9.  $dc = 'dc=organizacion,dc=com,dc=mx';
10. $dn = $uid.$ou.$dc;
11. $conexion = ldap_connect($host, $puerto);
12. ldap_set_option($conexion, LDAP_OPT_PROTOCOL_VERSION, 3);
13. $bind = ldap_bind($conexion, $dn, $password);
14. if ($bind){
15.     $existeUsuario = consultaUsuario($usuario);
16.     if (!$existeUsuario){
17.         sincronizaUsuario($usuario);
18.     }
19.     $bloqueo = verificaBloqueo($usuario);
20.     if ($bloqueo){
21.         desbloqueaUsuario($usuario);
22.     }
23.     $_SESSION["key"] = creaLlave(8);
24.     $_SESSION["usuario"] = $usuario;
25.     $_SESSION["clavemd5"] = md5($password);
26.     $key = $_SESSION["key"];
27.     $_SESSION["ticket"] = generaTicket($usuario);
28.     $_SESSION["sesion"] = generaSesion($password);
29.     registraAcceso($usuario, true);
30.     header ("Location: https://www.sistema.com.mx/contenidoProtegido1.php");
31. }
32. else{
33.     bloqueaUsuario($usuario);
34.     registraAcceso($usuario, false);
35.     header ("Location: https://www.sistema.com.mx/index.php?error=denegado");
36.     exit();
37. }?>
```

De las líneas 15 a la 17 corresponden a un proceso adicional, que sincroniza la cuenta del usuario existente en LDAP sobre la base de datos del sistema local, esto es necesario para cuando un usuario ingresa al sitio por primera vez, también para el control de acceso al contenido del sistema, pero con un detalle importante, las contraseñas de los usuarios ya no se almacenan más en la base de datos del sistema local.

Las líneas 19, 21, 29, 33 y 34 indican las funciones que corresponden a la implementación de las políticas de seguridad de la SSI sobre el módulo de autenticación. Estas funciones por razones de seguridad su detalle queda reservado.

El código de la tabla 3.7, indica una estructura más completa y general sobre cómo está la lógica funcional del módulo de autenticación centralizada.

3.7 Implementación y preparando el escenario

Para la implementación del módulo de autenticación centralizada, fue necesario en cada una de las fases realizar pruebas aisladas y en un ambiente de prueba.

Con un servidor LDAP local (de pruebas), se verificó que el procedimiento de generación de ID's de sesión y tickets se llevara a cabo correctamente. Con una página web dinámica se hicieron pruebas para visualizar la generación del ticket y el ID de sesión, corroborando efectivamente estaban siendo generados.

Al tener estas pruebas realizadas, el script resultante que se llamará "*MAC.php*" (**M**ódulo de **A**utenticación **C**entralizada), se procedió a sustituir a los agentes de filtrado y concesión en un sistema de pruebas, el cual es una copia exacta del sistema que está en producción. De este modo se consigue observar el comportamiento que tendrá en el sistema real, y la ejecución de las funciones que implementan las políticas de seguridad de la SSI.

Al llevar a cabo la sustitución de los dos agentes por el "*MAC.php*", se ajustó la configuración del servidor web a este nuevo cambio, de igual forma las ligas rotas que dejaron los agentes al ser removidos fueron ajustadas.

El script "*MAC.php*" también tenía que interactuar con el agente de gestionamiento, éste tiene que recibir el ticket y el ID de sesión para realizar el recálculo de estos parámetros, validándolos una vez que un usuario se ha autenticado y navega dentro del contenido del

sistema. La figura 3.10 ilustra un esquema final con todos los elementos necesarios, para la autenticación centralizada de usuarios.

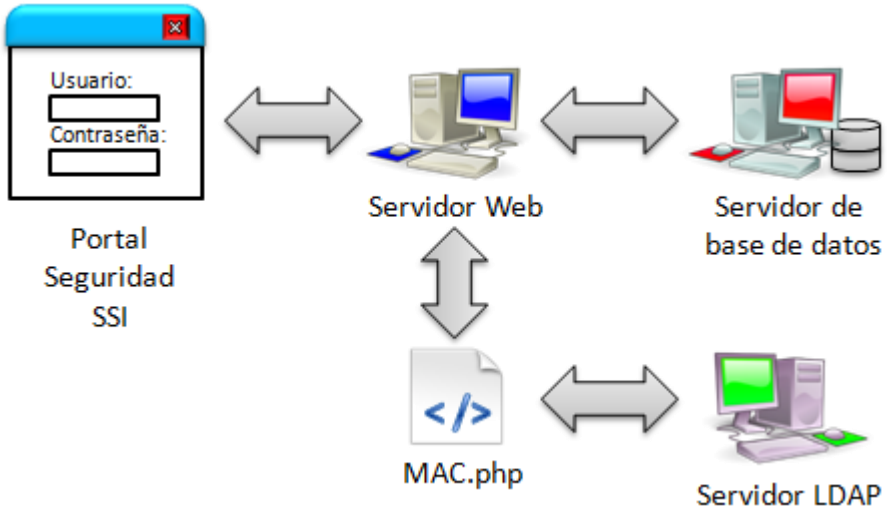

Figura 3.10 Esquema de los elementos que componen la autenticación centralizada de usuarios

Antes de realizar la prueba de autenticación en el sistema, se requiere de cuentas de usuario en el servidor LDAP local y una prueba de conexión con el servidor LDAP. El administrador de usuarios LDAP luce de la siguiente manera con un par de usuarios agregados, ver figura 3.11.

Figura 3.11 Administrador local de usuarios LDAP

El User ID corresponde a la cuenta del usuario, así como el nombre y apellido del mismo, el número de identificación del usuario (UID number) y el número de grupo al que pertenece (GID number). Desde esta pantalla es posible añadir, borrar y editar los datos de usuario.

La vista del árbol LDAP completo con el DN, los usuarios y los grupos agregados se muestra en la figura 3.12.

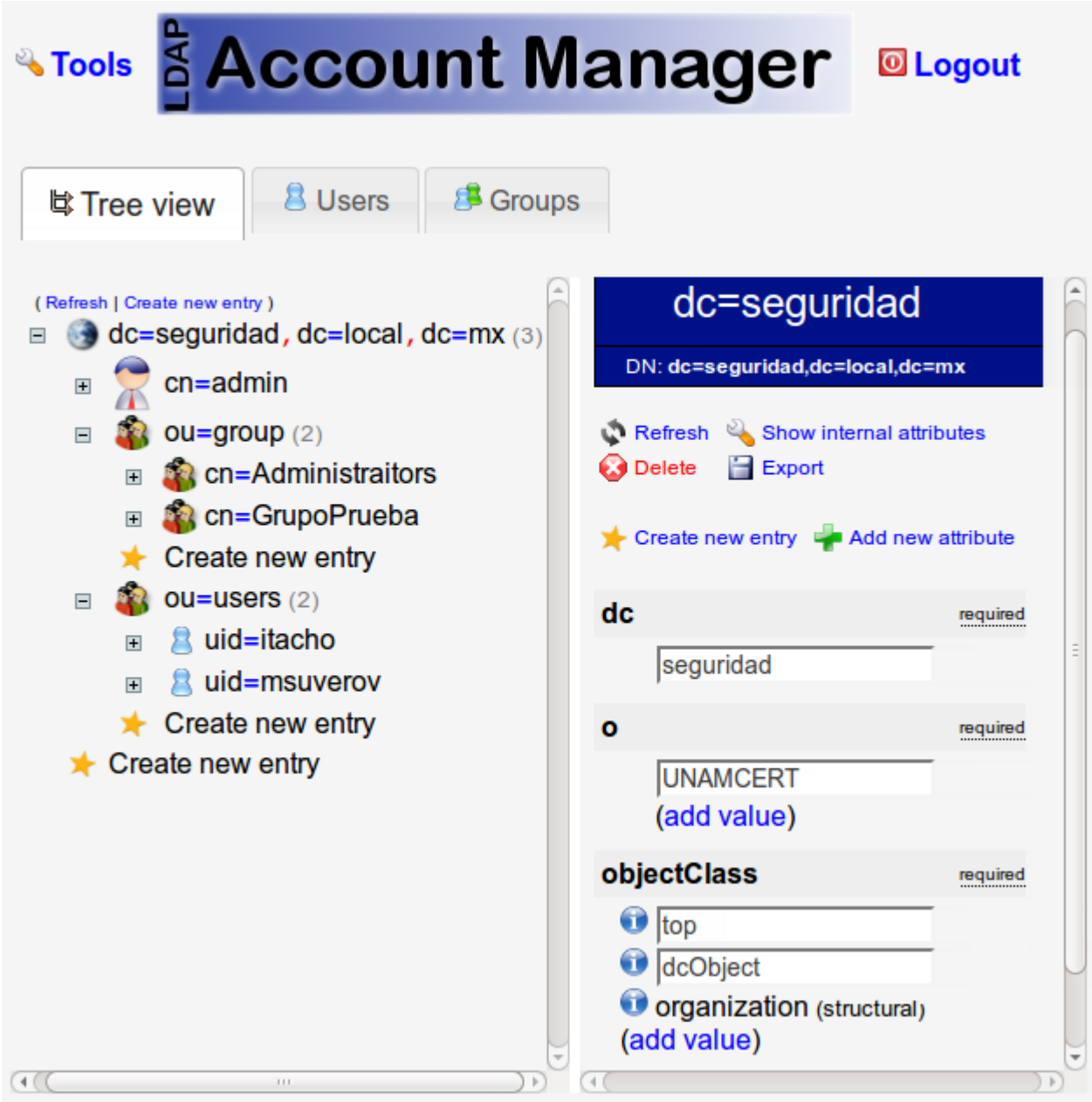

Figura 3.12 Árbol LDAP y su correspondiente DN

La entrada correspondiente al UID del usuario msuverov, se muestra en la figura 3.13.

