Universidad Nacional de La Plata
Facultad de Informática

Tesis de grado

"Integración de aplicaciones/servicios basados en Web usando firma digital"

2006

A.C. Viviana Fonseca
A.C. Dalila Romero

Agradecimientos

Prefacio

Módulo 1: Conceptos teóricos

Módulo 2: Montado de la Infraestructura PKI

Módulo 3: Implementación

Módulo 4: Conclusiones del trabajo

Bibliografía

Apéndice

Queremos agradecer:

A nuestros padres y hermanos por el apoyo incondicional que nos dieron durante nuestra vida y en especial por darnos la oportunidad de llegar a este momento.

A Ale, Juanjo y a nuestros pequeños Anita y Juanse, por comprender nuestras ausencias y ayudarnos a seguir adelante.

A nuestra amiga Paula, por darnos el empuje inicial, su apoyo, su tiempo, por estar cada vez que la necesitamos y por sus oportunas correcciones.

A Mauricio y Jorge, por aportarnos sus conocimientos.

A nuestro director Lic. Javier Diaz, por sus valiosas sugerencias y aportes para el desarrollo de este trabajo.

A todos aquellos que alguna vez preguntaron: "...para cuando la tesis..."

Dalila y Viviana

Prefacio

La firma digital es una tecnología que se ha ido expandiendo en aplicaciones de red, acceso a sitios seguros, conexiones seguras, etc. Esta tecnología esta descripta con mayor detalle en las tesis de grado "Utilizando Firma Digital" de Paula Venosa - Verónica Fredes y en "Implementando Firma Digital con J2EE" de Alejandro Falcone y Maria Clemens, sin embargo, el uso de firma digital en sistema transaccionales no se ha expandido.

El presente trabajo tiene por objetivo ilustrar cómo es la problemática de incorporar firma digital a un sistema mostrando qué cambios origina tanto en lo funcional como en la arquitectura del mismo y a la estructura de la base de datos

Este trabajo esta organizado en cuatro módulos. El primer módulo contiene los conceptos teóricos necesarios para introducirnos en firma digital y manejo de directorios.

El siguiente módulo detalla las tecnologías utilizadas, las cuales fueron integradas para montar la infraestructura PKI necesaria para el desarrollo de este trabajo.

El tercer módulo describe la aplicación tomada como base para incorporarle firma digital; detalla cuales son los cambios introducidos en la misma a raíz de dicha incorporación y cómo es resuelto el uso de firma digital para lograr el objetivo planteado.

Por último, el cuarto módulo contiene las conclusiones y líneas futuras de trabajo.

Módulo 1

Conceptos Teóricos

Introducción

Introducción

En la actualidad se usa Internet para realizar miles de transacciones en línea. Por ejemplo, muchos empleados comparten archivos e información confidencial vía correo electrónico o a través de la red; los clientes de un banco actualizan sus cuentas o pagan impuestos desde la computadora en sus hogares y una gran variedad de productos son adquiridos a través de formularios en línea. Debido a esto, la informatización de los procesos puede significar un notable ahorro de recursos materiales y humanos.

Por otro lado, la tecnología de Internet también puede ser usada para otros fines como, por ejemplo: interceptar y falsificar mensajes, robar información importante y espiar a organizaciones e individuos.

Lógicamente, conforme más información hay disponible en Internet, más importancia adquiere la protección, la organización y el control del acceso a esa información. Ante esta situación surge la necesidad de brindar seguridad en los datos, los servicios, las transacciones y las partes involucradas.

Dado que distintos servicios de Internet ofrecen protección, organización y control del acceso a la información por separado, será necesario integrarlos para poder brindar en un mismo producto seguridad en la transmisión de datos, autenticidad[1] de cada usuario, y una manera clara y consistente de organizar y manipular información a través de un servicio de directorios.

Con este fin, se investigó la forma de combinar distintas tecnologías para adecuar el módulo de docentes del sistema SIU-Guaraní [Ref. 1], el cual consiste en brindar servicios, a través de la web, a los profesores de una determinada unidad académica para que puedan realizar los trámites administrativos de manera cómoda, eficiente y segura.

En Argentina, desde 1998 el gobierno promocionó el uso de firma digital, sin embargo, hoy en día, hay muy pocas aplicaciones que utilizan firma digital más allá de sitios web y correo electrónico.

Actualmente, alrededor de 150 unidades académicas del país utilizan el sistema de gestión de alumnos SIU-Guaraní, desarrollado por el Ministerio de Educación de la Nación, el cual brinda a los docentes un módulo de acceso por Internet. Se eligió esta aplicación para incorporar el uso de firma digital a ciertas operaciones de la misma, ya que se encuentra ampliamente difundida. De este modo, se da la posibilidad de eliminar un control interno, evitando la impresión de actas de examen y/o cursadas para su firma y traslado a la oficina de alumnos y posterior foliado en el libro de actas correspondiente.

Para este fin, configuraremos una autoridad de certificación que se encargará de la gestión de certificados digitales para los usuarios del sistema SIU-Guaraní.

Para ello, todas las partes involucradas confiarán en los documentos digitales generados y los utilizarán para completar el circuito administrativo correspondiente. Con esto se genera una relación de confianza entre dichas partes.

[1] *Autenticidad*: garantiza que el mensaje proviene de la persona que efectivamente lo está enviando.

Al incluir firma digital en un circuito administrativo, debemos tener en cuenta que el envío de un documento firmado requiere la verificación de la firma por parte del receptor. Si este no pudiese verificar la firma, ya sea por no tener los medios digitales necesarios o porque el circuito administrativo requiere, por ejemplo, imprimir el documento, la firma digital carecería de sentido.

La utilización de firma digital en una aplicación agrega *certeza* ya que permite comprobar que la persona que realizó la operación es quien dice ser. Cómo consecuencia de esto, las acciones realizadas son *auditables*. Además, agrega el manejo digital de objetos digitales, dado que los documentos generados con firma digital se usan en forma digital, por ejemplo, si generamos un acta firmada digitalmente, no será necesaria una impresión de la misma para que tenga validez. Todo esto trae aparejado *consentimiento explícito* en cada momento, lo cual significa que cada documento debe ser firmado explícitamente y que la firma de uno no implica la firma de los siguientes.

Cuando una aplicación utiliza firma digital, requiere la capacidad de manejar objetos digitales. Dichos objetos tienen como atributo la firma digital, ésto produce modificaciones en la base de datos, ya que ahora los sujetos de validación por firma digital deben ser almacenados. Además, se producen cambios en los procesos y en la auditoría; ésto implica realizar cambios a nivel tecnológico. Para poder instrumentar la firma digital se deben utilizar distintas herramientas. En nuestro caso las herramientas son: OpenCa, OpenLdap, Php, Applets para firma de lado del cliente, Apache, IIS, etc.

Los procesos de auditoría de una aplicación se ven modificados al utilizar firma digital, debido a que los documentos para realizarla están almacenados digitalmente y se tomarán desde ahí.

Cabe destacar que en nuestra facultad se han desarrollado tesis demostrando mecanismos para implementar firma digital en general [Ref. 2] y para aplicaciones construidas con J2EE [Ref. 3].

La presente tesis tiene por objetivo mostrar el uso de firma digital en una aplicación real de uso extendido y mostrar el impacto que esto tiene en el desarrollo del sistema y su estructura interna, principalmente en la base de datos y en la incorporación de una base de datos para documentos firmados, así como el resultado en la usabilidad del sistema.

Referencias

[Ref. 1] www.siu.edu.ar
[Ref. 2] Trabajo de grado"Usando Firma Digital" Paula Venosa, Verónica Fredes
[Ref. 3] Trabajo de grado "Implementando Firma Digital con J2EE" de Alejandro Falcone y Maria Clemens

Firma Digital

Firma digital

Introducción

Desde el punto de vista técnico, como alternativa a la firma manuscrita sobre papel se ofrecen las firmas electrónicas y/o digitales.

Cuando se realiza una oferta o se acepta un contrato, es habitual que cada parte coloque algún identificador personal para mostrar que conocen el contenido del documento y lo aceptan. Tradicionalmente este identificador es una firma escrita. Como en el mundo digital no es posible realizarlo de esta forma es necesario contar con alguna alternativa; ésto no significa que la simple digitalización de una firma escrita sea un procedimiento adecuado, ya que es muy fácil que una tercera parte la copie y la reproduzca para adjuntarla a un documento electrónico.

*La **firma digital*** es un conjunto o bloque de caracteres que viajan junto a un documento, archivo o mensaje, la misma es capaz de acreditar quién es el autor o emisor del mismo (lo que se denomina autenticación[1]) y que nadie haya manipulado o modificado el mensaje en el transcurso de la comunicación (asegura la integridad[2] del mensaje).

Las funciones principales de seguridad que debe tener cualquier transacción son:

1) Confidencialidad - para mantener la información privada.
2) Integridad- para probar que la información no ha sido manipulada.
3) Autenticación- para probar la identidad de un individuo o aplicación.
4) No repudio- para asegurar que no se pueda negar el origen de la información

Las firmas digitales consisten básicamente en la aplicación de algoritmos de encriptación a los datos. La criptografía es el estudio de los métodos para enviar mensajes en forma encubierta para que sólo el destinatario pueda remover la máscara y leer el mensaje. La criptografía asegura la confidencialidad, encriptando un mensaje y usando una clave secreta en asociación con un algoritmo. Esto produce un mensaje ilegible que solamente el destinatario puede desencriptar, usando la clave original. La clave deberá permanecer secreta por ambas partes.

La encriptación se realiza utilizando un algoritmo específico para tal fin. La seguridad del algoritmo va en relación directa a su tipo, tamaño, tiempo de cifrado y a la no violación del secreto.

Características de la firma digital

Algunos aspectos a tener en cuenta en relación con la firma digital son:

- Debe permitir la identificación del que firma. Definimos el concepto de "autoría electrónica" como la forma de determinar que una persona es quien dice ser.

- No puede ser generada más que por el emisor del documento, infalsificable e inimitable.

---

[1] *Autenticidad*: garantiza que el mensaje proviene de la persona que efectivamente lo está enviando.
[2] *Integridad*: asegura que el contenido del documento no ha sido modificado.

- Los datos generados a partir de la firma electrónica deben ser suficientes para poder validarla, pero insuficientes para falsificarla.

- La posible intervención de una Autoridad de Certificación mejorará la seguridad del sistema.

- El agregado de una firma debe ser significativo y va unido indisociablemente al documento al que se refiere.

- No deben existir diferencias de tiempo ni de lugar entre aceptación por el firmante y el agregado de la firma a un documento.

Infraestructura de clave pública

Infraestructura de clave pública (PKI[3]) es el término utilizado para referirse a la infraestructura de seguridad, basada en criptografía de clave pública, que permite la gestión de **certificados digitales**. Un certificado es un documento digital que identifica a la persona. De la misma manera que el Documento Nacional de Identidad nos identifica ante el Estado, el certificado digital nos identifica unívocamente en cualquier transacción electrónica que realicemos. Es el "DNI digital" que necesitamos para navegar de forma segura por Internet.

En otras palabras, un certificado es un documento firmado digitalmente por una persona o entidad confiable denominada Autoridad de Certificación (CA), que vincula cierta información perteneciente a un sujeto con su clave pública.

En los procedimientos de gestión e intercambio seguro de documentos electrónicos, los certificados digitales garantizan la validez de las claves públicas utilizadas y su vinculación con el sujeto propietario.

Por tanto, en una Infraestructura de clave pública, tendremos que definir y establecer todos los métodos necesarios para gestionar los certificados digitales de forma óptima. Principalmente, hay que establecer procedimientos para:

1) Emitir certificados digitales.
2) Revocar certificados digitales.
3) Consultar certificados digitales.

El objetivo de la infraestructura **PKI**[3] [Ref. 1] es dotar a los miembros de una corporación de los mecanismos básicos de seguridad que ésta necesita, esto es, autenticación, integridad, confidencialidad y no repudio, tanto para las conexiones web (con el protocolo SSL[4]) como para las comunicaciones a través de correo electrónico (con el protocolo S-MIME[5] [Ref. 2]), entre otros usos.

PKI incluye servicios y protocolos de manejo de clave pública a través de la autoridad de certificación y de la autoridad de registración, pero no provee necesariamente opciones criptográficas en las claves. La definición más compleja entiende a PKI como los servicios de firma digital y criptografía provistos a las aplicaciones de usuario final.

Los servicios basados en *firma digital deben estar bien seguros que cada vez que ellos confían en una clave pública, la clave privada es un secreto para todos excepto para el emisor de la información.* Esta confianza se basa en el uso de certificados de clave pública los cuales son estructuras de datos que ligan valores de clave pública a los sujetos. La ligadura se realiza a través de una autoridad de certificación, la cual verifica la identidad del sujeto y firma digitalmente cada certificado.

Una PKI consta de cinco componentes fundamentales:

1) *Autoridad de certificación*, que emiten y revocan certificados
2) *Autoridad de registración*, que atestiguan la asociación entre la clave pública y la entidad propietaria del certificado.

---

[3] *PKI*: Public Key Infraestructure. Infraestructura de clave pública.
[4] *SSL: Secure Socket Layer*
[5] *S-MIME:* (Secure / Multipurpose Internet Mail Extensions) es un protocolo que añade firmas digitales y encriptación a los mensajes MIME. MIME es el formato estándar propuesto para correo electrónico

3) *Poseedores de certificados emitidos* que pueden firmar documentos digitales.
4) *Entidad final,* usuarios de certificados PKI y/o usuarios del sistema que son sujetos de un certificado.
5) *Repositorios,* que guardan y hacen disponibles tanto certificados como listas de revocación de certificados.

¿Por qué usar PKI?

Una infraestructura de clave pública permite realizar operaciones a través de Internet alrededor del mundo, protegiendo la información de la interceptación, manipulación o acceso no autorizado. Además, PKI protege la información de diferentes maneras:

- *Autentica la identidad, la integridad, la privacidad y el no repudio* por el uso del certificado digital
- *Autoriza transacciones*: Con las soluciones PKI, cualquier organización controlará las transacciones hechas con terceros, con la seguridad de saber quien es esa persona o institución.
- *Disminuye el riesgo de manejo de información.*

Referencias:

[Ref. 1] http://www.entrust.com/downloads/docs/protocols_pki.html
[Ref.2] http://ca.banesto.es/ayuda/faqs/imprime/faq3.html

LDAP

Ligthweight Directory Access Protocol

LDAP

Introducción

Hoy en día, la gente y los negocios están cada vez más relacionados con los sistemas de redes de computadoras para soportar aplicaciones distribuidas. Estas aplicaciones distribuidas necesitan interactuar con computadoras que están en una LAN[1] o en una WAN[2]. Para mejorar la funcionalidad, facilitar el uso y permitir un costo razonable en la administración de la información de aplicaciones distribuidas, teniendo en cuenta los servicios, recursos, usuarios y otros objetos accesibles desde las aplicaciones, es necesario organizar toda esta información de manera clara y consistente. Mucha de esta información puede ser compartida entre diferentes aplicaciones, pero debe ser protegida para prevenir modificaciones no autorizadas o revelar información privada.