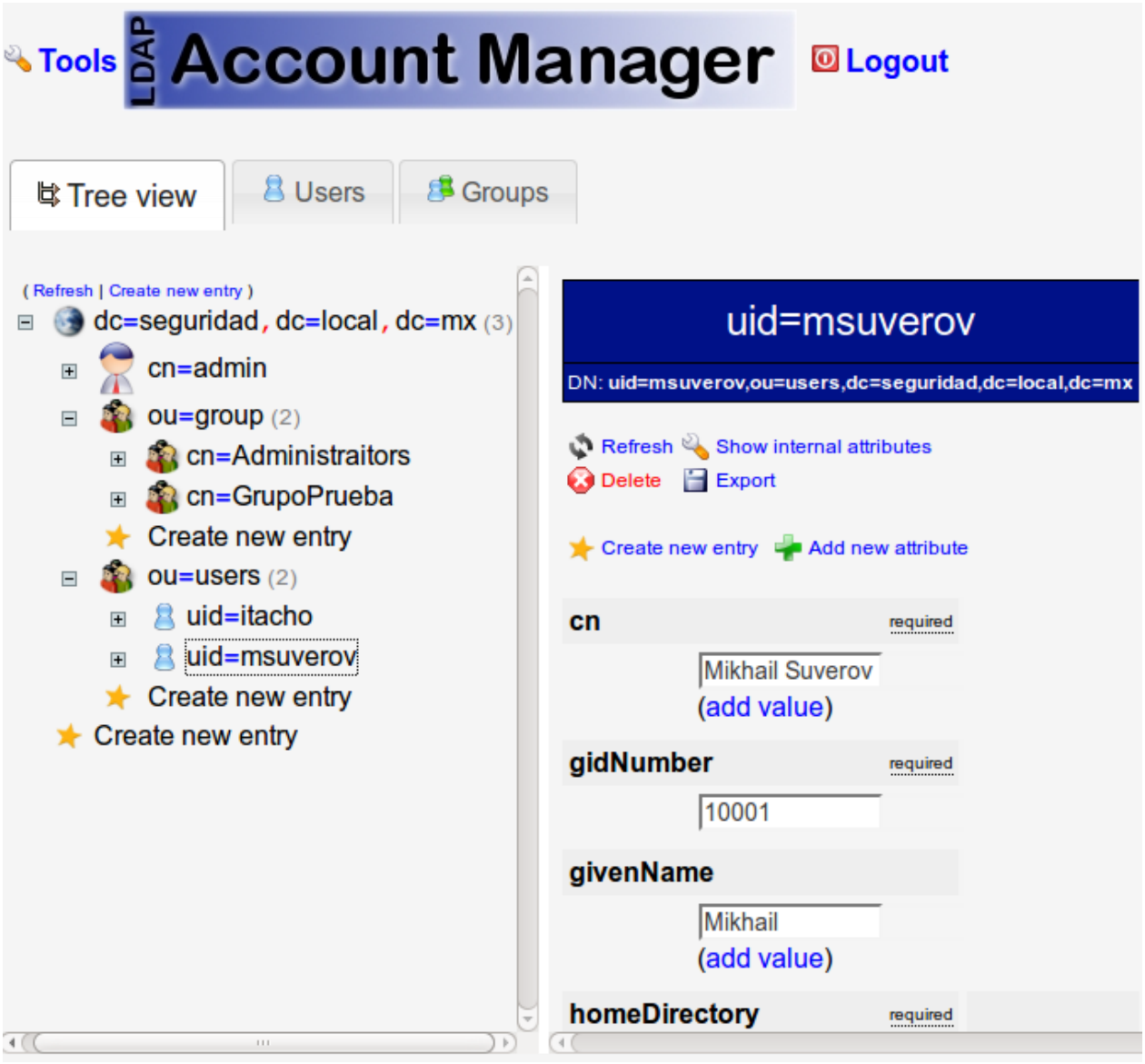

Figura 3.13 Entrada LDAP para el usuario msuverov

En la pantalla de la figura 3.13 es posible cambiar cualquier parámetro LDAP con relación a cada usuario. Y se puede ver el correspondiente DN dentro del directorio LDAP:

DN: uid=msuverov,ou=users,dc=seguridad,dc=local,dc=mx

Una vez agregados las cuentas de usuario en el directorio activo, el procedimiento de la prueba de conexión a LDAP se llevó a cabo mediante el script de conexión (script de prueba "*PruebaLDAP.php*") de la tabla 3.8.

Tabla 3.8 Script de prueba de conexión al servidor LDAP

CÓDIGO: PruebaLDAP.php

```
1.  <?php
2.  $LDAP_ou = 'ou=users';
3.  $LDAP_dc = 'dc=seguridad,dc=local,dc=mx';
4.  $ldap['user'] = "msuverov";
5.  $ldap['pass'] = "hola123";
6.  $ldap['host'] = "ldaps://10.0.1.110";
7.  $ldap['port'] = 636;
8.  $ldap['dn'] = 'uid='.$ldap['user'].','.$LDAP_ou.','.$LDAP_dc;
9.  $ldap['conn'] = ldap_connect( $ldap['host'], $ldap['port'] );
10. ldap_set_option($ldap['conn'], LDAP_OPT_PROTOCOL_VERSION, 3);
11. $ldap['bind'] = ldap_bind( $ldap['conn'], $ldap['dn'], $ldap['pass'] );
12.    if ($ldap['bind']){
13.       session_cache_limiter('nocache,private');
14.       echo "Conexión correcta al servidor LDAP<br>";
15.       echo "Autenticado como usuario: ".$ldap['user'];
16.    }
17.    else{
18.       echo "Error: problema con la conexión al servidor LDAP";
19.    }
20. ?>
```

El script de prueba de conexión al servidor LDAP, genera un mensaje de salida con 2 opciones, confirmando la conexión (líneas 14 y 15) y la autenticación satisfactoria del usuario que se asignó para la prueba, para este caso fue el usuario “msuverov”. La otra opción es cuando no se haya podido establecer la conexión se notifica con un mensaje de error (línea 18). En la figura 3.14 se muestra el resultado obtenido al ejecutar el script en el navegador web.

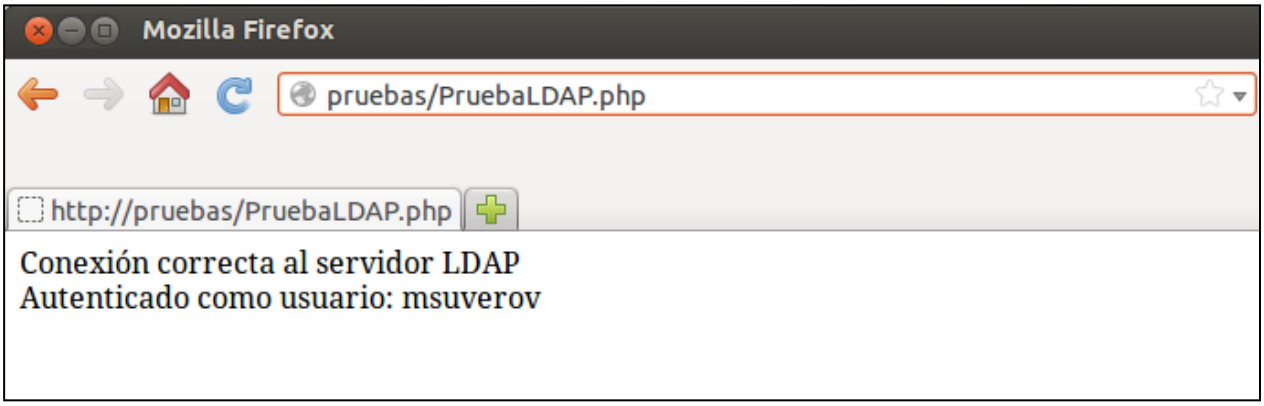

Figura 3.14 Respuesta generada por el script PruebaLDAP.php en el navegador web.

También se hizo una prueba de conexión mediante una terminal de comandos, el comando ejecutado y su respuesta se muestra en la figura 3.15.

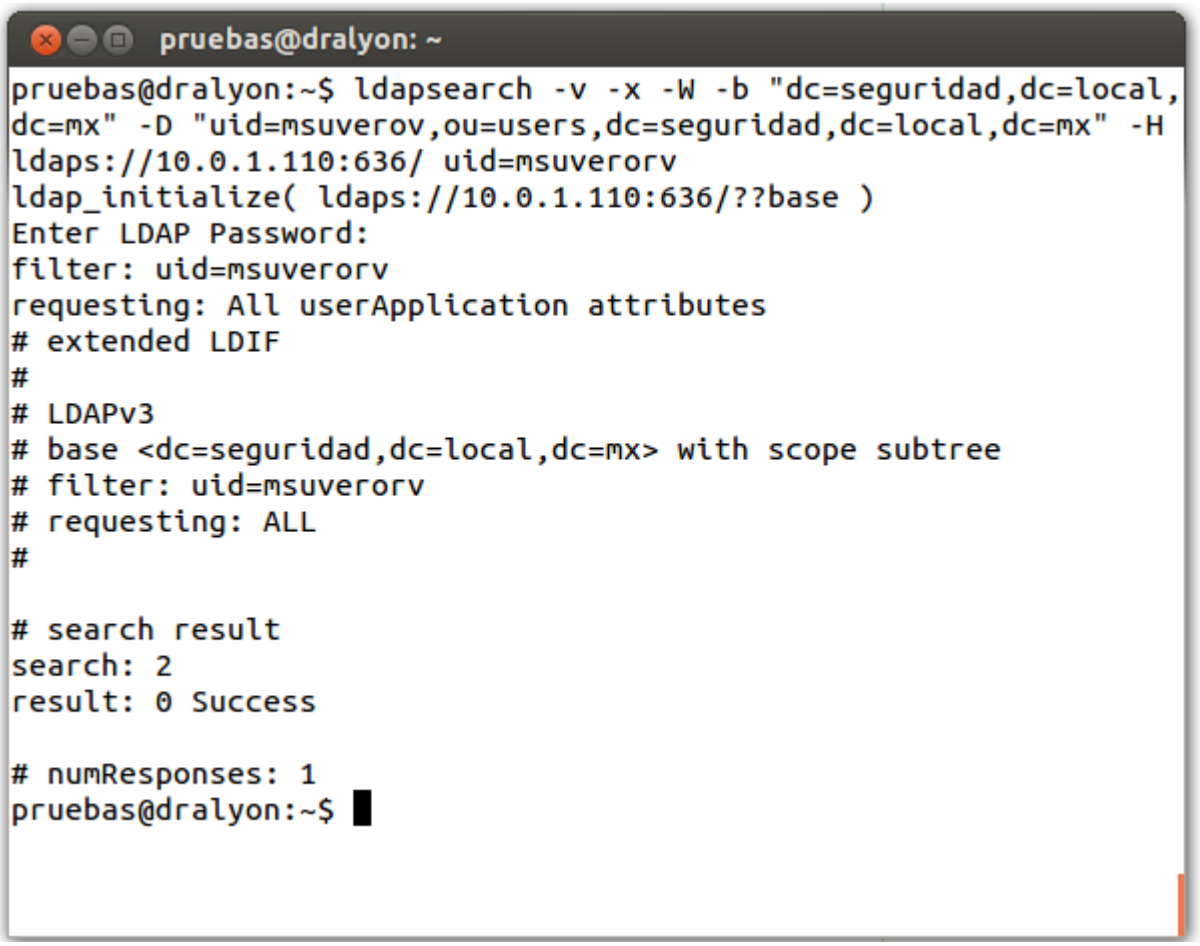

Figura 3.15 Prueba de conexión al servidor LDAP mediante una terminal de comandos

En la ventana de la terminal se puede observar que se ha podido realizar una búsqueda en el directorio LDAP obteniendo una respuesta indicando en qué lugar del árbol base se encuentra la entrada solicitada, para este caso se utilizó solamente el UID como filtro para realizar la búsqueda y verificar la conexión con el servidor LDAP.

CAPÍTULO 4

AUTENTICACIÓN CENTRALIZADA DE USUARIOS

4.1 Iniciando sesión en el sistema

Teniendo el escenario listo, y verificada la conexión con el servidor LDAP local, se procedió a realizar el inicio de sesión activando ya el "*MAC.php*" en el sistema de pruebas que fue objetivo de la autenticación centralizada. En la figura 4.1 se muestra la pantalla de inicio del sistema, solicitando las credenciales de acceso.

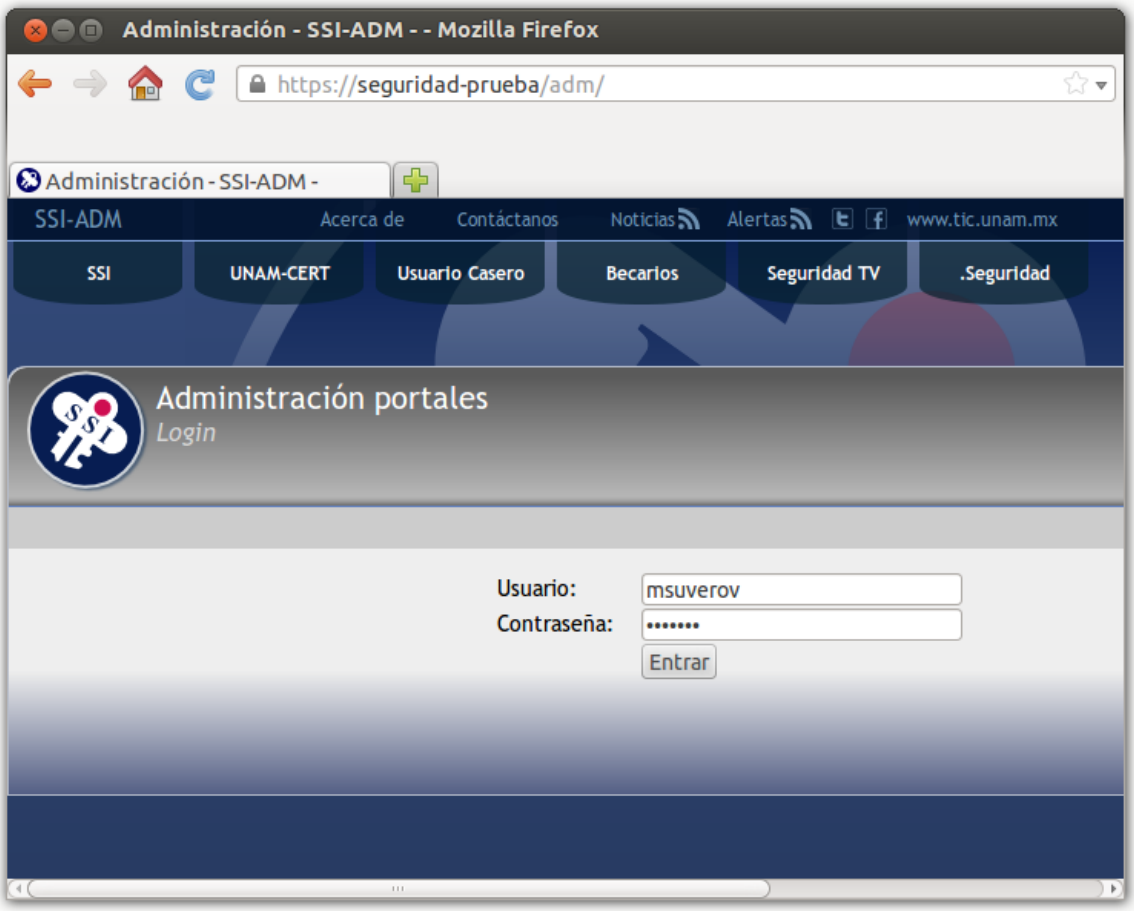

Figura 4.1 Sistema de pruebas de la administración de contenidos del portal seguridad de la SSI

Con base a los usuarios creados en el directorio activo, al ingresar las credenciales de "*msuverov*" el servidor web realizará la sincronización de la cuenta en LDAP con el servidor de base de datos, la sincronización consiste en almacenar casi todos los datos del usuario a excepción de la contraseña, ya que ésta únicamente se encontrará almacenada en el directorio activo.