La descripción de la información de varios usuarios, aplicaciones, archivos, impresoras, y otros recursos accesibles desde una red, es a veces colocada en una base de datos especial, llamada *directorio*. Como el número de diferentes redes y aplicaciones crece, el número de directorios especializados de información también crece, resultando ésto en islas de información que no pueden ser compartidas y son difíciles de mantener. Si toda esta información pudiera ser mantenida y accedida de una manera consistente y controlada, se lograría la integración de un entorno distribuido en un sistema consistente y completo.

Ligthweight Directory Access Protocol (LDAP) es un estándar abierto que ha evolucionado para dar respuesta a estas necesidades, define un método estándar para acceder y actualizar información en un directorio y se ejecuta sobre TCP/IP.

LDAP está ganando aceptación como el método de acceso de directorios de la Internet y también está transformando las estrategias en las intranets corporativas. Esto está siendo soportado por un número creciente de vendedores de software e incorporado en un número mayor de aplicaciones.

LDAP fue diseñado en la Universidad de Michigan para adaptar un sistema de directorio complejo (llamado X.500[3]) a la Internet moderna. Un servidor de directorios se ejecuta sobre una computadora host en Internet, y muchos programas clientes que entienden el protocolo pueden interactuar con dicho servidor.

¿Qué es un directorio?

Un *directorio* es un listado de información acerca de objetos ordenados con algún criterio que brinda detalles de cada objeto. Ejemplos comunes son: un directorio telefónico de una ciudad o un catálogo de biblioteca. Para un directorio telefónico, los objetos listados son personas; los nombres están ordenados alfabéticamente, y los detalles dados para cada persona son dirección y número de teléfono.

En términos informáticos, un directorio es una base de datos especializada, también llamada repositorio de datos, que almacena información tipada y ordenada acerca de los objetos.

Los directorios permiten a los usuarios o aplicaciones, buscar recursos que tienen características necesarias para una tarea en particular. Por ejemplo, un

---

[1] *LAN*: **L**ocal **A**rea **N**etwork o red de área local.
[2] *WAN*: **W**ide **A**rea **N**etwork o red de área amplia.
[3] *X.500*: Servicio de directorio estándar. Ver página 5

directorio de personas puede ser usado para buscar la dirección de correo electrónico de una persona o su número de fax. Un directorio de servidores de aplicación, puede ser recorrido para buscar un servidor que pueda acceder a cierta información de un cliente.

Los términos *páginas blancas* y *páginas amarillas* son a veces utilizados para describir como es empleado un directorio. Si el nombre de un objeto (persona, impresora) es conocido, sus características pueden ser retornadas (número de teléfono, páginas por minuto); ésto es similar a buscar un nombre en las páginas blancas de un directorio telefónico. Si no se conoce el nombre de un objeto individual, se puede buscar en el directorio una lista de objetos que cumplan cierto requerimiento; ésto es como buscar una lista de peluquerías en las páginas amarillas de un directorio telefónico. De todos modos, los directorios almacenados en una computadora son mucho más flexibles que las páginas amarillas, porque son usualmente utilizados para buscar según algún criterio, no sólo por un conjunto de categorías predefinidas.

Diferencias entre Directorios y Bases de datos

Un *directorio* es una base de datos especializada que posee características que la diferencian de las bases de datos relacionales. Una característica importante consiste en que los directorios son accedidos para lectura o búsqueda, muchas más veces que para actualización.

Debido a que los directorios deben ser capaces de soportar gran cantidad de accesos de lectura, están optimizados para ello. El acceso para escritura puede estar limitado al administrador del sistema o al propietario de cada parte de la información.

Ya que los directorios se aplican al almacenamiento de información relativamente estática y están utilizados para ese propósito, no son apropiados para almacenar información que cambia rápidamente. Por ejemplo, el número de trabajos concurrentes en una cola de impresión probablemente no deba ser cargado en una entrada de directorio para una impresora, porque la información tiene que ser actualizada frecuentemente. En cambio, la entrada de directorio para una impresora puede contener la dirección de red de un servidor de impresión. El servidor de impresión puede ser consultado para saber la longitud actual de la cola. La información en el directorio (la dirección del servidor de impresión) es estática, mientras que el número de trabajos en la cola de impresión es dinámico.

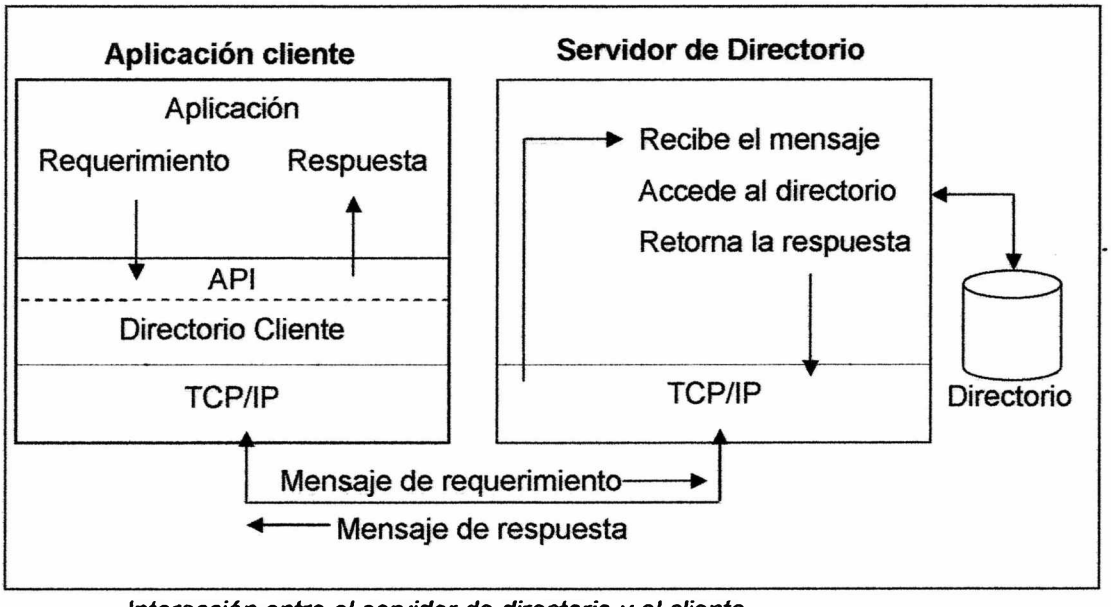
Otra diferencia importante entre los directorios y las bases de datos de propósito general, es que los directorios pueden no soportar transacciones (aunque algunas implementaciones lo hacen). Las transacciones son operaciones atómicas, ésto significa que las operaciones que implica la transacción no pueden ser realizadas parcialmente.

Como los directorios tratan mayormente con requerimientos de lectura, la complejidad de las transacciones puede ser evitada. Si dos personas intercambian oficinas, sus entradas al directorio necesitan ser actualizadas con nuevos números de teléfono, dirección, etc. Si una entrada es actualizada, y la otra se actualiza después, existe un período en el cual el directorio mostrará que dos personas tienen el mismo número de teléfono. Como las actualizaciones son poco frecuentes, tales anomalías son consideradas aceptables.

El tipo de información almacenada en un directorio usualmente no requiere consistencia estricta, por ésto es aceptable que alguna información esté temporalmente desactualizada. Como son no transaccionales, no deberían ser usados para almacenar información sensible a inconsistencias.

Otra diferencia importante entre los directorios y las bases de datos de propósito general es la forma en que la información puede ser accedida. Muchas bases de datos soportan un estándar, con un método de acceso muy poderoso llamado SQL[4], este permite funciones de actualización y consulta complejas a costo del tamaño del programa y complejidad de la aplicación. Los directorios LDAP en cambio, usan un protocolo de acceso simplificado y optimizado que puede ser utilizado en aplicaciones relativamente simples y pequeñas.

Los directorios son usualmente accedidos utilizando el modelo de comunicación cliente/servidor. Una aplicación que quiere escribir o leer información en un directorio no accede al directorio directamente. En lugar de esto, llama a una función o a una API[5] que genera un mensaje a ser enviado a otro proceso. Este segundo proceso accede a la información en el directorio en nombre del requerimiento de la aplicación. Los resultados de leer o escribir son entonces retornados a la aplicación que hizo el requerimiento.



Interacción entre el servidor de directorio y el cliente

El requerimiento es realizado por el directorio cliente, y el proceso que busca información en el directorio es llamado servidor de directorio. En general, los servidores proveen un servicio específico a clientes. A veces un servidor puede ser el cliente de otros servidores con el fin de reunir la información necesaria para procesar un requerimiento.

[4] *SQL*: **S**tructured **Q**uery **Lenguaje** o lenguaje estructurado de consulta. lenguaje estándar de comunicación con bases de datos.
[5] *API*: **A**pplication **P**rogramming **I**nterface, define la interfaz de programación de un lenguaje de programación en particular usada para acceder a un servicio.

Seguridad

La seguridad de la información almacenada en un directorio es muy importante.

Algunos directorios son utilizados para ser accedidos públicamente en la Internet, pero no cualquier usuario debería ser capaz de realizar cualquier operación. Por ejemplo, si una compañía tiene un servicio de directorio en su intranet, no basta sólo con tener seguridad ante los usuarios externos a la intranet sino que también se debe contar con seguridad para los usuarios propios de la intranet.

Una política de seguridad define quien tiene que tipo de acceso a que información. La política de seguridad es definida por la organización que mantiene el directorio.

Un directorio debe soportar las capacidades básicas necesarias para implementar una política de seguridad. El directorio puede no proveer directamente características de seguridad subyacentes, pero puede estar integrado con un servicio confiable de seguridad de red que provee los servicios de seguridad básicos.

Es necesario un método para autentiçar usuarios; la autenticación verifica que los usuarios son quienes dicen ser. Un esquema de autenticación básico consiste en un nombre de usuario y una clave. Una vez que los usuarios están autenticados se debe determinar si tienen la autorización o permiso para realizar la operación requerida sobre el objeto específico.

La autorización está a veces basada en listas de control de acceso (ACLs). Una ACL es una lista de autorizaciones que puede ser vinculada a objetos y atributos en el directorio. Una lista ACL tipa los accesos por usuario; para crear una lista corta y manejable, los usuarios con los mismos derechos de acceso son incluidos en grupos de seguridad. La siguiente tabla muestra un ejemplo de una lista de control de acceso para una entrada de directorio de empleados:

Usuario o grupo	Derecho de acceso
Propietario	Leer, modificar (pero no borrar)
Administradores	Todo
Personal	Leer todos los campos
Otros usuarios	Lectura restringida

Un directorio que es accedido por todas las aplicaciones es una parte vital de la infraestructura soportada por un sistema distribuido.

Un servicio de directorio provee una vista lógica simple de los usuarios, recursos y otros objetos que forman parte de un sistema distribuido; lo cual permite a los usuarios y aplicaciones acceder transparentemente a los recursos de la red. Esto significa que el sistema es percibido como un todo integrado, no como una colección de partes independientes. Los objetos pueden ser accedidos por nombre o función sin conocer los identificadores de bajo nivel tales como direcciones de host, nombre de servidores de archivo, etc.

Los permisos son configurados por el administrador para permitir que sólo ciertos usuarios accedan a la base de datos LDAP, y opcionalmente puedan mantener ciertos datos privados. Los servidores LDAP también proveen servicios de

autenticación, entonces los servicios de web, correo electrónico y archivos compartidos, por ejemplo, pueden usar una lista simple de usuarios y claves autorizados.

X.500: servicio de directorio estándar

Debido a que LDAP está basado en el protocolo X.500 [Ref.1], en esta sección se detallarán sus principales características.

El estándar X.500 fue creado por el CCITT[6] en 1998, el cual se transformó en la ISO 9594.

X.500 organiza las entradas del directorio en un espacio de nombre jerárquico capaz de soportar grandes cantidades de información. También define poderosas herramientas de búsqueda para recuperar fácilmente información. Debido a su funcionalidad y escalabilidad, X.500 es frecuentemente usado con módulos diseñados para la interoperación entre servicios de directorio incompatibles.

X.500 especifica que la comunicación entre el directorio cliente y el directorio servidor usa el protocolo de acceso a directorios (DAP). Sin embargo, como un protocolo de nivel de aplicación, el DAP requiere la pila de protocolos OSI entera para operar. Soportar la pila de protocolos OSI requiere más recursos que los que están disponibles en algunos entornos pequeños. Por lo tanto es deseable una interfaz para un servidor de directorios X.500 que use menos cantidad de recursos o protocolos lightweight.

LDAP fue desarrollado como una alternativa lightweight para DAP. LDAP requiere la pila de protocolos TCP/IP[7] en lugar de la pila de protocolos OSI. LDAP también simplifica algunas operaciones X.500.

LDAP define el protocolo de comunicación entre el directorio cliente y el servidor, pero no define una interfaz de programación para el cliente.

La API de LDAP define una API en lenguaje C para el acceso a un directorio usando LDAP v2, la cual no es un estándar oficial [Ref. 2].

LDAP: ¿protocolo o directorio?

LDAP define un protocolo de comunicación. Define el transporte y formato de mensajes usados por un cliente para acceder a datos en un directorio X.500-like. LDAP no define el servicio de directorio en sí mismo.