La sincronización de la cuenta es un proceso tranparente para los usuarios, en la siguiente figura (ver figura 4.2) se puede observar que se ha podido acceder al sistema.

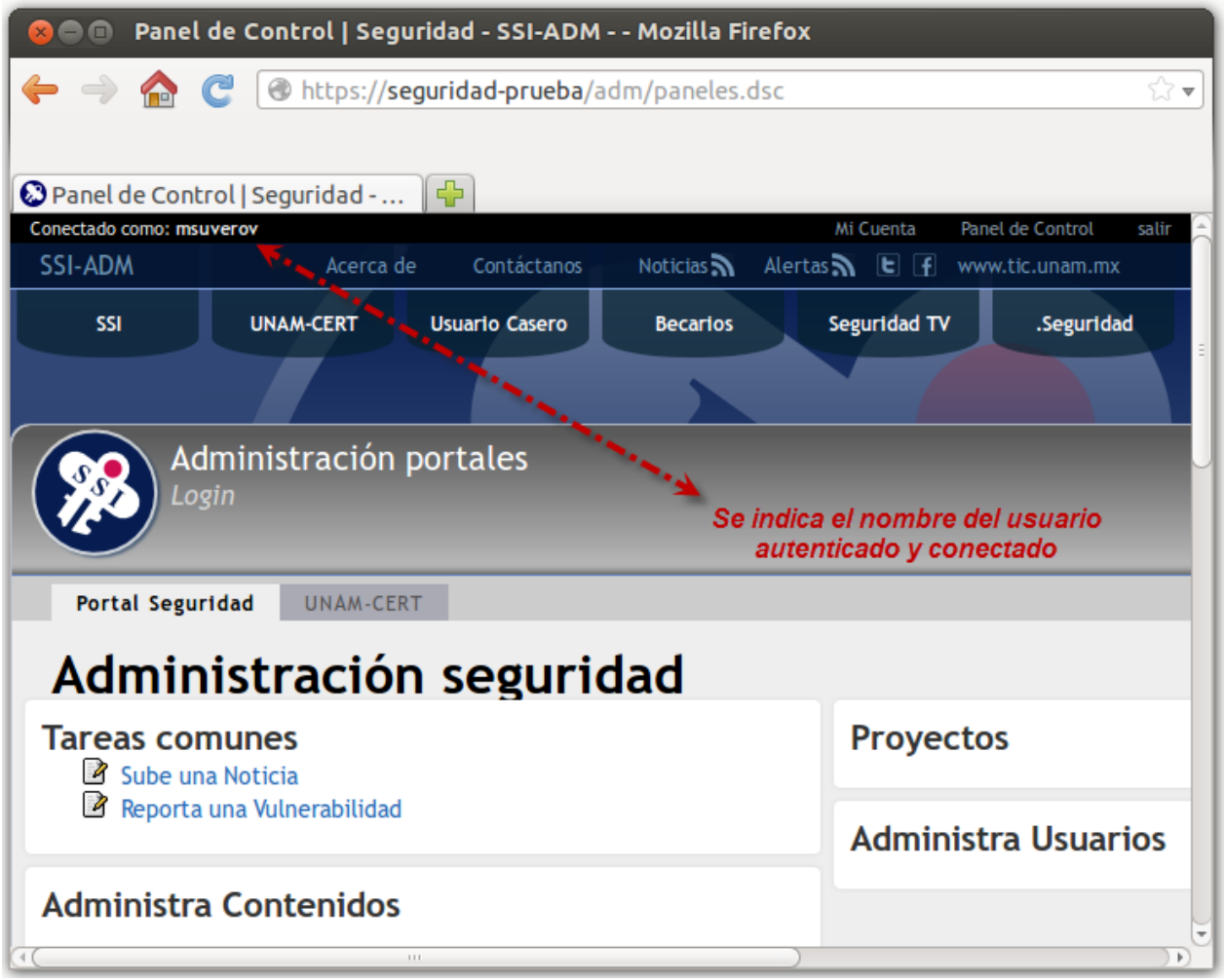

Imagen 4.2 Usuario "msuverov" autenticado en el sistema

En la figura 4.2 al observar la URL, se está utilizando el protocolo HTTPS para el cifrado de la conexión, además se nota que el identificador de sesión ya no se transmite vía URL, sino mediante las variables de sesión establecidas dentro del código fuente. En la pantalla inicial se muestra las opciones disponibles que tiene el usuario tales como subir una noticia y reportar

una vulnerabilidad, las demás opciones como proyectos, administra contenidos y administra usuarios no contienen opciones disponibles aún para “msuverov”.

Del lado superior derecho de la figura 4.2 también se pueden observar otras opciones más, una de ellas corresponde a los datos de la cuenta del usuario que pueden verse en la figura 4.3 a mayor detalle.

Figura 4.3 Detalles de la opción “Mi cuenta”

Si el usuario deseara cambiar su contraseña, esto debe hacerse mediante el administrador del correo electrónico, ya que ahora las contraseñas se encuentran en el servidor LDAP, por lo que ya no es posible hacer esta operación mediante las opciones de cuenta del sistema portal seguridad.

4.2 Control de acceso al sistema

Una de las características que posee el portal de seguridad para los usuarios, es el control de acceso al contenido del sistema. Anteriormente se había mencionado que se requería determinada información del usuario, para que con ello fuera posible llevar a cabo las operaciones de acceso y denegación al contenido que tiene derecho cada usuario.

El sistema portal seguridad cuenta con un administrador de contenidos nativo, que permite acceder a los archivos que contienen las publicaciones e información relevante que se publica en el portal. Además permite administrar y controlar el acceso que posee cada usuario a esta información, tanto a archivos como a directorios, no todos los usuarios son administradores generales y tampoco todos los usuarios tienen acceso a toda la información.

Para ejemplificar este procedimiento en el sistema, se concederán privilegios de administrador general al usuario “msuverov”. En la figura 4.4 se puede corroborar este hecho.

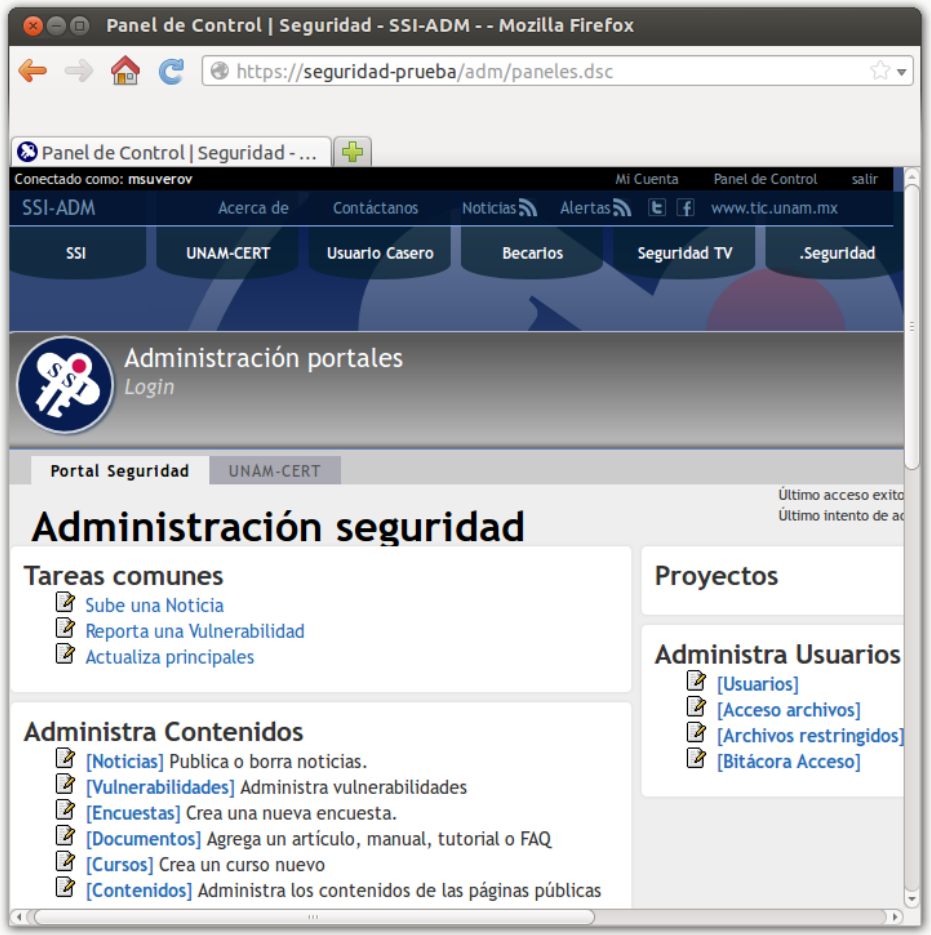

Figura 4.4 Privilegio de administrador al usuario “msuverov”

En la figura 4.4 se puede observar que las opciones en el perfil del usuario “msuverov” han aumentado. Al acceder a Contenidos->Usuario Casero se tienen los siguientes archivos mostrados en la figura 4.5.

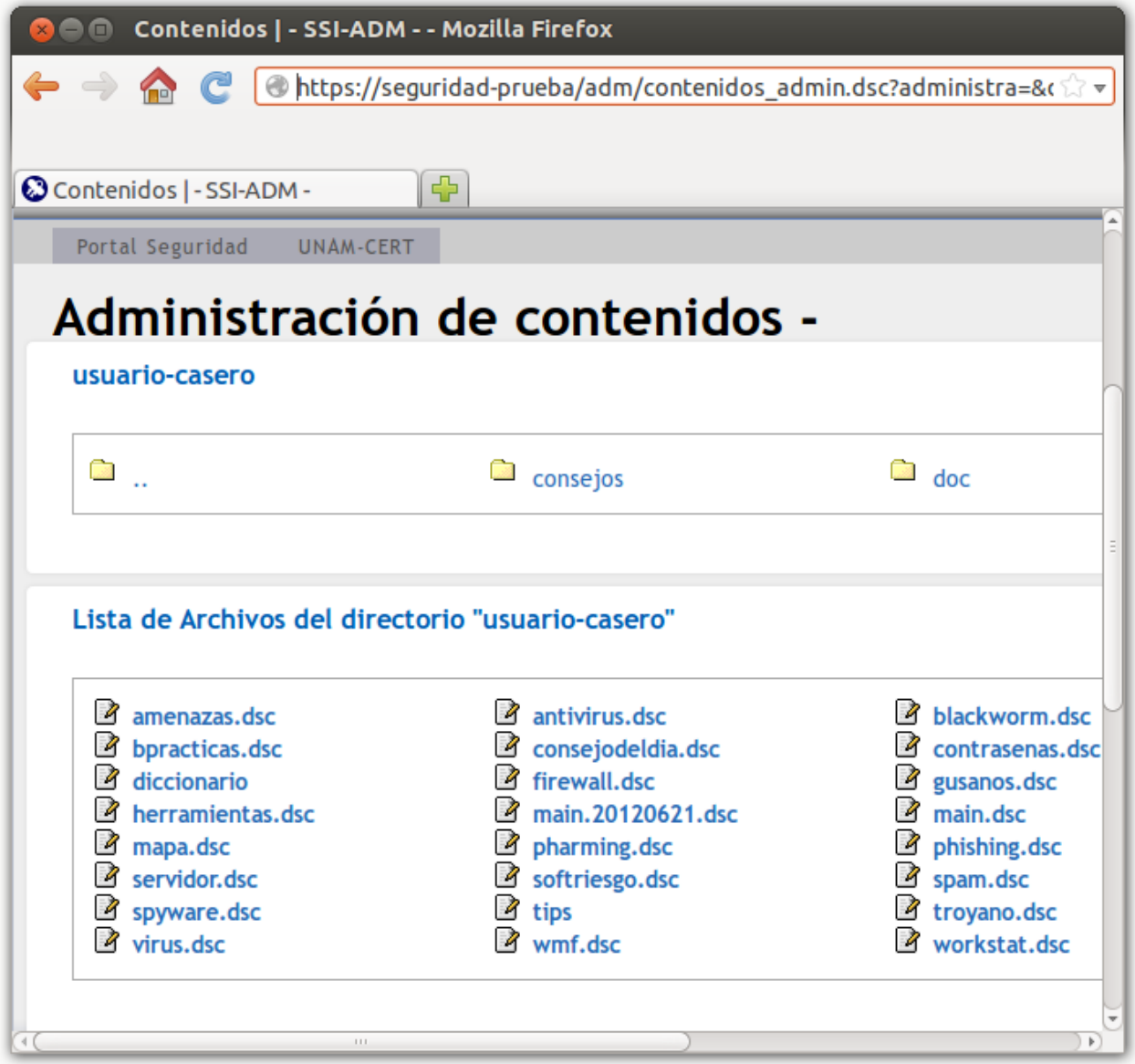

Figura 4.5 Archivos y directorios de la sección usuario-casero

Estos archivos corresponden información básica de conceptos y amenazas en la web para todo usuario que utiliza una computadora, dirigida esta información en su mayor parte a usuarios que no conocen mucho sobre estos términos.

Siguiendo con el ejemplo, ahora se concederán permisos al usuario “itacho” para acceder a los archivos de administración correspondientes a Noticias, en la figura 4.6 se puede visualizar la activación de estos permisos.

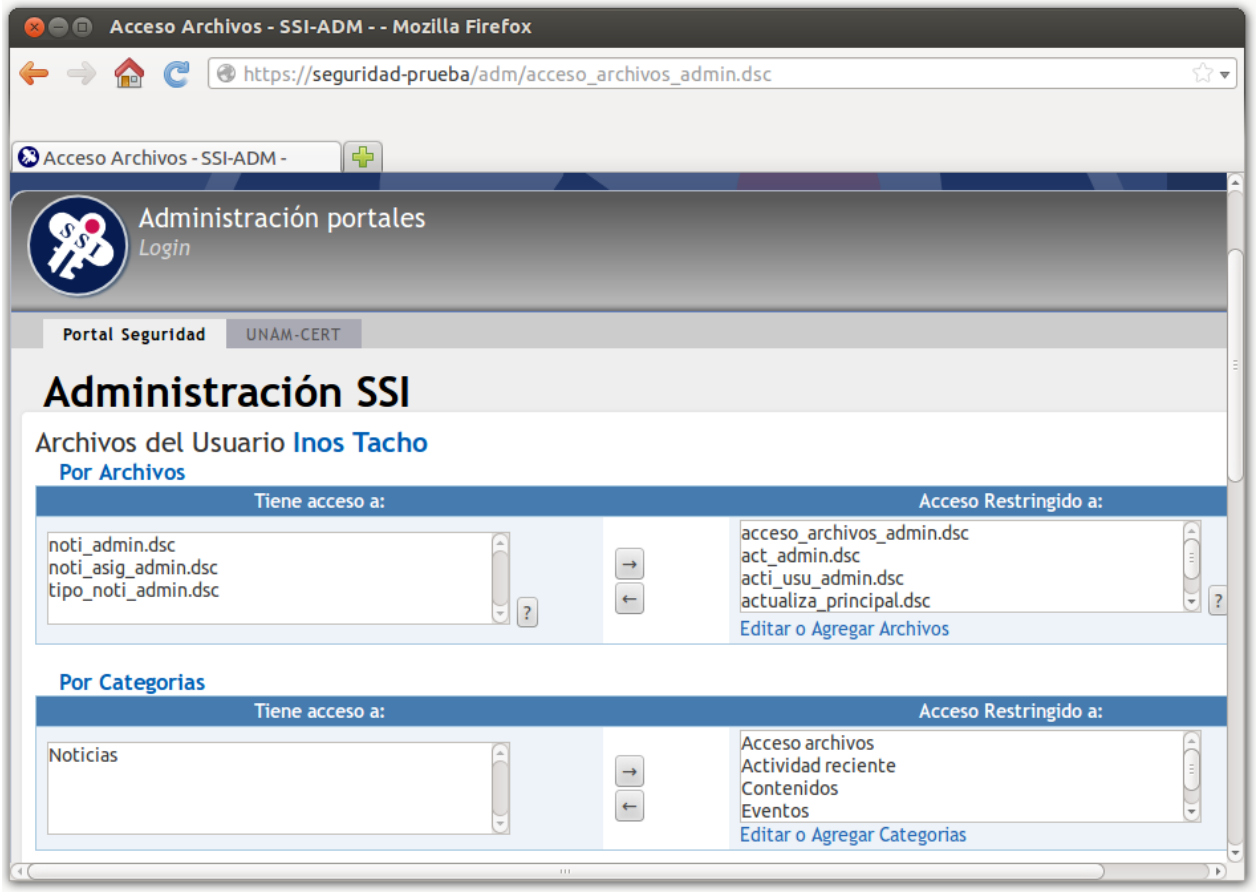

Figura 4.6 Concesión de permisos de acceso al usuario “itacho” a la categoría noticias

Es posible establecer acceso a toda una sección de información o simplemente a determinados archivos de una sección, esto se efectúa dependiendo del rol que desempeñe el usuario dentro de la SSI, para este caso se concedió el privilegio de acceso a la categoría completa.