Una aplicación cliente inicia un mensaje LDAP llamando a una API LDAP pero un servidor de directorio X.500 no entiende mensajes LDAP. En realidad, el cliente LDAP y el servidor X.500 usan diferentes protocolos de comunicación (TCP/IP vs. OSI[8]). EL cliente LDAP realmente se comunica con un proceso gateway (también llamado proxy o front end) que reenvía los requerimientos al servidor de directorios X.500. Este gateway es conocido como servidor LDAP, que atiende

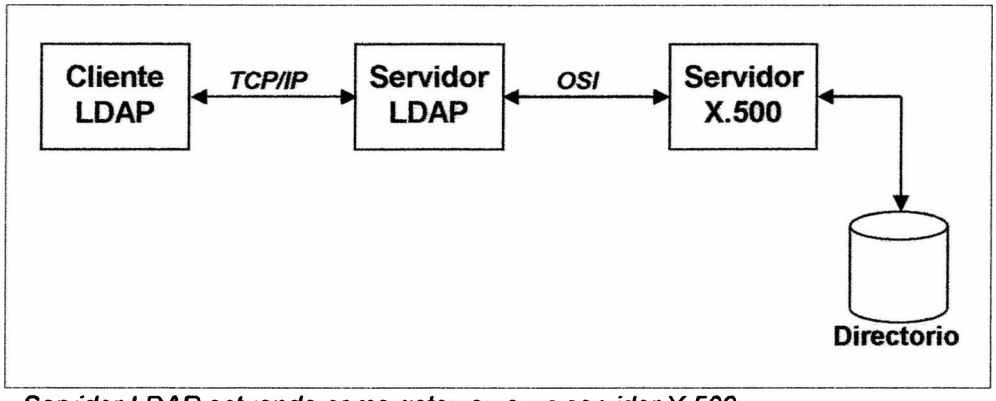
---

[6] *CCITT*: **C**omité **C**onsultatif **I**nternationale de **T**elegraphie y **T**elephonie

[7] *TCP/IP*: Protocolo de comunicación compuesto por dos protocolos: Protocolo de Control de Transmisión (TCP) y Protocolo de Internet (IP)

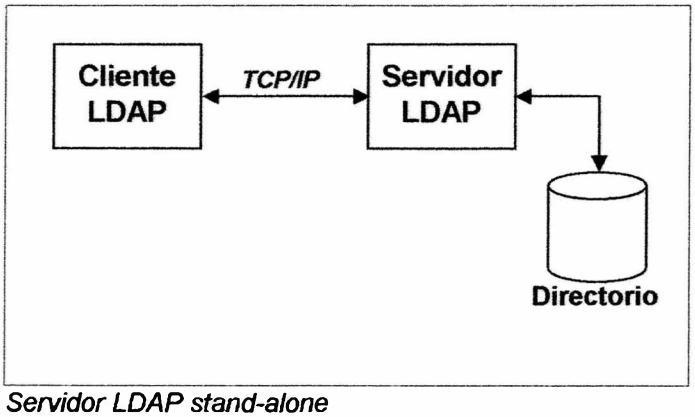
[8] *OSI*: El **modelo de referencia de Interconexión de Sistemas Abiertos** (OSI) lanzado en 1984 fue el modelo de red descriptivo creado por ISO (Organización Internacional para Estandarización). Proporcionó a los fabricantes un conjunto de estándares que aseguraron una mayor compatibilidad e interoperabilidad entre los distintos tipos de tecnología de red producidos por las empresas a nivel mundial.

requerimientos a un cliente LDAP. Este se convierte en un cliente del servidor X.500. El servidor LDAP debe comunicarse usando TCP/IP y OSI.



Servidor LDAP actuando como gateway a un servidor X.500

Se puede tener directamente un servidor LDAP que almacene y acceda el mismo al directorio sin la necesidad de utilizar un servidor X.500 (por consiguiente no es necesario tener la pila de protocolos OSI). Por supuesto, ésto hace al servidor LDAP mucho más complejo ya que debe almacenar y retornar entradas del directorio. Estos servidores LDAP son llamados servidores LDAP stand-alone porque no dependen de un servidor de directorios X.500. Como LDAP no soporta todos los servicios soportados por un servidor X.500, un servidor LDAP stand-alone sólo necesita soportar los servicios requeridos por LDAP.



Servidor LDAP stand-alone

Los servidores LDAP existen en tres niveles:

- hay grandes servidores públicos, como BigFoot e Infospace,
- servidores organizacionales en universidades y corporaciones,
- servidores LDAP pequeños para grupos de trabajo.

La mayoría de los clientes de correo electrónico corrientes están configurados para buscar direcciones de correo electrónico en un directorio LDAP (Outlook, Eudora, Netscape, etc.).

LDAP tiene aplicaciones generales, tal como servicios y dispositivos de búsqueda en Internet e intranets.

Netscape Communicator (versión 4.5 y posteriores) puede almacenar preferencias de usuario y bookmarks en un servidor LDAP.

Los servidores LDAP ordenan todos los datos en sus entradas, pueden usarse "filtros" para seleccionar los objetos, atributos o grupos que se desee, y retorna exactamente la información que se solicita. Por ejemplo, una búsqueda LDAP traducida sería: "Buscar todas las personas que vivan en La Plata cuyo nombre contenga "Juan" que tengan una dirección de correo electrónico. Por favor, retornar su nombre completo, correo electrónico y descripción".

LDAP: Implementación

Algunas características de LDAP:

LDAP es orientado a objetos: Para representar los atributos de una entidad se puede utilizar entre otros el tipo de datos ObjectClass, cuyo valor representa el nombre de una clase de objetos. Estas clases definen un conjunto de atributos que pueden o deben ser una entrada, o extienden la definición heredada desde otra clase.

Una clase de objetos tiene atributos que son obligatorios, es decir, deben tener si o si un valor, y atributos que son opcionales, pueden o no tener un valor.

Si se necesita utilizar una clase con atributos que no están en ninguna de las clases predefinidas se pueden agregar nuevas clases a partir de las clases existentes.

LDAP tiene sus orígenes en el mundo de las bases de datos: Una colección de clases de objetos que especifican atributos para las entradas en un servidor LDAP se llama ***esquema***. El objeto ObjectClass es la raíz de la base de datos LDAP.

LDAP no está limitado a almacenar información en estructuras de árbol estrictas: sugiere una estructura de árbol jerárquica, pero no la exige. Es importante tener esto en mente cuando su código evita crear suposiciones incorrectas acerca de la jerarquía de datos en el servidor.

Arquitectura de LDAP

La arquitectura de LDAP define el contenido de mensajes intercambiados entre un cliente y un servidor LDAP. Los mensajes especifican las operaciones requeridas por el cliente (buscar, modificar, borrar, etc.), las respuestas desde el servidor y el formato de datos acarreados en los mensajes. Los mensajes LDAP son enviados sobre TCP/IP, un protocolo orientado a la conexión; entonces hay también operaciones para establecer y desconectar una sesión entre cliente y el servidor.

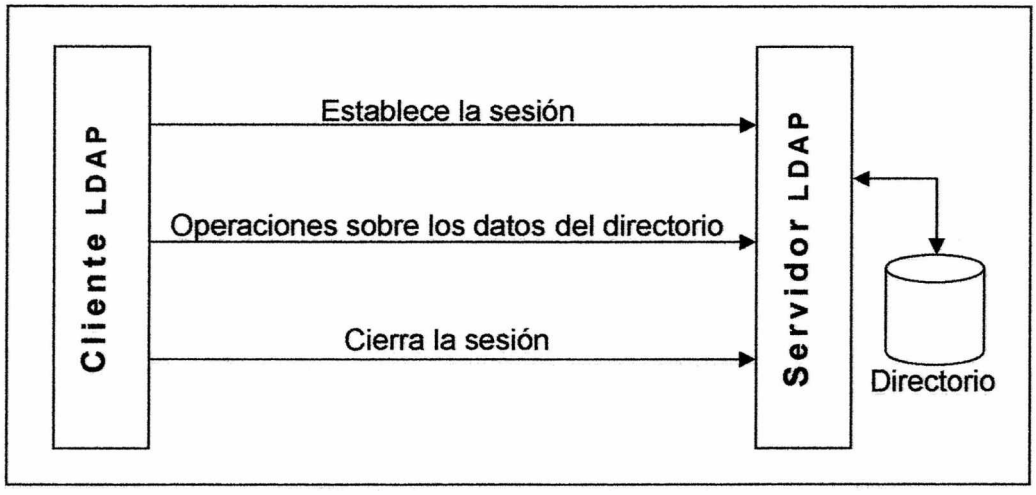
Para el diseñador de un directorio LDAP lo más importante es el modelo lógico que esta definido por los mensajes y tipos de datos, cómo está organizado el directorio, qué operaciones son posibles, como está protegida la información, etc.

La interacción general entre un cliente y un servidor LDAP tiene la siguiente forma:

- El cliente establece una sesión con un servidor LDAP, especificando el nombre o dirección IP y número de puerto TCP/IP donde está escuchando el servidor LDAP.

El cliente puede proveer un nombre de usuario y una clave para autentificarse con el servidor o conectarse anónimamente. El cliente y el servidor pueden también establecer una sesión segura a través de encriptación de datos.

- El cliente realiza operaciones sobre los datos del directorio. LDAP permite lecturas y actualización, ésto hace que la información sea manejada mediante consultas. LDAP también permite búsquedas en el directorio de datos usando criterios arbitrarios establecidos por los clientes.
- Cuando un cliente no tiene más requerimientos cierra la sesión con el servidor.



Interacción general entre un cliente LDAP y un servidor LDAP

Los tres puntos anteriores no están definidos por el protocolo LDAP y la arquitectura en sí misma. El encargado de crear, mantener y terminar la conexión es el protocolo de comunicación entre el servidor y el cliente de LDAP. Esto es bien conocido por la API LDAP que permite que las aplicaciones interactúen fácilmente con los servidores LDAP. La API puede ser considerada una extensión de la arquitectura LDAP.

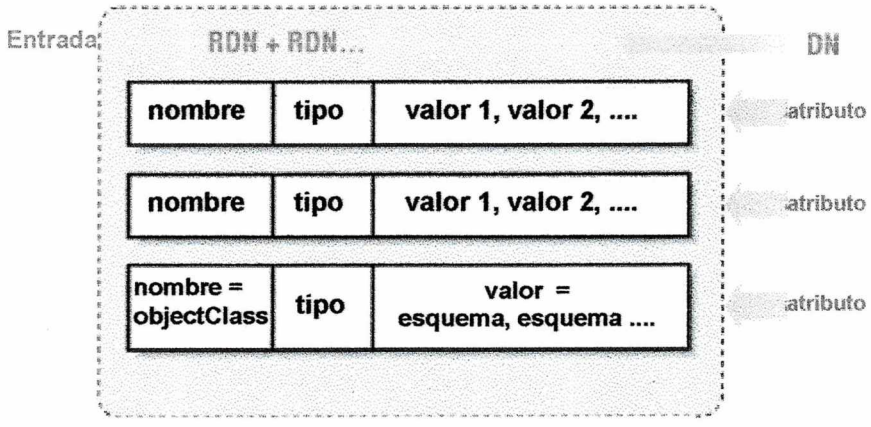
El modelo LDAP

LDAP puede ser entendido mejor considerando los cuatro modelos sobre los cuales está basado:

- Información: describe la estructura de información almacenada en un directorio LDAP.
- Nombrado (Naming): describe cómo es organizada e identificada la información en un directorio LDAP.
- Funcional: describe qué operaciones pueden ser realizadas sobre la información almacenada en un directorio LDAP.
- Seguridad: describe cómo la información de un directorio LDAP puede estar protegida contra accesos no autorizados.

Modelo de información

LDAP utiliza una estructura de datos conocida como ***entrada*** para almacenar la información en el directorio. La figura siguiente muestra las partes que componen una *entrada.*



Estructura de datos de una entrada de LDAP

Una entrada de un directorio usualmente describe un objeto tal como una persona, una impresora, etc. Cada entrada tiene un nombre denominado *nombre distinguido* (DN[9]) que la identifica unívocamente. El DN consiste en una secuencia de partes llamadas *nombres distinguidos relativos* (RDN[10]). Cada RDN es derivado de los atributos de la entrada del directorio, en el caso más simple un RDN tiene la forma <nombre_atributo>= <valor>. Las entradas pueden ser organizadas jerárquicamente en una estructura similar a la de árbol basado en sus DNs. Este árbol de entradas de directorios es llamado árbol de información de directorios (DIT[11])

Una ***entrada*** tiene un conjunto de componentes llamados ***atributos*** que contienen los datos de la entrada. Hablando en términos de bases de datos, son como los campos en un registro de base de datos.

Un ***atributo*** tiene un *nombre*, un *tipo* y un *conjunto de valores* que pertenecen a ese tipo. Por ejemplo, si se está almacenando información de empleados, la entrada debe tener un atributo teléfono que tenga un tipo numeroDeTelefono. Los valores de este atributo deben ser el número de teléfono de los empleados. Un tipo también tiene una sintaxis que determina que clase de datos pueden ser usados (textos, números, etc.), cómo está ordenado y cómo es usado en una búsqueda (¿es case-sensitive?). Un atributo puede contener más de un valor. Los nombres de los atributos son case-sensitive.

Se pueden asociar restricciones con los tipos de atributos para limitar el número de valores que pueden ser almacenados en el atributo o limitar el tamaño total de un valor.