Para visualizar la activación del acceso a la categoría Noticias, se accede al sistema con la cuenta “itacho”.

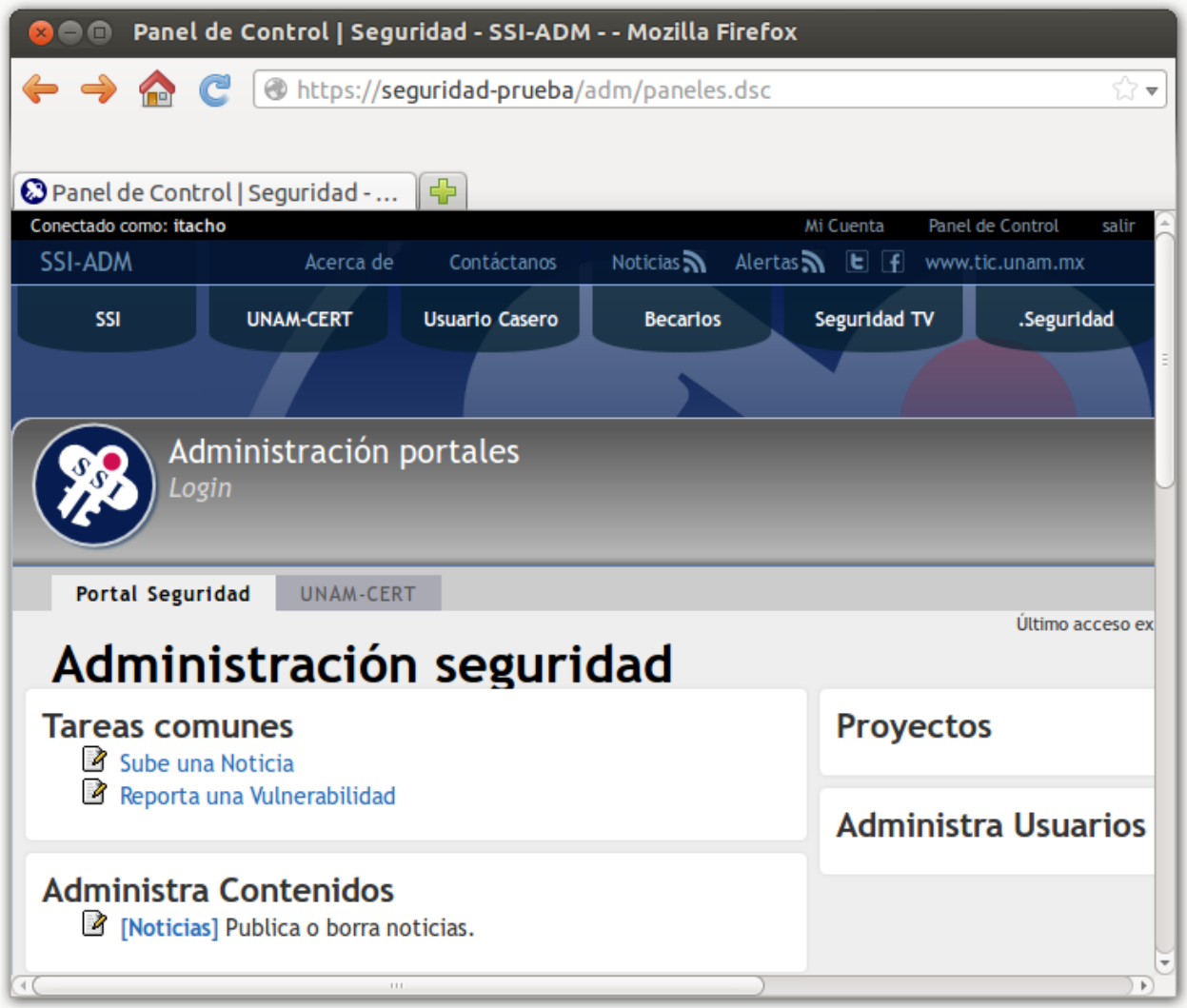

Figura 4.7 Inicio de sesión con la cuenta "itacho"

En la figura 4.7 se puede visualizar que se ha añadido un acceso a la parte administrativa de contenidos, ahora el usuario "itacho" puede publicar y editar las noticias que otros usuarios suban al portal de seguridad. De este modo el usuario "itacho" es el administrador de noticias del portal seguridad.

4.3 Administración de usuarios

Otro de los puntos a tratar en este capítulo, es sobre la administración de usuarios tanto dentro como fuera del sistema.

Por un lado se tiene la parte interna, en la que antes de activar el módulo de autenticación centralizada la administración de usuarios se llevaba a cabo dentro del mismo sistema. El administrador general tenía la facultad de crear, editar y eliminar cuentas de

usuario, cada vez que se integraba un nuevo miembro a la SSI tenía que ser dado de alta en el sistema, de igual manera cuando deja de laborar en la SSI, su cuenta era eliminada.

Una vez activado el módulo de autenticación centralizada, el alta o baja de cuentas de usuario ya no depende del administrador general del sistema, sino del administrador del servidor LDAP.

Por otro lado la administración externa de altas, bajas y edición de cuentas de usuarios corresponde a la administración de usuarios externa, es decir ahora está fuera del alcance del sistema portal seguridad, recayendo esta actividad sobre un solo administrador de usuarios.

Ahora el administrador del servidor LDAP al momento de dar de alta un nuevo usuario, el sistema de administración del portal seguridad sincroniza su cuenta de usuario cuando ingresa al sistema por primera vez. Y cuando este usuario deje de laborar en la SSI, al darse de baja la cuenta de usuario en el servidor LDAP, simplemente ese usuario ya no tendrá acceso al sitio de administración del portal seguridad, aunque ingrese sus credenciales de acceso correctamente, ya que el sistema ahora se autentica de manera centralizada.

La administración de usuarios de manera interna está más enfocada hacia el control de acceso, a establecer permisos y privilegios a los usuarios al contenido del sistema.

4.4 Política de creación de contraseñas

La SSI en una de sus políticas de seguridad sugiere la creación de contraseñas robustas, al momento de contar con varios accesos a diferentes sistemas, la tendencia de creación de una contraseña robusta disminuye. La mayoría de las veces sucede por la periodicidad que tienen los usuarios para acceder a los sistemas, por ejemplo cuando se accede a un sistema de manera esporádica, es más fácil que el usuario olvide su contraseña en comparación con el acceso al correo electrónico, que es de uso más frecuente y difícilmente olvidaría la contraseña.

En ocasiones cuando se dispone el acceso a varios servicios, lo más común es acceder a ellos con una única contraseña, esto para no olvidarlas. La desventaja de esta práctica es que si la contraseña maestra se compromete, se tendrá acceso a todos los servicios a los que se disponga dicho acceso.

Tener una única contraseña para todos nuestros accesos a los sistemas implica un riesgo, más aún cuando la contraseña no está robustecida. Cuando se trata de sistemas de autenticación independiente, cada sistema almacena contraseñas en su respectiva base de datos, y la administración de las contraseñas recae en los administradores de cada sistema, en cambio teniendo sistemas en los que la autenticación se lleva a cabo de manera centralizada, la

administración de los usuarios y las contraseñas recae sobre el administrador del servidor de autenticación, pero de igual manera se tiene el riesgo de utilizar una única contraseña.