Los ***esquemas*** definen el tipo de objetos que pueden ser almacenados en el directorio. Los esquemas también listan los atributos de cada tipo de objetos y cuando estos atributos son requeridos u opcionales. El chequeo de esquemas asegura que todos los atributos requeridos para una entrada estén presentes antes

---

[9] *DN*: Distinguished Name
[10] *RDN*: Realtive Distinguished Name
[11] *DIT*: Directory Information Tree

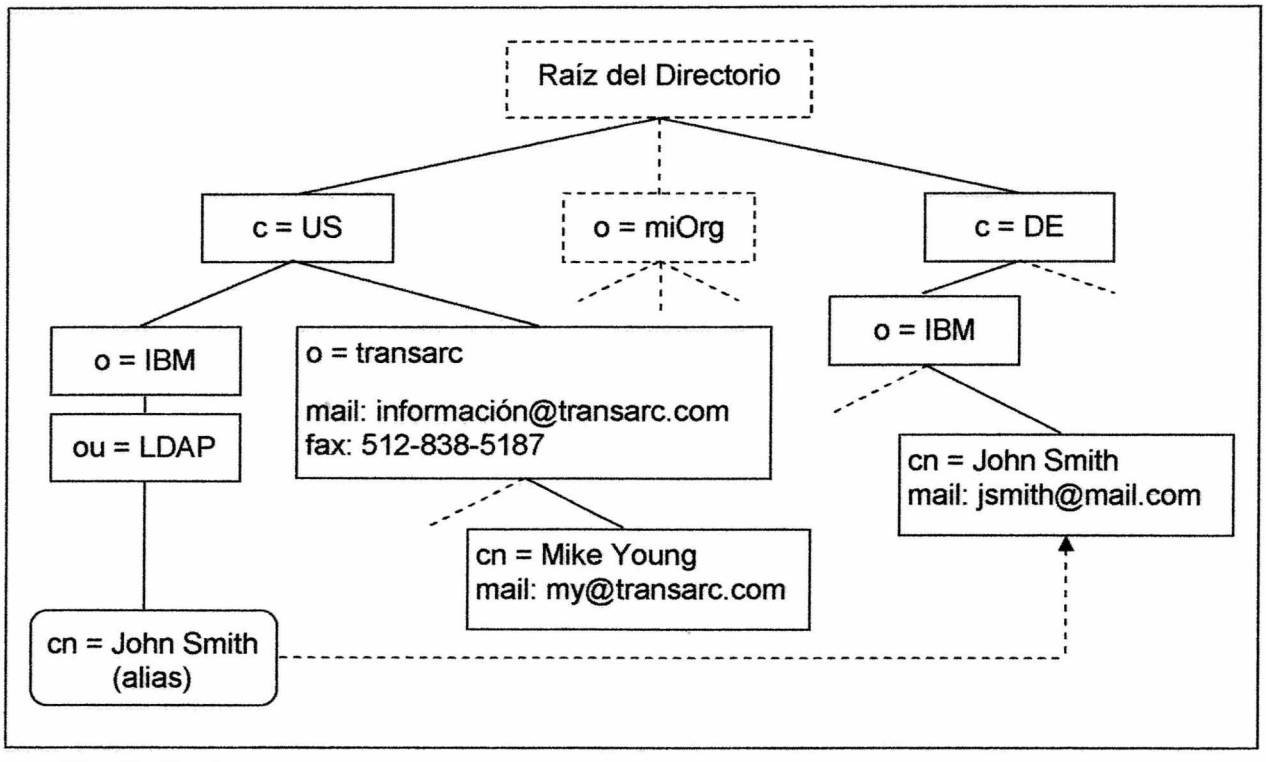
de que la entrada sea almacenada. El esquema también define la herencia y subclaseado de objetos y en qué lugar de la estructura DIT pueden aparecer los objetos. Cada servidor puede definir sus propios esquemas.

Cada entrada tiene un atributo especial que se llama objectClass. ObjectClass contiene muchos valores que, cuando se combinan con parámetros del servidor y del usuario, determinan qué atributos deben y pueden existir en esa entrada en particular.

Modelo de nombrado

Define cómo son identificadas y organizadas las entradas. Las entradas son organizadas en una estructura similar a un árbol llamada DIT, y son acomodas en árbol basados en sus DNs.

Por ejemplo, en la siguiente figura



Ejemplo de DIT

Cada cuadro representa una entrada al directorio. La entrada raíz es conceptual, no existe realmente. Los atributos son listados dentro de cada entrada.

La organización de las entradas en el DIT está restringida por las definiciones de clases de objetos correspondientes. Es usual seguir un esquema organizacional o geográfico. Las entradas son nombradas de acuerdo a su posición en el DIT. La entrada al directorio en la esquina inferior derecha de la figura tiene el DN: cn= John smith, o=IBM, c= DE. Los DNs se leen desde la hoja hacia la raíz.

El DIT es descripto como una estructura similar a un árbol pero no lo es; esto se debe a los alias. Los alias permiten que la estructura de árbol tenga ciclos, lo cual puede ser utilizado si una entrada pertenece a más de una organización o si un DN comúnmente usado es muy complejo.

Los DNs son usados como claves primarias para las entradas en el directorio. LDAP define una representación de strings orientada al usuario para los DNs.

Un servidor puede no almacenar el DIT entero, entonces los servidores necesitan estar enlazados para que el DIT entero sea formado por un directorio distribuido.

Modelo funcional

LDAP define operaciones para acceso y modificación de las entradas al directorio. Las operaciones de LDAP pueden ser divididas en tres categorías:

- Consulta: incluye búsquedas y operaciones de comparación usadas para retornar información desde el directorio.
- Actualización: incluye operaciones de agregado, borrado, modificación y modificación de RDN usadas para actualizar la información almacenada en el directorio.
- Autenticación: incluye el enlace y las operaciones usadas para conectar y desconectar desde y hacia el servidor LDAP, establecer derechos de acceso y protección de la información.

La operación más común es la búsqueda, ésta es muy flexible y tiene algunas opciones complejas.

A continuación se explicarán con más detalle las distintas operaciones que un cliente puede requerir a un servidor LDAP:

Búsqueda:

Es una operación que permite a un cliente pedirle a un servidor LDAP que busque en una porción del DIT cierta información según el criterio especificado por el usuario para leer y listar los resultados. La búsqueda puede ser muy general o muy específica. La operación de búsqueda permite especificar el punto de comienzo en el DIT, cuan profunda será la misma en el DIT, qué atributos de una entrada deben ser considerados y qué atributos se deben retornar de las entradas que cumplan la condición.

Comparación:

Es una operación que compara una entrada según un valor de atributo, si la entrada tiene ese valor la operación retorna verdadero, sino retorna falso.

Actualización:

Estas operaciones modifican el contenido del directorio. Algunas operaciones son las siguientes:

- Add: inserta nuevas entradas en el directorio.
- Delete: borra entradas existentes del directorio. Solo pueden ser borradas hojas.
- Modify: cambia los atributos y valores en una entrada existente. Permite que nuevos atributos sean agregados y que atributos existentes sean borrados o modificados.
- Modify DN: cambia el componente menos significativo de un DN o mueve un subárbol de entradas a una nueva localización en el DIT.

Autenticación:

Estas operaciones son usadas para establecer y finalizar una sesión entre un cliente LDAP y un servidor LDAP. La sesión puede ser asegurada en distintos niveles: desde una sesión anónima e insegura, una sesión autenticada en la cual el cliente se identifica con una clave, a un mecanismo seguro con una sesión encriptada usando SASL[12]. SASL fue agregada a LDAP versión 3 para mejorar la débil autenticación de LDAP versión 2. Las operaciones de autenticación son las siguientes:

- Bind: inicia una sesión LDAP entre un cliente y un servidor. Permite al cliente proveer su identidad para autenticarse con el servidor.
- Unbind: termina una sesión cliente / servidor.
- Abandon: permite a un cliente requerir que el servidor abandone una operación pendiente.

Controles y operaciones extendidas:

Permiten al protocolo LDAP ser extendido sin cambiar el protocolo en si mismo. Los controles modifican el comportamiento de una operación, y las operaciones extendidas agregan nuevas operaciones al protocolo LDAP.

Modelo de seguridad

Este modelo ésta basado en la operación de bind. Hay muchas operaciones de bind diferentes y su mecanismo de seguridad también es diferente. Una posibilidad es cuando el cliente en su requerimiento de acceso proporciona su DN identificador y su clave en un texto simple. Si no proporciona DN y clave es considerado una sesión anónima por el servidor LDAP. El uso de un texto simple para la clave es fuertemente discutido cuando el servicio de transporte subyacente no garantiza confiabilidad y puede entonces resultar en la divulgación de la clave a partes no autorizadas.

Adicionalmente, el bind Kerberos es posible en LDAP versión 2, pero esto fue discontinuado en LDAP versión 3. En lugar de ésto, LDAP versión 3 incorpora, con el comando bind, soporte con el mecanismo SASL.

Además, las operaciones extendidas del protocolo están disponibles en la versión 3 de LDAP. Una extensión relacionada con la seguridad es TLS[13].

Seguridad

Así como la seguridad cobra gran importancia en el mundo de las redes de computadoras, lo mismo ocurre para LDAP. El término seguridad, en el contexto de este documento, generalmente se utiliza para cubrir los siguientes aspectos: *autenticación, integridad, confiabilidad y autorización.* Los primeros tres aspectos son los que abarca LDAP versión 3 estándar. En cuanto a la autenticación, hay muchos métodos que pueden ser usados con este propósito, los más importantes son los siguientes:

- No autenticación
- Autenticación básica
- Autenticación simple y capa de seguridad (SASL)

No autenticación

[12] *SASL*: **S**imple **A**uthentication and **S**ecurity **L**ayer, es un framework de autenticación.
[13] *TLS*: **T**ransport **L**ayer **S**ecurity, se basa en SSL (Security Socket Layer) y crea sesiones encriptadas.

Este método debe ser utilizado cuando la seguridad de los datos no es un problema y cuando no están involucrados permisos especiales de control de acceso.

Esta opción es utilizada cuando se dejan los campos de DN y clave vacíos en la llamada a la operación bind de la API. El servidor de LDAP entonces asume automáticamente una sesión anónima de usuario y garantiza el acceso con el control apropiado definido para esta clase de acceso.

Autenticación básica

Este mecanismo de seguridad en LDAP es negociado cuando la conexión entre el cliente y el servidor es establecida.

Cuando se utiliza este método, el cliente se identifica con el servidor a través de su DN y clave. El servidor considera al cliente autenticado si el DN y la clave enviados por este coinciden con la clave para el DN almacenado en el directorio.

Autenticación simple y capa de seguridad (SASL)

SASL es un framework para agregar mecanismos de autenticación adicionales a los protocolos orientados a la conexión.

En SASL, los protocolos de conexión tales como LDAP, IMAP[14] y otros son representados por perfiles; cada perfil es considerado una extensión que permiten al protocolo y a SASL trabajar juntos. Cada protocolo que usa SASL necesita ser extendido con comandos para identificar el mecanismo de autenticación y para realizar la autenticación. Opcionalmente, la capa de seguridad puede negociar encriptación de datos después de la autenticación y así asegurar confidencialidad. LDAP versión 3 incluye tales comandos.

En la operación de bind de SASL, los parámetros clave que influyen en el método de seguridad usado son: *dn, mecanismo* y *credenciales*.

Como LDAP versión 2 no poseía métodos de encriptación de datos, algunos vendedores como Netscape o IBM, agregaron sus propias llamadas a SSL en la API de LDAP. Una potencial desventaja es que puede haber incompatibilidad entre diferentes vendedores. Por lo tanto, LDAP versión 3 propone incluir SSL o, más correctamente, su sucesor TLS, mediante operaciones extendidas al protocolo.

Tareas de administración de LDAP

Las herramientas básicas para la administración de un servidor LDAP se pueden categorizar de la siguiente manera:

- Configuración del servidor (creación inicial)
- Configuración del DIT
- Administración de los contenidos
- Configuración de la seguridad
- Administración de la replicación y referencias
- Administración del control de acceso
- Administración del archivo log y logging
- Administración de recursos y herramientas de análisis de performace

Por falta de herramientas de administración en los estándares, LDAP contiene algunos mecanismos básicos pero potentes que es importante conocer:

[14] *IMAP*: **I**nternet **M**essage **A**ccess **P**rotocol. Protocolo de red de acceso a mensajes electrónicos almacenados en un servidor. Mediante IMAP se puede tener acceso al correo electrónico desde cualquier equipo que tenga una conexión a Internet.

Herramientas de línea de comandos LDAP

La mayoría de los kits de desarrollo de software LDAP proveen un conjunto de aplicaciones de línea de comando de fácil uso para el cliente, que pueden realizar las operaciones básicas del protocolo LDAP como buscar en el directorio, modificar, agregar, borrar o renombrar entradas. Estas herramientas son también capaces de leer sus datos desde archivos LDIF[15] entonces pueden ser usadas para administrar directorios de datos voluminosos. También pueden ser usados con otros lenguajes de comando de programa, como los scripts de Unix o Perl.

Intercambio de formato de datos LDAP

LDAP soporta el intercambio de formato de datos LDAP el cual es muy conveniente como mecanismo de administración de datos. Este mecanismo posibilita una fácil manipulación de grandes cantidades de datos.

LDIF es usado para importar y exportar información del directorio entre servidores de directorios basados en LDAP.

La forma básica de la entrada de un directorio representada en LDIF es:

[<id>]
dn: <distinguished name>
objectClass: <object class>
objectClass: <object class>
...
<attribute type> [;languaje tag]: < attribute value>
<attribute type> [;languaje tag]: < attribute value>
...

Solamente el DN y el menor del object Class son requeridos. A continuación se muestra la definición previa de una entrada a un directorio en un archivo LDIF:

Campo	Definición
[<id>]	Número decimal positivo que representa el ID de la entrada. Las herramientas de creación de la base de datos generan este ID
dn : [<nombre distinguido>]	Especifica el nombre distinguido (distinguished name) para la entrada
objectClass : <clase de objeto>	Especifica un tipo de objeto para usar en la entrada. La clase de objeto identifica los tipos de atributos, o esquemas, permitidos y requeridos por la entrada
<attribute type>	Especifica un tipo de atributo para usar en la entrada
languaje tag	Especifica el lenguaje del texto en el valor del atributo
<attribute type>	Especifica el valor del atributo a ser usado con el tipo de atributo

Plataforma de soporte

La arquitectura de LDAP no depende de ningún sistema operativo o plataforma de hardware. Fue diseñado para correr sobre múltiples plataformas y

[15] *LDIF*: LDAP Data Interchange Format, intercambio del formato de datos LDAP

con recursos limitados. El prerrequisito es una pila de comunicación TCP/IP operativa.

Los conjuntos de herramientas para los clientes y las implementaciones para el servidor LDAP están disponibles para la mayoría de las plataformas de sistemas operativos y la funcionalidad de los clientes esta contenida ya en algunas aplicaciones.

Versiones recientes de navegadores sobre varias plataformas como Netscape's Communicator o Microsoft's Internet Explorer, son capaces de realizar búsquedas en directorios LDAP.

Administración y diseño de un directorio LDAP

Introducir LDAP en una organización involucra más que agregar otro protocolo. Requiere planeamiento acerca de cómo van a ser diseñados los contenidos del directorio y cómo debe ser desarrollado en la infraestructura física.

Crear un diseño que tenga la flexibilidad para acomodarse a los cambios de la organización es probablemente la tarea más importante en la implementación de un servicio de directorio.

Cuando se diseña un servicio de directorio, el proyecto puede ser divido en muchos proyectos más pequeños:

- Planear el contenido del directorio incluye decisiones sobre cómo almacenar los datos en el directorio y cómo serán acomodados en la estructura de árbol.
- La administración de datos y el control de acceso son importantes cuando se administra un servicio de directorios. Los planes deben ser hechos para identificar recursos para el mantenimiento de los datos actualizados y la identificación de recursos con la autoridad para decidir políticas de control de acceso con respecto a los datos del árbol de directorios.
- En el dimensionamiento del servicio de directorios, las consideraciones que deben ser tenidas en cuenta se refieren a qué clientes accederán a que datos, desde dónde y con qué frecuencia. Si hay aplicaciones de clientes que usan mucho el directorio deben asegurarse que la red esté disponible y que el ancho banda sea suficiente entre los servidores de aplicación y los servidores de directorio.

Definición del modelo de datos

Hay muchos pasos involucrados en el diseño de un árbol de directorio, tales como decidir que tipos de datos contendrán las entradas, que esquema se usará y finalmente, como serán acomodadas las entradas en la estructura de árbol. Durante el diseño, deben ser tomados en cuenta diferentes aspectos:

- ¿Qué tipo de aplicación o aplicaciones utilizarán el directorio?
- ¿El servicio de directorios de LDAP interactuará con el servicio de directorios X.500?
- ¿Cómo será mapeada la infraestructura de la organización en un directorio?
- ¿Cuáles serán los requerimientos de administración y escalabilidad?

Algunos tipos de datos son más apropiados que otros para un servicio de directorio. No deberían ser puestos en un directorio objetos de datos grandes y no estructurados no deberían ser puestos en el directorio.

Sabiendo que tipo de datos es usado en el servicio de directorios, para que será usado el directorio y como serán actualizados los datos, es posible comenzar a estructurarlos. Esto se hace a través del diseño de esquemas, eligiendo un sufijo de directorio, ramificando el árbol de directorios y finalmente, creando un estilo de nombres para la entrada de directorios.

Construyendo aplicaciones disponibles para LDAP

LDAP es una herramienta ideal para manejar grandes volúmenes de información, ya que es un protocolo poderoso y fácil de manejar, el cual puede ser ejecutado por un cliente disponible para LDAP, por ejemplo un manejador sobre varias plataformas.

Un cliente de LDAP puede ser cualquier tipo de aplicación, por ejemplo un procesador de texto el cual usa LDAP para verificar la dirección del receptor de una carta.

Para poder construir clientes de LDAP existen dos APIs las cuales están implementadas una en el lenguaje C y la otra en Java. La API en C es descripta, en la RFC 1823[16], que especifica la versión 2 del protocolo LDAP. Hay una nueva versión en construcción [Ref. 3], que especifica la versión 3 de LDAP y en un futuro hará obsoleta la RFC 1823.

No hay una RFC disponible para JNDI[17], el cual fue desarrollado por Sun Microsystems y es soportado por la mayoría de los vendedores, incluyendo Hewlett-Packard, Novell e IBM [Ref. 4].

Kit de desarrollo de software LDAP (SDKs)

Un SDK de LDAP es un conjunto de librerías y archivos. Está disponible para gran variedad de sistemas operativos incluyendo distintas plataformas Unix y Microsoft Windows. La mayoría de los SDKs incluyen herramientas de líneas de comando.

Hay vendedores que ofrecen servidores de directorios LDAP y también SDKs para permitir la comunicación entre servidores y aplicaciones, entre ellos están, por ejemplo Netscape e IBM. Un SDK disponible libremente en internet es el desarrollado por la Universidad de Michigan [Ref. 5].

Herramientas de línea de comando LDAP

La mayoría de los SDKs traen un conjunto de aplicaciones simples de línea de comando, ya sea en código fuente o programas ejecutables fáciles de usar. Estas herramientas fueron construidas usando las funciones de la API LDAP y pueden servir como ejemplo de aplicaciones. Permiten realizar operaciones básicas, tales como búsquedas, modificación, agregado y borrado de entradas en el directorio que se encuentra en el servidor LDAP. Cada operación básica es realizada por un programa simple:

- ldapsearch: es la interfaz de línea de comando para la función ldap_search() de la API LDAP y permite buscar en el directorio de un servidor LDAP.

---

[16] *RFC* 1823: "The LDAP Application Program Interface"
[17] *JNDI*: **J**ava **N**aming and **D**irectory **I**nterface

- ldapadd: es la interfaz de línea de comando para la función ldap_add() de la API LDAP y permite agregar una entrada en el directorio de un servidor LDAP. La funcionalidad de llamada a la API ldap_add() esta incluída en ldap_modify(), difiere de ldap_add() sólo en un aspecto: la opción -a (para agregar) es por defecto seteada cuando se invoca ldapadd.
- ldapmodify: es la interfaz de línea de comando para la función ldap_modify() de la API y permite modificar entradas existentes en el directorio de un servidor LDAP.
- ldapdelete: es la interfaz de línea de comando para la función ldap_delete() de la API y permite borrar entradas en el directorio de un servidor LDAP con un DN que debe ser especificado.
- ldapmodrdn: es la interfaz de línea de comando para la función ldap_mordn2() de la API y permite modificar el RDN de las entradas a un directorio de un servidor LDAP.

Estos nombres se corresponden a un entorno UNIX. Cada utilidad corresponde a una operación del protocolo LDAP. Combinando estas herramientas y usando, por ejemplo, un lenguaje de scripting como Perl, se puede fácilmente construir aplicaciones más complejas. Además, éstas estarán disponibles en programas CGI basados en web. Las herramientas de líneas de comando mencionadas soportan SSL, el cual asume por supuesto, que el servidor tiene disponible su puerto SSL y que SSL está apropiadamente configurado.

URL[18] para LDAP

Como es ya conocido, las URLs proveen una forma estándar de referirse a recursos en la Internet o en una intranet. El ejemplo más común es una página web tal como: *http://www.info.unlp.edu.ar/alumnos.html*. En este caso, http hace referencia al protocolo de transferencia de hipertexto (HTTP) usado por los navegadores, *www.info.edu.ar* es el host a contactar, y *alumnos.html* es el nombre de un archivo en el host. Usando esta URL, un navegador puede retornar y mostrar la página. Las URLs pueden estar definidas para otros protocolos.

Como LDAP se ha convertido en un protocolo importante en la Internet, ha sido definido un formato de URL para recursos LDAP en la RFC 2255. Las URLs para LDAP comienzan con ldap:// o con ldaps:// si en servidor LDAP se comunica usando SSL.. las URLs de LDAP pueden simplemente nombrar un servidor LDAP, o pueden especificar una búsqueda compleja en el directorio.

La sintaxis para una URL LDAP es.

ldap[s]://[<host> [:<port>]] [/ [<dn> [? [<attributes>] [? [<scope>] [? [<filter>] [? <extensions>]]]]]]

Las URLs de LDAP pueden especificar cualquier cosa desde un servidor LDAP a un simple atributo de una simple entrada al directorio y ofrecer la mayoría de la funcionalidad provista por las funciones de la API. Un uso común de las URLs de LDAP son las referencias. También pueden ser usadas para aplicaciones; por ejemplo, un correo electrónico o la libreta de direcciones de un cliente podría almacenarse en una lista de distribución como una URL de LDAP

La mayoría de los SDKs de LDAP incluyen un conjunto de funciones para el manejo de URLs de LDAP.

---

[18] *URL*: **U**niform **R**esouce **L**ocator: cadena de caracteres con la cual se asigna una dirección única a cada uno de los recursos de información disponibles en Internet.

Integración de Java con LDAP

Java es un lenguaje orientado a objetos especialmente creado para Internet y navegadores. También permite que pequeñas aplicaciones llamadas Applets sean descargadas en un navegador desde la red y ejecutadas de una manera segura, o se puede ejecutar la aplicación en el servidor (estas aplicaciones son llamadas Servlet)

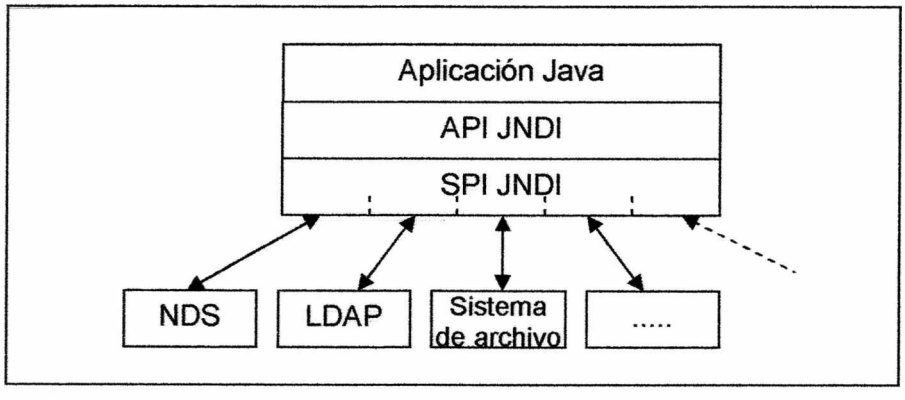
Un desarrollador de la aplicación tiene dos alternativas para acceder a LDAP desde una aplicación Java. La Java LDAP API, llamada JDAP es una clase LDAP [Ref. 6] y JNDI, desarrollada por Sun Microsystems como parte de su conjunto de API para Java Enterprice. Ambas soportan sólo una interfaz de programación sincrónica. JDAP es una continuación de la API de C para LDAP mientras que JNDI[19] provee una interfaz de directorio y naming generalizado.

Un servicio de nombres organiza y nombra objetos. Provee una asociación conocida como enlace entre un nombre y un objeto, este enlace no debe ser confundido con la conexión entre un cliente y un servidor que también se llama enlace.

Un servicio de directorio puede ser considerado como un tipo específico de servicio de nombres, en el cual los objetos enlazados a los nombres son entradas al directorio. Las entradas al directorio están compuestas de atributos que almacenan valores describiendo la entidad representada por la entrada.

JNDI provee una interfaz de servicios de directorio y de nombres generalizada. Por ejemplo, JNDI puede ser usado para retornar archivos desde un sistema de archivos. En este caso un sistema de archivos actuando como un servicio de nombres puede retornar el archivo que está enlazado a un nombre de archivo particular. JNDI puede también ser usado para acceder a un directorio LDAP realizando búsquedas y retornando atributos.

JNDI provee una API que usan las aplicaciones para acceder a un servicio de nombre y de directorio. JNDI provee también una SPI[20] que permite el acceso a un servicio de directorio subyacente. La SPI, es escrita por el vendedor del servicio de directorio y nombres subyacente, permite conectarse al framework JNDI.



API JNDI e interfaces SPI

JNDI provee todas las operaciones para LDAP versión 3.

[19] *JNDI*: **J**ava **N**aming and **D**irectory **I**nterface, permite utilización de directorios distribuidos.
[20] *SPI*: **S**ervice **P**rovider **I**nterface

Ambiente de Computación Distribuida (DCE) y LDAP

DCE es un estándar creado por la OSF[21] [Ref. 7]. DCE provee un conjunto de herramientas y servicios que soportan aplicaciones distribuidas, escalables y confiables en un entorno heterogéneo.

DCE provee los siguientes servicios:

- Soporta threads para el uso múltiples threads de control con un proceso simple, sobre sistemas operativos que no los soportan.
- DCE RPC[22] soporta encriptación de datos y es independiente de la plataforma ya que provee conversión de los tipos de datos.
- La seguridad provee autenticación, autorización, encriptación y auditoría para proteger el acceso a datos y recursos.
- El servicio de directorio provee un repositorio central para información acerca de recursos en el sistema distribuído.
- El servicio de tiempo distribuído (DTS[23]) mantiene sincronizados los relojes del sistema de computadoras distribuidas.
- El servicio de archivo distribuído (DFS[24]) provee acceso independiente de la ubicación a los archivos a través de la red.

LDAP está siendo integrada en DCE para tomar ventaja de la infraestructura de directorios comunes de Internet. Esto permitirá incrementar la información compartida entre entornos DCE y no DCE. Los usuarios tendrán más alternativas en la elección de qué servicio de directorio y herramientas de administración usarán para almacenar y administrar sus datos en directorio DCE.

El servicio de directorios DCE consiste del servicio de directorio de celdas CDS[25], el servicio de directorio global (GDS[26]) y el agente de directorio global GDA[27]. CDS es un servicio de directorio replicado y distribuido que almacena información acerca de los recursos en una celda DCE. Una celda DCE es un conjunto de máquinas y recursos que son administrados como una unidad. Las celdas DCE pueden ser tan pequeñas como unas pocas máquinas y usuarios o tan grandes como miles de máquinas y millones de usuarios.

Si una aplicación nunca accede a recursos fuera de su celda DCE, sólo se requiere CDS. Si una aplicación necesita comunicarse con recursos en otras celdas DCE entonces se requiere GDA. El GDA accede a un directorio global (esto es, no CDS) donde los nombres de celdas DCE pueden estar registrados. Este directorio global puede ser un directorio DNS o un directorio X.500. El uso de GDA permite a una organización enlazar múltiples celdas DCE juntas usando un directorio privado en una intranet o un directorio público en Internet.

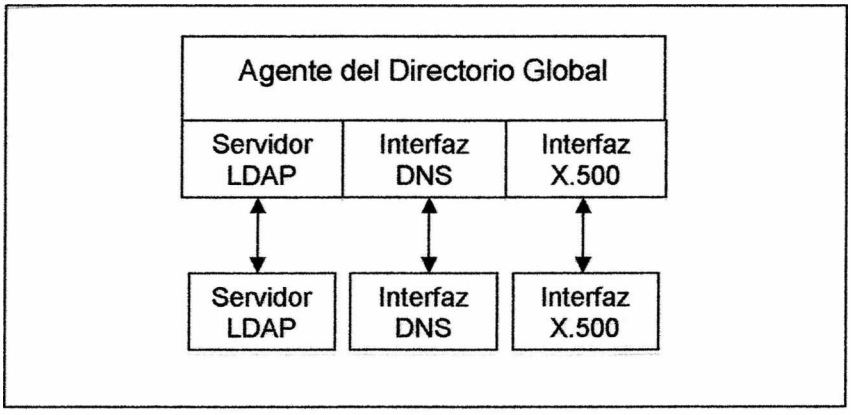
Interfaz LDAP para el GDA

El hecho de que LDAP esté siendo integrado en un DCE permite a las celdas DCE ser registradas en directorios LDAP. El nombre de una celda DCE remota e información acerca de servidores CDS en esa celda es registrada en un servidor de

---

[21] *OSF*: **O**pen **S**oftware **F**undation, en 1996 se unió con X/Open y formaron The Open Group
[22] *RPC*: **R**emote **P**rocedure **C**all: protocolo que permite a un programa de ordenador ejecutar código en otra máquina remota sin tener que preocuparse por las comunicaciones entre ambos.
[23] *DTS*: Distributed Time Service
[24] *DFS*: Distributed File Service
[25] *CDS*: Cell Directory Service
[26] *GDS*: Golbal Directory Service
[27] *GDA*: Global Directory Agent

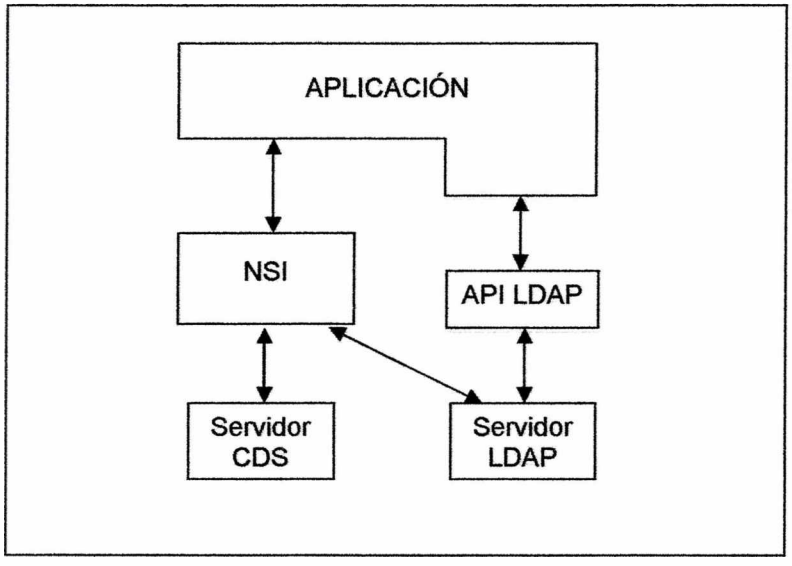
directorio LDAP. El GDA en una celda que quiere conectarse a la celda remota es configurado para acceder al directorio LDAP.