La creación de una contraseña robusta implica que, ésta debe ser al menos de 8 caracteres, combinando letras mayúsculas, minúsculas, números y al menos un carácter especial; de esta forma al tener una contraseña robusta se mitiga el riesgo de que pueda ser comprometida. Se debe evitar utilizar parámetros o información asociada a información personal, es decir evitar el uso de fechas de cumpleaños, nombres, números telefónicos etc. De esta manera se puede definir una correcta política de creación de contraseñas.

CONCLUSIONES

El proyecto del módulo de autenticación centralizada (MAC), fue enfocado principalmente a cambiar el esquema de autenticación de usuarios en los sistemas que se encuentran dentro la SSI. Se consiguió también sustituir el anterior esquema de autenticación, ya que ése implicaba considerar una recopilación de los agentes de autenticación y dependencias adicionales de software al querer migrar algún sistema web a otro servidor, además que el tiempo de migración se vería afectado en un momento dado.

Con esta implementación del módulo de autenticación centralizada, se consigue depender únicamente del lenguaje de programación PHP, en dado caso que se deseara migrar de servidor el servicio web, la migración se hace de manera más versátil.

A pesar de que existe una variedad de soluciones que ofrecen autenticación centralizada, éstas requieren determinados recursos y requerimientos de software, y que por cuestiones internas de la SSI o compatibilidad, pueden no ser factibles de implementar. El MAC no exige recursos ni requerimientos de software especiales, por el contrario, se adjunta con el código fuente del sistema o servicio web, también fue diseñado y pensado para nuevos proyectos.

Actualmente el MAC se encuentra implementado en uno de los sistemas de producción, el cual es el portal de administración de seguridad. Se pretende también implementarlo en algunos otros servicios web, pero para ello se requieren seguir ciertos procedimientos de solicitudes de cambio en dichos sistemas, ya que éstos son de uso crítico.

Existen grandes ventajas y beneficios al utilizar el MAC, dentro de los que destacan la ejecución de una adecuada política de creación de contraseñas, al manejar una única contraseña para acceder a todos los sistemas a los que se tenga acceso, un solo administrador se encarga de gestionar las cuentas de usuario, esto último resulta muy útil, ya que existe una constante rotación del personal que labora en la SSI, de este modo solamente basta dar de alta o de baja a un usuario en el servidor de autenticación y evitar así que cada administrador de cada sistema al que tenía acceso ese usuario lo haga.

Así como existen ventajas y beneficios se encuentran un par de desventajas, una de ellas consiste en que si la contraseña de un usuario se compromete, se tendrá un acceso total a todos los servicios que tenga acceso dicho usuario. Por otro lado, si el servidor de autenticación es vulnerado, toda la información respecto a los usuarios y la autenticación a los sistemas que se sirvan de él para la autenticación, se vería completamente comprometido.

Las desventajas son mitigables, ya que existen controles de seguridad para todos y cada uno de los sistemas de información. Cada sistema cuenta con el protocolo HTTPS activado, lo que permite cifrar la información de las credenciales de acceso, permitiéndole al MAC tratar dichas credenciales de manera segura, y una gestión adecuada de los identificadores de sesión que se generan cada vez que un usuario accede al sistema, evitando así un posible robo de sesión.

La SSI cuenta con procedimientos para la mejora continua, por lo que el MAC estará sujeto a ello, sometido a pruebas de penetración para detección de vulnerabilidades, que en caso de ser detectadas, éstas sean corregidas y el MAC sea robustecido.

TRABAJO A FUTURO

Una vez observado el funcionamiento del módulo de autenticación en un ambiente de producción, y ver las ventajas de administración de cuentas de usuario; se piensa ahora en la posibilidad de implementación en otros servicios web, algunos de ellos no son de uso recurrente, pero cada cierto tiempo su uso es muy concurrido. De esta manera permite tener un tiempo dedicado para su implementación.

Por otro lado, ¿qué sucedería si se presentan nuevos proyectos en los cuales se requiera una autenticación y administración de usuarios?, pensando en esta cuestión, se pretende trabajar sobre un configurador del módulo de autenticación, en el que se puedan proporcionar datos del nuevo sistema a desarrollar, por ejemplo se pueden considerar las siguientes opciones:

- Permitir establecer un nombre de archivo para el módulo, en vez de MAC.php, que esto pueda ser un parámetro personalizable eligiendo otro nombre.
- Si se requiere centralizar la autenticación, que el configurador pueda recibir los datos del servidor de autenticación, como son IP del servidor, puerto y los parámetros del DN.
- Poder introducir la URL y la página de acceso principal al sistema en el configurador, para que el módulo de autenticación pueda realizar el direccionamiento correcto dentro de las páginas del sistema y también se lleve a cabo el adecuado despliegue y manejo de errores.
- Si también se desean implementar políticas de seguridad sobre los accesos de los usuarios al sistema, crear de manera automática las tablas requeridas en la base de datos, permitiendo especificar los nombres de las tablas, sus campos y las referencias a la tabla principal de usuarios del sistema, esto también sería muy útil para la sincronización de las cuentas de usuario con el servidor de autenticación.

Los puntos anteriores corresponden a las partes del módulo que pueden ser personalizables, la lógica y el tratamiento del ticket y del ID de sesión se conservaría, pero también se puede pensar en un método diferente para gestionar estos parámetros.

Una opción de cambio o de mejora para la seguridad y el manejo de credenciales de usuario, sería tokenizar estos parámetros, esto es, asociar valores únicos a datos confidenciales que son empleados como reemplazo de estos últimos, con la característica diferencial de que el token no permite inferir el dato confidencial con el que se relaciona, minimizando de esta manera el riesgo de almacenamiento inseguro y la necesidad de implementación de controles

asociados a datos confidenciales, ya que el token en sí no es catalogado como un dato confidencial. En síntesis sería emplear un mecanismo de autenticación muy similar a OAuth, para la autenticación y el manejo de credenciales de usuario.