Interfaz LDAP para el GDA

Interfaz LDAP para el CDS

DCE provee dos interfaces de programación para el servicio de directorio: Interfaz de Servicio de Nombre (NSI[28]) y el Servicio de Directorio X/Open (XDS[29]). Las aplicaciones DCE usan comúnmente NSI.



Interfaz LDAP para NSI

Servidor nativo LDAP

LDAP asume un rol de infraestructura de directorio estándar y las componentes DCE se integran adecuadamente con LDAP, por lo cual, reemplazar CDS con LDAP es posible.

---

[28] *NSI*: **N**ame **S**ervice **I**nterface
[29] *XDS*: **X**/Open **D**irectory **S**ervice

Otro software de middleware

DCE es un software de middleware. Otros softwares para el middleware pueden ser bases de datos, sistemas operativos de red, y corredor de objetos distribuídos. Todos ellos comparten el mismo problema: necesitan tener algún soporte para almacenar información de ubicación y servicio.

Referencias:

[Ref. 1] http://www.openldap.org/doc/admin22/

[Ref. 2] http://www.ietf.org/rfc/rfc1823.txt

[Ref. 3] http://www.ietf.org

[Ref. 4] Detalles de la documentación y especificaciones pueden ser encontradas en *java.sun.com/products/jndi/index.html*

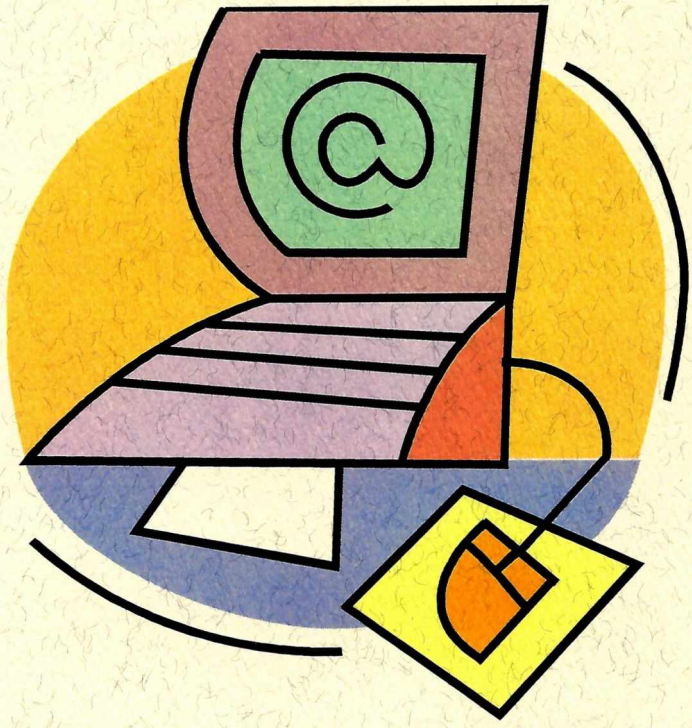
[Ref. 5] ver apéndice A del documento "Understanding LDAP", http://www.redbooks.ibm.com

[Ref. 6] Definido en el IETF draft *The Java LDAP Application Program Interface*

[Ref. 7] www.opengroup.org

Módulo 2

Montado de la Infraestructura PKI



OpenCA

OpenCA como PKI

¿Qué es OpenCA?

OpenCA es un conjunto de programas y scripts que permiten implementar una Infraestructura de Clave Pública (PKI[1]). Es open source e implementa los protocolos más usados con criptografía. Está basada en algunos proyectos open source, entre ellos: OpenLDAP, OpenSSL, Apache, Apache mod_SSL.

OpenCA consta de tres componentes: la Autoridad de Certificación, la Autoridad de Registración y el Repositorio.

La *Autoridad de Certificación* es el servidor de la CA y se encarga de emitir y revocar certificados.

La *Autoridad de Registración* es el servidor de la RA y su tarea principal es atestiguar la asociación entre la clave pública y la entidad propietaria del certificado.

Por razones de seguridad, es recomendable que la CA no sea accedida a través de Internet. La CA puede comunicarse con la RA de manera manual, por ejemplo, usando medios removibles.

Es recomendable que el acceso a la RA esté restringido, de manera tal, que sólo el operador de la RA pueda accederla. Un *operador* es la persona encargada de manejar la RA o la CA a través de su interfaz pública, usando un certificado digital como medio de autenticación. El operador es el intermediario por el cual OpenCA interacciona con los usuarios e internet.

Servicios que brinda

La OpenCA permite configurar y ejecutar una PKI. El propósito de una PKI es proveer claves y manejo de certificados confiables y eficientes. Para conseguir esto la OpenCA no solamente ofrece certificados, sino también los maneja, es decir, determina cuanto tiempo van a ser válidos; mantiene listas de certificados que ya no son válidos (Listas de Revocación de Certificados o CRLs), etc. Con este fin provee los siguientes servicios:

- Verificación de solicitud de Certificados.
- Procesamiento de solicitud de Certificados.
- Firma, asignación y manejo de Certificados.

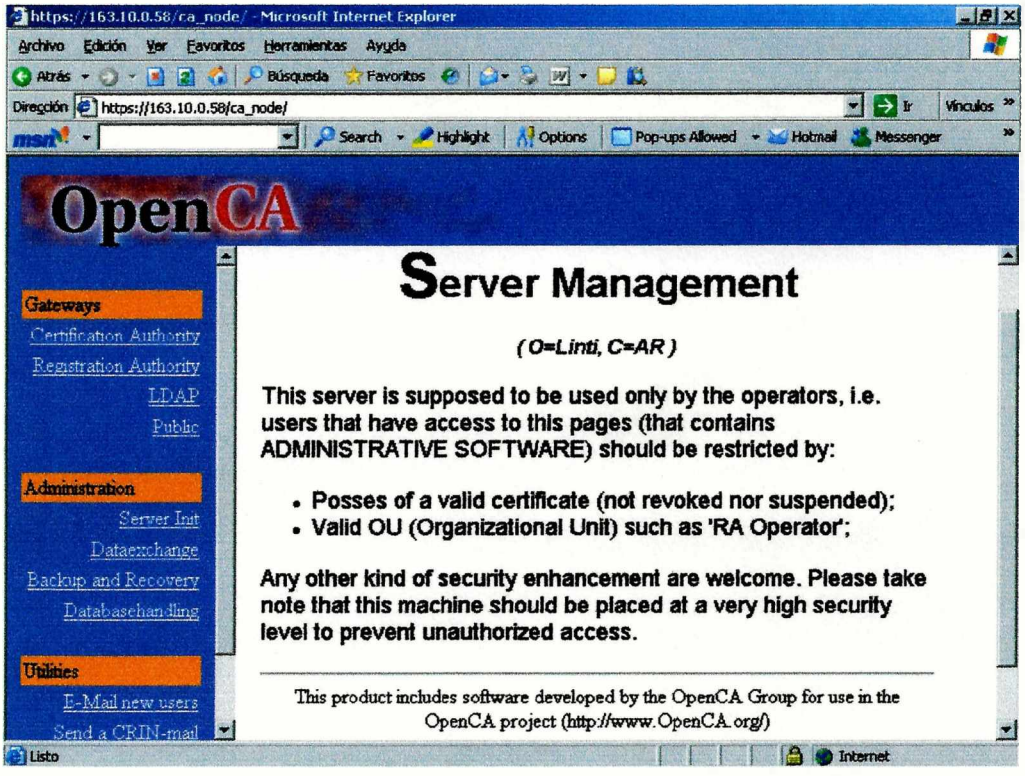
La interfaz de OpenCA tiene gran variedad de opciones que la hacen muy funcional, aunque algunas de esas opciones no resultan muy claras.

Funcionalidad del Servidor de la CA

A continuación se muestra la interfaz del módulo correspondiente a la CA, en la cual se visualizan las distintas opciones para su administración y uso. Las mismas se pueden dividir en las siguientes categorías:

- Inicialización/administración
- Requerimientos
- Certificados
- Lista de revocación de certificados

[1] *PKI*: **P**ublic **K**ey **I**nfraestructure



Las siguientes secciones enumeran las opciones presentadas al operador cuando usa la interfaz web.

Inicialización / Administración de la CA

Para inicializar y administrar la CA se deben utilizar las siguientes opciones:

- *Generar una nueva clave privada de la CA:* es muy importante generar una nueva clave secreta (pass-phrase) para una CA y sobreescribir la vieja después que los certificados fueron emitidos.

El usuario es consultado antes de sobreescribir la clave privada. Esta clave privada es en realidad el pass-phrase que será usado para proteger la clave privada de la CA. Generalmente, los algoritmos de clave pública en el procedimiento de creación de una clave, generan un conjunto de números muy largo: una parte de ellos constituye la clave privada; luego, para mayor seguridad, encripta todo. Cuando es necesaria la clave privada, la aplicación pregunta por el pass-phrase, desencripta la clave privada encriptada y la usa.

Nota: la primera clave privada se genera en la fase de instalación del servidor.

- *Generar un nuevo requerimiento de certificado de la CA:* el Requerimiento de Firma del Certificado (CSR[2]), es generado para ser luego firmado. Esta firma puede hacer hacerse de una de las siguientes formas:
  - Autofirmado con la clave pública de la CA, generada con la opción anterior, en caso de tratarse de una CA raíz.
  - Firmado por otra CA.

[2] *CSR* : **Certificate Signing Request**

- *Exportar un requerimiento de certificado de la CA:* esta opción exporta el CSR que fue generado con la opción anterior. Se crea un archivo para el CSR en el sistema de archivos. Esta opción no es necesaria si el certificado fue autofirmado.

- *Generar un certificado de la CA autofirmado:* esta opción usa el CSR generado para crear el certificado de la CA. Es firmado con la clave pública de la CA.

- *Exportar el certificado de la CA:* exporta el certificado de la CA. Deben ponerse a disposición del público copias de este certificado.

Requerimientos

- *Importar requerimientos:* importa requerimientos (CSRs) aprobados por el operador de la RA, para que sean firmados por la CA. El administrador del servidor de la RA utilizó el comando "Exportar requerimientos" para exportar el CSR a un medio removible. Con este comando, el administrador del servidor de la CA lo retornará firmado.

- *Requerimientos pendientes*: muestra los requerimientos (CSRs) pendientes que residen en la CA y esperan ser firmados.
- Debemos observar que una terminología similar, requerimientos pendientes, es usada en la RA con diferente significado: en la RA un requerimiento pendiente es un CSR que espera para ser aprobado por el administrador de la RA y ser enviado a la CA.

- *Borrar requerimientos*: muestra los requerimientos borrados de la CA. De acuerdo a cómo están relacionados el servidor de la CA y el servidor de la RA, ésta última firma cada CSR con su propia clave privada y el servidor de la CA chequea la firma: si ésta es verificada, crea el certificado, sino lo borra y lo muestra aquí.

- *Remover los requerimientos borrados*: remueve físicamente los requerimientos borrados del sistema de archivos del servidor de la CA.

Certificados

- *Certificados emitidos:* muestra todos los certificados que alguna vez fueron emitidos por la CA.

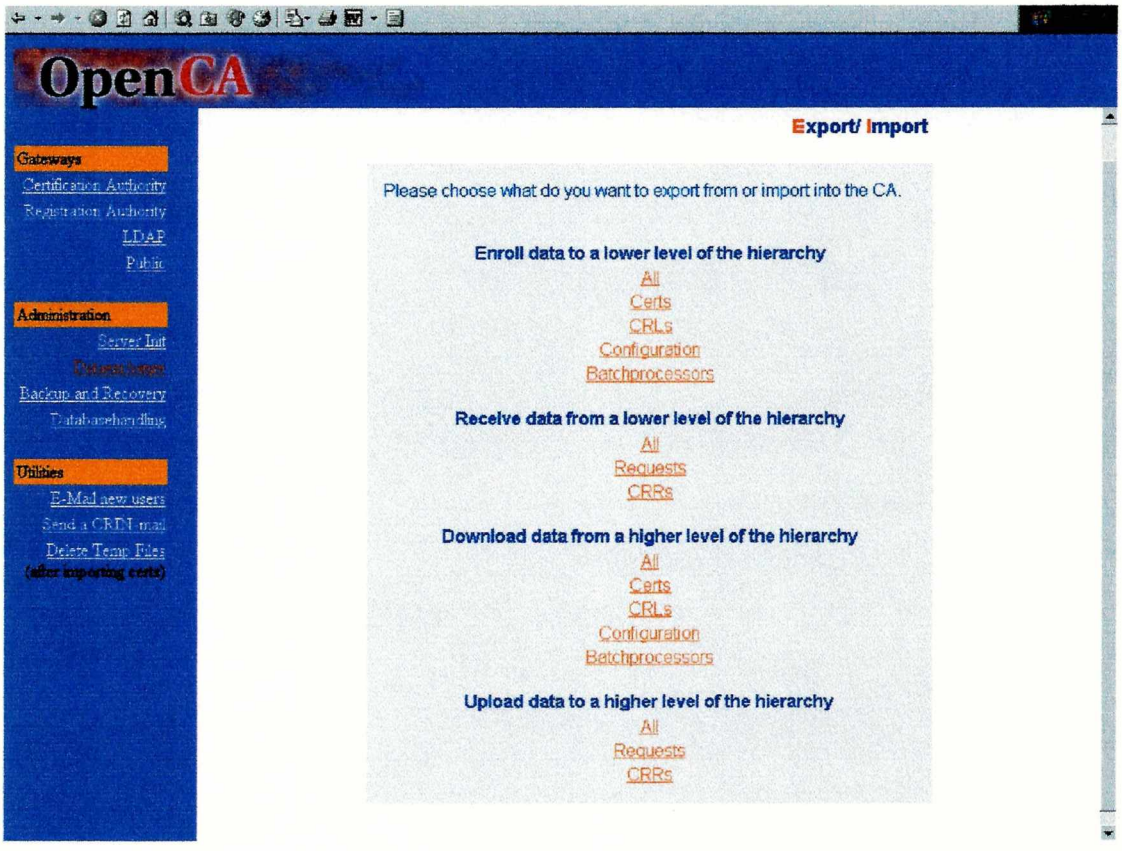
- *Exportar certificados*: exporta los certificados a un medio removible para que sean enviados al servidor de la RA. Es responsabilidad la RA distribuir los certificados a cada propietario.