BIBLIOGRAFÍA Y MESOGRAFÍA

[1] Allan Liska, *The practice of network security: deployment strategies for production environments*, Estados Unidos, Prentice Hall, 2002

[2] Instituto Nacional de Tecnologías de la Comunicación; *Estudio de mecanismos de Identidad Digital sobre Televisión Digital Terrestre*, [en línea], pp.25 URL:http://es.scribd.com/doc/55627154/23/M1-AUTENTICACION-MEDIANTE-EL-PAR-USUARIO-CONTRASENA

[3] María Carmen España Boquera, *Servicios avanzados de telecomunicación*, España, Díaz de Santos, 2003

[4] Izaskun Pellejero, Fernando Andreu, Amaia Lesta, *Fundamentos y aplicaciones de seguridad en redes WLAN*, España, Marcombo, 2006

[5] *Autenticación de usuarios* [en línea], http://www.rediris.es/cert/doc/unixsec/node14.html

[6] s/a, *Autenticación de usuarios*, volumen 9 [en línea], http://www.juniper.net/techpubs/software/screenos/screenos6.0.0/translated/Spanish/ce_v9_sp.pdf

[7] *Sistemas de autenticación* [en línea], http://www.arcos.inf.uc3m.es/~folcina/pfc-html/node25.html

[8] Christopher Leidigh, *Principios de la seguridad en redes* [en línea], http://cxo-community.com/articulos/blogs/blogs-seguridad-informatica/2136-principios-de-la-seguridad-de-redes-parte-ii.html

[9] Ravikanth Ponnapalli, *Secure Coding Practical steps to defend your web apps*, Sans Institute [en línea], http://software-security.sans.org/resources/paper/reading-room/secure-implementation-enterprise-single-sign-on-product-organization

[10] *Introducción a la autenticación TSL*, Microsoft TechNet, [en línea], http://technet.microsoft.com/es-es/library/cc782610%28WS.10%29.aspx

[11] Jason Garman, *Kerberos The Definitive Guide*, O'Really, Estados Unidos, 2003

[12] Daniel J. Barrett, Richard E. Silverman, Robert G. Byrnes, *SSH, the Secure Shell: The Definitive Guide*, Estados Unidos, O'Reilly, 2005

[13] Bill von Hagen; Brian K. Jones, *Linux Server Hacks*, Estados Unidos, O'Reilly, Volumen 2, 2006

[14] William Stallings, *Fundamentos de seguridad en redes*, España, Prentice Hall, 2004, Segunda edición

[15] *Tipos de protocolo IPSec modo transporte*, Microsoft TechNet, [en línea], http://technet.microsoft.com/es-es/library/cc739674%28WS.10%29.aspx

[16] *Tipos de protocolo IPSec modo túnel*, Microsoft TechNet, [en línea], http://technet.microsoft.com/es-es/library/cc737154%28WS.10%29.aspx

[17] *Authentication, Authorization and Access Control*, Apache Org., [en línea], http://httpd.apache.org/docs/2.2/howto/auth.html

[18] Fernando Berzal, Francisco José Cortijo, Juan Carlos Cubero, *Desarrollo Profesional de Aplicaciones Web con ASP.NET*, iKor Consulting

[19] *PHP Sesiones*, Wextensible, [en línea], http://www.wextensible.com/temas/php-sesion/

[20] API PHP-LDAP (documentación) [en línea], http://mx2.php.net/manual/es/book.ldap.php

[21] Philippe Mathon, *VPN: Implementación en Windows Server*, ENI, España, 2003

[22] John Viega, Matt Messier, Pravir Chandra, *Network Security with OpenSSL*, O'Reilly, Estados Unidos, 2002

[23] *Certificados de IIS y de cliente* [en línea], http://support.microsoft.com/kb/907274/es

[24] *Introducción al concepto de certificados* [en línea], http://es.kioskea.net/contents/crypto/certificat.php3

[25] *OAuth Facebook developers* [en línea], http://developers.facebook.com/docs/concepts/login/login-architecture/

[26] *OAuth Twitter developers* [en línea], https://dev.twitter.com/docs/auth/oauth/faq

[27] Carter Gerald, *LDAP System Administration*, Estados Unidos, O'Reilly, 2003

[28] Rafael Calzada Pradas, *Introducción al servicio de directorio* [en línea], http://www.rediris.es/ldap/doc/ldap-intro.pdf

[29] Timothy A. Howes, Ph.D, Marck C.Smith, Gordon S.Good, *Understanding and Deploying LDAP directory services*, Estados Unidos, Addison-Wesley, Segunda Edición, 2003

[30] David Fernandez Vaamonde, *Implementaciones OpenLDAP* [en línea], http://david.f.v.free.fr/ponencias/OpenLDAP/node3.html

[31] *LDAP* [en línea], http://www.arcos.inf.uc3m.es/~folcina/pfc-html/node28.html

[32] OpenLDAP Software servidor LDAP (documentación) [en línea], http://www.openldap.org

[33] Cortés Morales Roberto, *Introducción al análisis de sistemas y la ingeniería de software*, Costa Rica, EUNED, 1998

[34] *Manual pages LDAP.conf* [en línea], http://www.openldap.org/software/man.cgi?query=ldap.conf&apropos=0&sektion=0&manpath=OpenLDAP+2.4-Release&format=html

[35] *Ventajas e inconvenientes del centralizar la gestión de usuarios* [en línea], http://blackshell.usebox.net/archive/ventajas-e-inconvenientes-del-centralizar-la-gestion-de-usuarios.html

[36] *Ventajas en el uso de LDAP* [en línea], http://www.ldap-es.org/contenido/04/11/1.2.-ventajas-en-el-uso-de-ldap

[37] "522.1 Web Basics and authentication security", *Defending web applications security essentials*, Sans Institute, 2011

[38] "522.5 Cutting-Edge Web Security", *Defending web applications security essentials*, Sans Institute, 2011

[39] David Eduardo Acosta, *Fundamentos de tokenización y el cumplimiento de PCI y DSS* [en línea],
http://www.isecauditors.com/sites/default/files/files/RedSeguridad_49_Tokenizacion.pdf

REFERENCIAS

[1] García Pelayo y Gross Ramón, Larousse diccionario básico de la lengua española, Larousse, 2011.

[2] Idem

[3] Allan Liska, The practice of network security: deployment strategies for production environments, Estados Unidos, Prentice Hall, 2002, p. 123

[4] Ibid., p. 124

[5] Instituto Nacional de Tecnologías de la Comunicación; Estudio de mecanismos de Identidad Digital sobre Televisión Digital Terrestre, [en línea], pp.25 URL:http://es.scribd.com/doc/55627154/23/M1-AUTENTICACION-MEDIANTE-EL-PAR-USUARIO-CONTRASENA [consulta: 12 de junio de 2011]

[6] Idem.

[7] Idem., p. 26

[8] María Carmen España Boquera, Servicios avanzados de telecomunicación, España, Díaz de Santos, 2003, pp.291-294

[9] Izaskun Pellejero, Fernando Andreu, Amaia Lesta, Fundamentos y aplicaciones de seguridad en redes WLAN, España, Marcombo, 2006, pp. 76-94

[10] Introducción a la autenticación TSL; Microsoft TechNet, [en línea], http://technet.microsoft.com/es-es/library/cc782610%28WS.10%29.aspx [consulta del 21 de agosto de 2011]

[11] Izaskun Pellejero, op. cit., pp. 76-94

[12] Introducción a la autenticación TSL; Microsoft TechNet, [en línea], http://technet.microsoft.com/es-es/library/cc782610%28WS.10%29.aspx [consulta del 21 de agosto de 2011]

[13] María del Carmen Romero Ternero, et al., Servicios en Red, España, Paraninfo, 2010, pp. 154-155

[14] Daniel J. Barrett; Richard E. Silverman; Robert G. Byrnes; SSH, the Secure Shell: The Definitive Guide, Estados Unidos, O'Reilly, 2005, pp. 3

[15] Bill von Hagen; Brian K. Jones, Linux Server Hacks, Estados Unidos, O'Reilly, Volumen 2, 2006 pp. 29

[16] Daniel J. Barrett, op. cit., pp. 124

[17] William Stallings, Fundamentos de seguridad en redes, España, Prentice Hall, 2004, Segunda edición, pp. 179-180

[18] Tipos de protocolo IPSec modo transporte; Microsoft TechNet, [en línea], http://technet.microsoft.com/es-es/library/cc739674%28WS.10%29.aspx [consulta del 3 de agosto de 2011]

[19] Tipos de protocolo IPSec modo túnel; Microsoft TechNet, [en línea], http://technet.microsoft.com/es-es/library/cc737154%28WS.10%29.aspx [consulta del 3 de agosto de 2011]

[20] Izaskun Pellejero, op. cit., pp.126

[21] Authentication, Authorization and Access Control, Apache Org., [en línea], http://httpd.apache.org/docs/2.2/howto/auth.html, [consulta del 5 de enero de 2012]

[22] Idem.

[23] Fernando Berzal, Francisco José Cortijo, Juan Carlos Cubero, Desarrollo Profesional de Aplicaciones Web con ASP.NET, iKor Consulting, pp. 87

[24] PHP Sesiones, Wextensible, [en línea], http://www.wextensible.com/temas/php-sesion/, [consulta 23 de septiembre de 2012]

[25] Carter Gerald, LDAP System Administration, Estados Unidos, O'Reilly, 2003, pp. 7

[26] Timothy A. Howes, Ph.D, Marck C.Smith, Gordon S.Good, Understanding and Deploying LDAP directory services, Estados Unidos, Addison-Wesley, Segunda Edición, 2003, pp. 54

[27] Timothy A. Howes, Ph.D, Marck C.Smith, Gordon S.Good, Understanding and Deploying LDAP directory services, Estados Unidos, Addison-Wesley, Segunda Edición, 2003, pp. 60

[28] Timothy A. Howes, Ph.D, Marck C.Smith, Gordon S.Good, Understanding and Deploying LDAP directory services, Estados Unidos, Addison-Wesley, Segunda Edición, 2003, pp. 69

[29] Carter Gerald, LDAP System Administration, Estados Unidos, O'Reilly, 2003, pp. 26

[30] Cortés Morales Roberto, Introducción al análisis de sistemas y la ingeniería de software, Costa Rica, EUNED, 1998, pp. 82

[31] Manual pages LDAP.conf [en línea], http://www.openldap.org/software/man.cgi?query=ldap.conf&apropos=0&sektion=0&manpath=OpenLDAP+2.4-Release&format=html, [consulta del 14 de julio de 2012]