Lista de revocación de certificados (CRL[3])

- *Exportar la CRL:* exporta la CRL al servidor de la RA. Este tiene la responsabilidad de hacer que la CRL sea conocida y esté disponible para cada usuario.

A continuación se muestra la interfaz disponible para importar y/o exportar certificados y CRLs.

[3] *CRL:* **C**ertificate **R**evocation **L**ist



Funcionalidad del Servidor de la RA

Cuando un operador local de la RA recibe un requerimiento de usuario se comunica con el servidor de la RA para tener acceso a la CA. Ningún usuario se comunica directamente con el servidor de la RA. Este servidor debe tener un nivel de seguridad muy alto para prevenir accesos no autorizados y es administrado por el Administrador de la RA. Las distintas opciones para su administración y uso se pueden dividir en las siguientes categorías:

- Requerimientos
- Certificados
- Lista de revocación de certificados

Requerimientos

- *Exportar requerimientos:* exporta requerimientos aprobados al servidor de la CA.

- *Requerimientos pendientes*: muestra los CSR que esperan ser aprobados por el operador de la RA. La aprobación puede basarse en documentos identificatorios u otras credenciales.

- *Requerimientos aprobados*: muestra los CSR que ya han sido aprobados por el administrador del servidor de la RA. Este CSR será enviado al servidor de la CA usando la función *Exportar requerimientos.*

- *Remover los requerimientos exportados*: una vez que los requerimientos están aprobados y son exportados a la CA, pueden ser removidos con esta opción.

Certificados

- *Importar certificado de la CA*: importa el certificado de la CA y lo graba en el sistema de archivo local. Esta copia del certificado será publicada.

- *Importar nuevos certificados*: importa los certificados firmados recientemente desde la CA. Estos son copiados en el sistema de archivo local.

- *Exportar certificados a LDAP*: exporta los certificados al servidor LDAP especificado. *Los usuarios recobrarán sus certificados accediendo al servidor LDAP, en lugar de contactar directamente al sitio público de la CA.*

Lista de revocación de certificados (CRL)

- *Importar la CRL:* importa la CRL desde la CA para que sea publicada.
- *Exportar el Requerimiento de Certificado de Revocación (CRR*[4]*)*: exporta el Requerimiento de Revocación ya aprobado al servidor de la CA. Este revocará dicho certificado.

Utilidades

- *Enviar un correo electrónico a usuarios para nuevos certificados emitidos*: informa a los usuarios que el certificado ha sido preparado y que ellos deben seguir un procedimiento, mencionado en el mail, para recuperarlo.

- *Borrar archivos temporales (luego de importar certificados):* es un comando de limpieza. Cuando los usuarios son notificados, se crean archivos temporales para indicar que el correo electrónico será enviado. Si estos archivos no son borrados, la próxima vez que se envíen correos electrónicos de notificación, los usuarios que ya tengan dicha notificación la recibirán nuevamente.

Funcionalidad de la interfaz PKI-usuario final

Los servidores públicos o los servidores a los que los usuarios realmente tienen acceso, están configurados para ser muy seguros y son usados para consultar acerca de certificados, requerir un certificado, etc.. Este es el único punto de entrada a la infraestructura de la CA desde Internet.

[4] *CRR*: **C**ertificate **R**evocation **R**equests

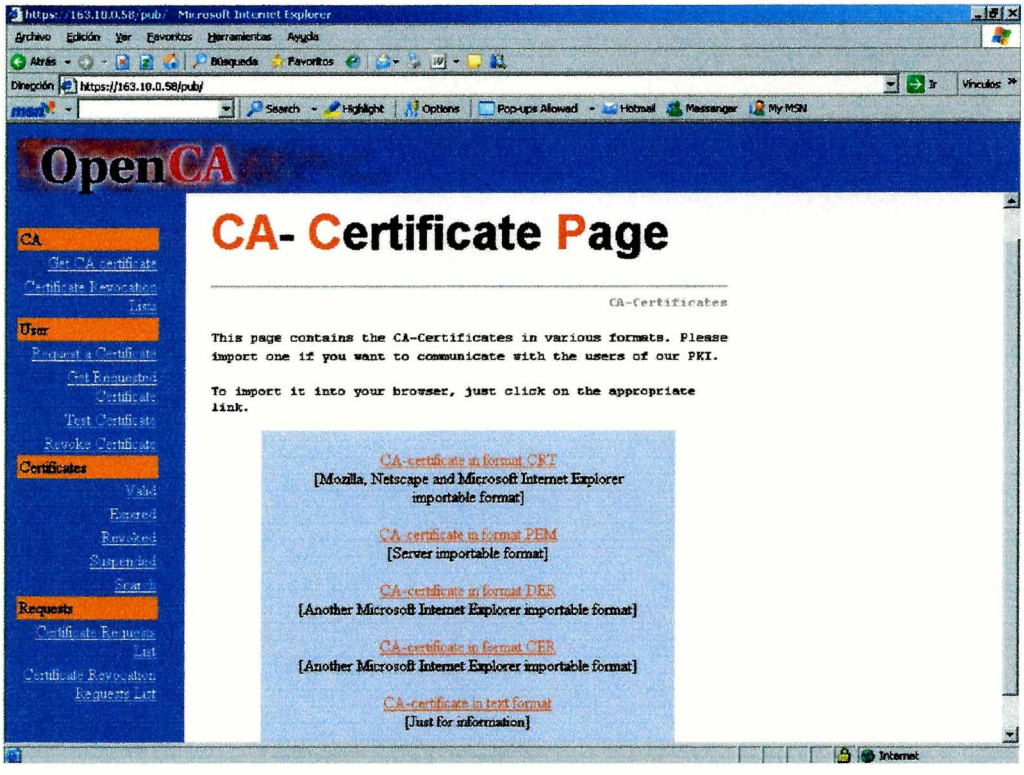
A continuación se muestra la interfaz pública de la CA, en la cual se enumeran las opciones que brinda para su utilización. Las mismas se pueden dividir en las siguientes categorías:

- Obtener el certificado de la CA
- Lista de revocación de certificados
- Requerir un certificado
- Obtener el certificado requerido
- Lista de certificados emitidos

Obtener el certificado de la CA

Permite al usuario importar el certificado de la CA en su navegador. Este es un procedimiento básico y muy importante y normalmente se realiza solo una vez en el tiempo de vida del certificado de la CA. Este certificado también es llamado *Certificado Raíz*. Es el punto de inicio para permitir al cliente comunicarse de manera segura con la CA.

A continuación se muestra la interfaz que permite dicha operación:

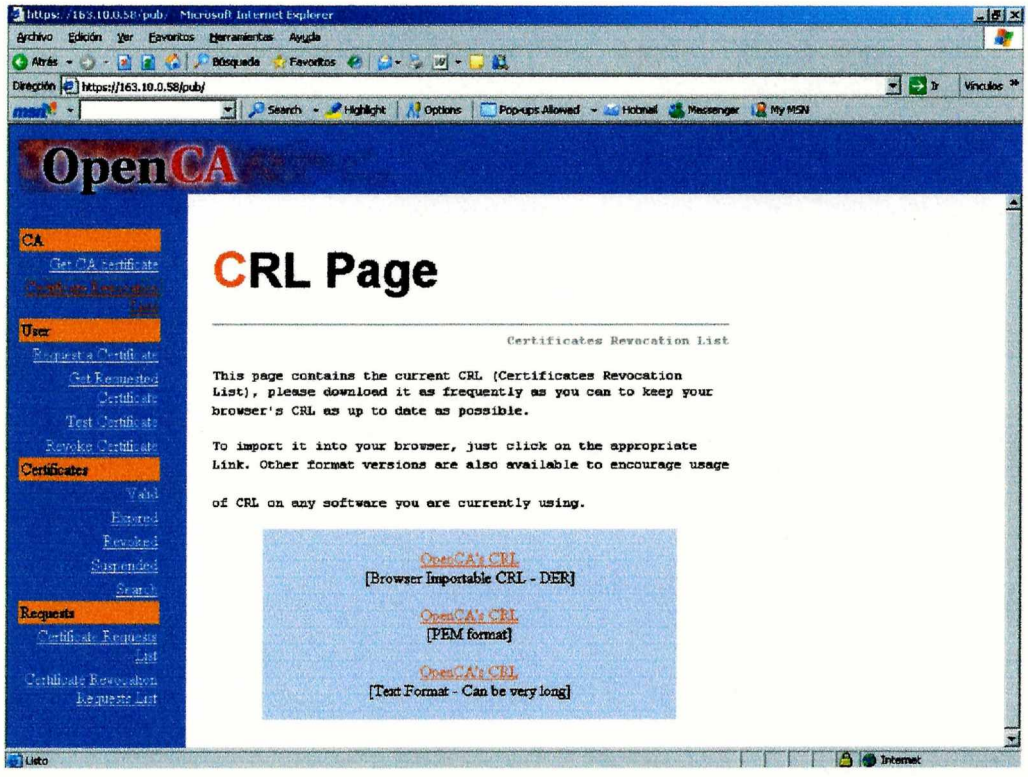


Lista de revocación de certificados (CRL)

Conduce a la página donde está publicada la CRL. Aquí, la CRL, producida por la CA, será importada en el navegador u otra aplicación.

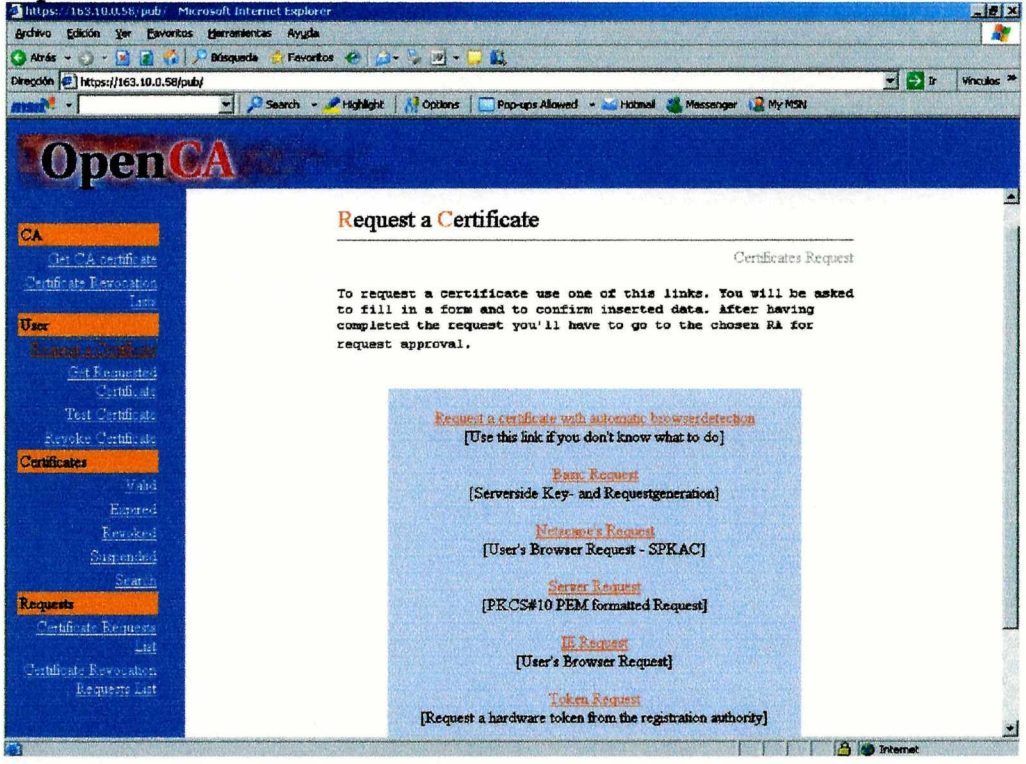
- CRL de la OpenCA (formato DER): con esta opción, se genera una CRL importable por el navegador, para ser automáticamente incluida en la lista CRL del navegador.
- CRL de la OpenCA (formato PEM): con esta opción, la CRL es generada en formato PEM.

- CRL de la OpenCA (formato TXT): con esta opción, la CRL es generada en formato TXT. El archivo generado por este comando puede ser muy grande.



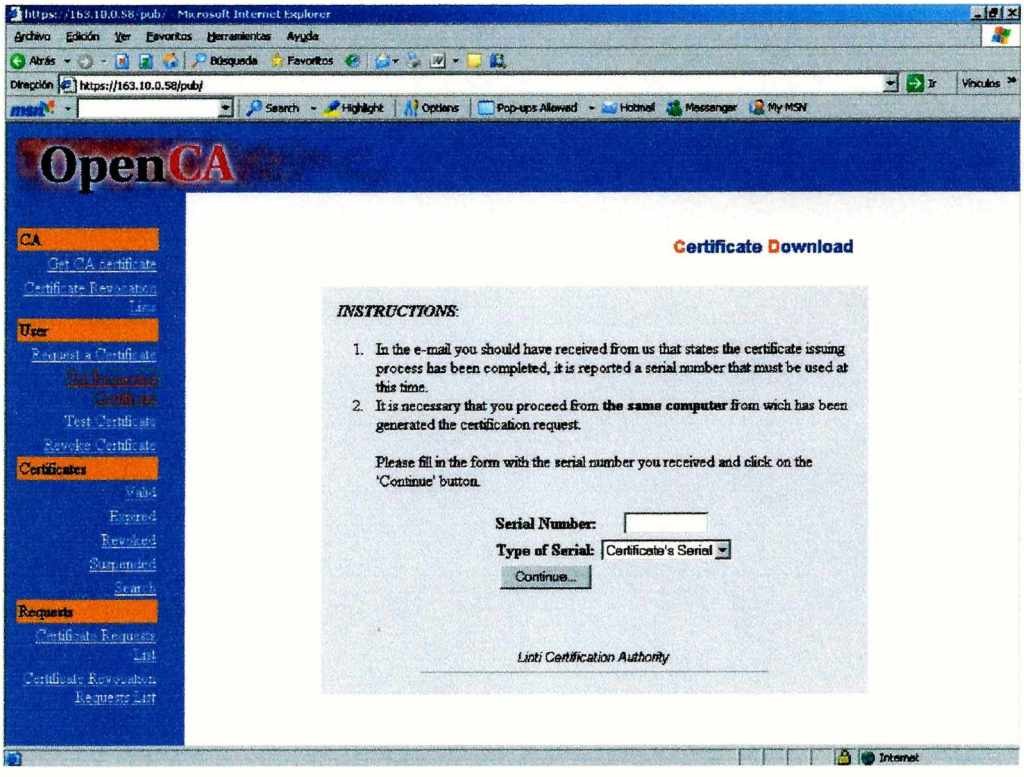
Requerir un certificado

Inicia el procedimiento para requerir un certificado. Esto se realiza a través de la siguiente pantalla:



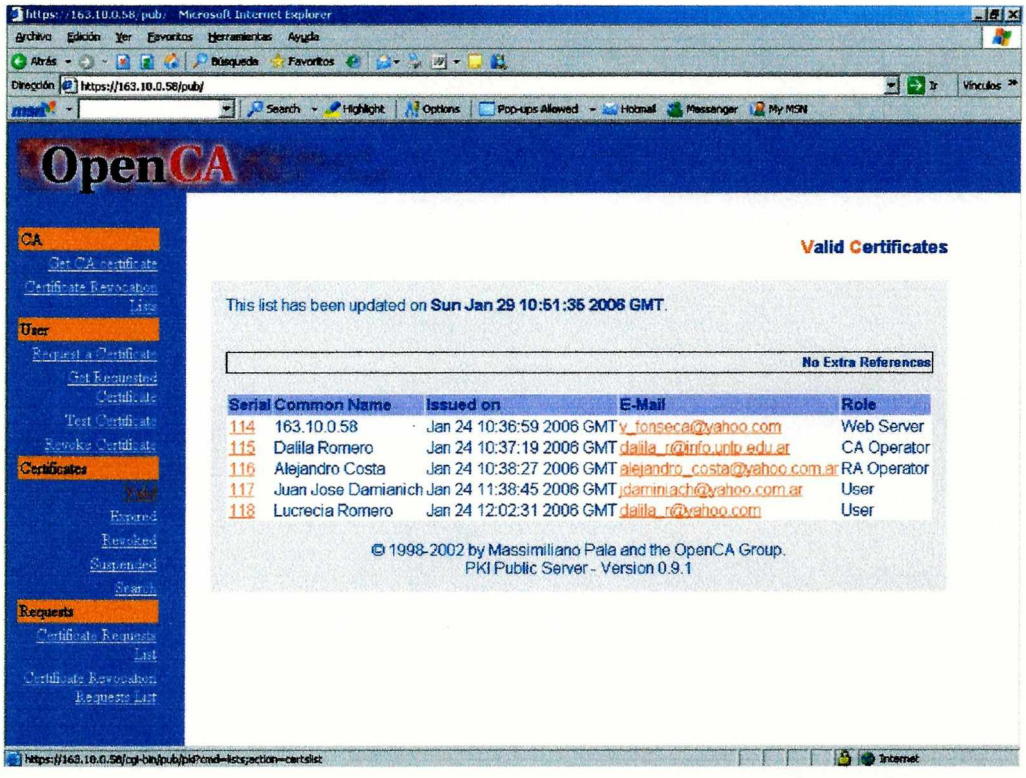
Obtener el certificado requerido

Permite al usuario recobrar su certificado emitido e importarlo en la aplicación. El usuario ha recibido el correo electrónico de notificación desde la RA con instrucciones para recuperar el certificado. En el correo electrónico, hay un número de serie del certificado que tiene que ser presentado al operador de la RA para recobrarlo. Este número de serie sólo sirve como identificación, no se usa para autenticación.



Lista de certificados emitidos

Consiste en una interfaz web que es una vista del repositorio. Muestra una lista de los certificados emitidos por esta CA. Desde allí se puede obtener el certificado de un tercero.



Al seleccionar un certificado se muestran los datos del mismo. Por ejemplo, si se selecciona el certificado con cn= Dalila R, se visualizará:


OpenCA
CA
Get CA certificate
Certificate Revocation Lists
User
Request a Certificate
Get Requested Certificate
Test Certificate
Revoke Certificate
Certificates
Valid
Expired
Revoked
Suspended
Search
Requests
Certificate Requests List
Certificate Revocation Requests List
Certificate Details
CERTIFICATE DATA
This is the relevant data about the selected certificate.
Variable
Value
Certificate Version:
3
Serial Number:
1
Common Name:
Dalila R
E-Mail:
dalila_r@yahoo.com
Distinguished Name:
serialNumber=1
CN=Dalila R
OU=Internet
O=UNLP
C=AR
Role:
CA Operator
Fingerprint:
6A:DF:0C:A8:ED:29:1F:29:7B:54:0C:6D:E0:59:0B:6A
Issued by:
CN=root
O=UNLP
C=AR
Valid From:
Jul 29 14:16:35 2003 GMT
Expiration on:
Jul 28 14:16:35 2004 GMT
Current Status:
Valid
Netscape CA Revocation Url:
https://ca/pub/crl/cacrl.crl
Netscape Cert Type:
SSL Client, S/MIME, Object Signing
Netscape Comment:
Certification Authority Administrator of UNLP
Netscape Revocation Url:
https://ca/pub/crl/cacrl.crl
X509v3 Authority Key Identifier:
keyid:1F:65:97:A9:B8:28:A3:2A:2D:73:EA:79:45:25:B6:66:39:41:B2:7B
DirName:/C=AR/O=UNLP/CN=root serial:00
X509v3 Basic Constraints:
CA:FALSE
X509v3 CRL Distribution Points:
URI:https://ca/pub/crl/cacrl.crl
X509v3 Extended Key Usage:
TLS Web Client Authentication, E-mail Protection, Microsoft Smartcardlogin
X509v3 Issuer Alternative Name:
X509v3 Key Usage:
Digital Signature, Non Repudiation, Key Encipherment
X509v3 Subject Alternative Name:
email:dalila_r@yahoo.com
X509v3 Subject Key Identifier:
74:D8:C6:D9:C4:F0:EB:42:3A:FE:43:59:B6:6A:6F:A2:66:C7:31:28
Download the certificate
Revoke the certificate
© 1998-2002 by Massimiliano Pala and the OpenCA Group
PKI Public Server - Version 0.9.1

Instalación y configuración

A continuación se detallarán las consideraciones e instrucciones a tener en cuenta para la instalación de OpenCA.

Pre-requisitos

Antes de instalar OpenCA se debe tener instaladas las siguientes herramientas:

- OpenSSL (SNAP-20011026+)
- Perl (5+ con soporte DBM)
- Apache Web Server
- mod_ssl (para Apache)

Opcionalmente, se pueden también instalar otras herramientas:

- Soporte para LDAP
  - OpenLDAP
  - perl-ldap (módulo perl)

La instalación de OpenCA involucra a sus tres componentes: el servidor de la CA, el servidor de la RA. El proceso de instalación consiste en setear los archivos de configuración, copiar las páginas HTML en los directorios apropiados y finalmente agregar los scripts CGI[5] en los directorios correspondientes.

Para obtener los archivos necesarios para la instalación de estas componentes se debe ingresar a la página de OpenCA, www.openca.org y descargarlos desde ahí.

Los pasos generales para la instalación de OpenCA se describen a continuación.

Instalación del servidor de la CA

Es la instalación de la Autoridad de Certificación. En primer lugar, se debe descomprimir OpenCA con el siguiente comando:

root# **tar xvfz OpenCA-0.2.0.tar.gz**

para instalar el software, se debe entrar al directorio creado (OpenCA-0.2.0) y ejecutar

root# **make install-ca**

Cuando se instala el componente OpenCA para el Servidor de la CA se deben usar los siguientes parámetros:

[5] *CGI*: Common Gateway Interface (en inglés «Pasarela de Interfaz Común», abreviado CGI) es una importante tecnología de la World Wide Web que permite a un cliente (explorador web) solicitar datos de un programa ejecutado en un servidor web. CGI especifica un estándar para transferir datos entre el cliente y el programa.

Parámetros para la instalación del Servidor de la CA

Parámetro	Valor
Directorio de instalación de la OpenSSL	/usr/local/ssl
Directorio base para el directorio del Servidor de la CA Base	/usr/local/RAServer
Usuario de Servidor web	nobody.nobody
Usar el comando found de OpenSSL	**Y**
Continuar con la instalación	**yes**

Luego, para instalar las páginas web que acompañan al servidor de la CA se debe ejecutar:

root# **make install-CA-web**

con los siguientes parámetros:

Servidor de la CA - Parámetros para la instalación del servidor web

Parámetro	Valor
Directorio para páginas HTML	/usr/local/apache/htdocs/ca
Directorio para CGI	/usr/local/apache/cgi-bin
Continuar con la instalación	**yes**

Finalmente, se deben seguir las instrucciones de las páginas web para inicializar el servidor de la CA, crear la clave privada de la CA y el certificado.

Instalación del servidor de la RA

Es la instalación de la Autoridad de Registración. Es recomendable que el servidor de la RA sea instalado en una máquina distinta que el del servidor de la CA. De cualquier forma, pueden ser instalados en la misma máquina.

Los pasos para la instalación del servidor de la RA son los mismos que se enunciaron para crear el servidor de la CA.

En primer lugar, se debe descomprimir la OpenCA con el siguiente comando:

root# **tar xvfz OpenCA-0.2.0.tar.gz**

para instalar el software del servidor de la RA, se debe entrar al directorio creado (OpenCA-0.2.0) y escribir

root# **make install-raserver**
root# **make install-raserver-web**

cuando se instala el servidor de la RA se deben usar los siguientes parámetros:

Parámetro	Valor
Directorio de instalación de la OpenSSL	/usr/local/ssl
Directorio base para el Servidor de la CA	/usr/local/RAServer
Usuario de Servidor web	nobody.nobody
Usar el comando found de OpenSSL	**Y**
Continuar con la instalación	**yes**

Parámetros para la instalación del Servidor de la RA

Servidor de la RA - Parámetros para la instalación del servidor web

Parámetro	Valor
Directorio para páginas HTML	/usr/local/apache/htdocs/ra
Directorio para CGI	/usr/local/apache/cgi-bin
Continuar con la instalación	**yes**

Instalación de la interfaz pública

En esta fase se instalará el operador de la Autoridad de Registración, por la cual tendrán acceso los usuarios finales.

Para instalar el software de la interfaz pública, se debe entrar al directorio creado (OpenCA-0.9.0) y ejecutar

root# **make install-secure**

Nuevamente, es recomendable que la interfaz pública sea instalada en un sistema distinto que el del servidor de la CA y el del servidor de la RA. Pueden ser instalados en el mismo sistema aunque ésto puede acarrear dificultad para usarlo y probablemente, error en el testeo.

Los pasos para la instalación del operador de la RA son los mismos que se enunciaron para crear el servidor de la CA y el servidor de la RA.

Operador de la RA - Parámetros para la instalación del servidor web

Parámetro	Valor
Directorio para páginas HTML	/usr/local/apache/htdocs/rao
Directorio para CGI	/usr/local/apache/cgi-bin
Continuar con la instalación	**yes**

Experiencia en la implementación

OpenCA posee tres componentes la CA, la RA y la interfaz pública. En un comienzo se utilizó la versión 0.9.0 de OpenCA. Al tener varios problemas en su configuración y funcionamiento, se decidió instalar una versión más nueva: la 0.9.1, para lo cual también se debió cambiar la versión de OpenSSL por la 0.9.7.

En el momento de la instalación de OpenCA es muy importante configurar bien las opciones, sobre todo si se integrará esta herramienta con OpenLDAP. Dichas opciones se setean con el comando de instalación de la siguiente manera:

```
./configure --with-httpd-user=www --with-httpd-group=wwwadmin --with-
country=AR --with-loc="La Plata" --with-org=UNLP --with-web-host=ca --with-
httpd-host=ca --with-ca-organization=UNLP --with-ca-country=AR --with-ldap-
host=ca --with-ldap-port=389 --with-ldap-root="cn=LDAP Manager, o=LINTI,c=AR" --with-
ldap-root-pwd=secret --with-hierarchy-level=ra --with-openssl-
prefix=/usr/local/openssl0.9.7b
```

Al cambiar la versión de la OpenCA, fue necesario modificar ciertas opciones de los archivos de configuración, estos archivos no son iguales en ambas versiones, por ejemplo: el archivo *online.conf* de la versión 0.9.0 es reemplazado por *ra-node.conf* en la versión 0.9.1.

Una vez realizadas las modificaciones necesarias, se hicieron pruebas en la interfaz web de la CA. Surgieron entonces diferentes errores; a continuación se detallarán los más relevantes:

Error 690

Configuration Error. Error while loading configuration (/usr/local/OpenCA/etc/database/DB.conf)

En principio, fueron revisados los archivos de configuración concluyendo que estaba todo bien direccionado. Luego, dado que el archivo *DB.conf* estaba en: */usr/local/OpenCA/etc/servers/*, y según la explicación del error el sistema lo buscaba en */usr/local/OpenCA/etc/database/*, se creó, a la altura del directorio *servers*, el directorio *database* y se copió adentro el archivo *DB.conf*, con lo cual el error fue corregido.

Luego, al entrar a inicializar la base de datos se produjo el siguiente error:

Error 690
Configuration Error. Missing Configuration Keyword : genDBSheet.

El cual también fue solucionado [Ref1]. Una vez hecho ésto, se generó la clave secreta de la CA.

Luego, se creó un requerimiento de certificado para la CA usando la clave secreta de la misma, con las siguientes opciones:

Email address: root@ca.linti.unlp.edu.ar
CN root
O Linti
C AR

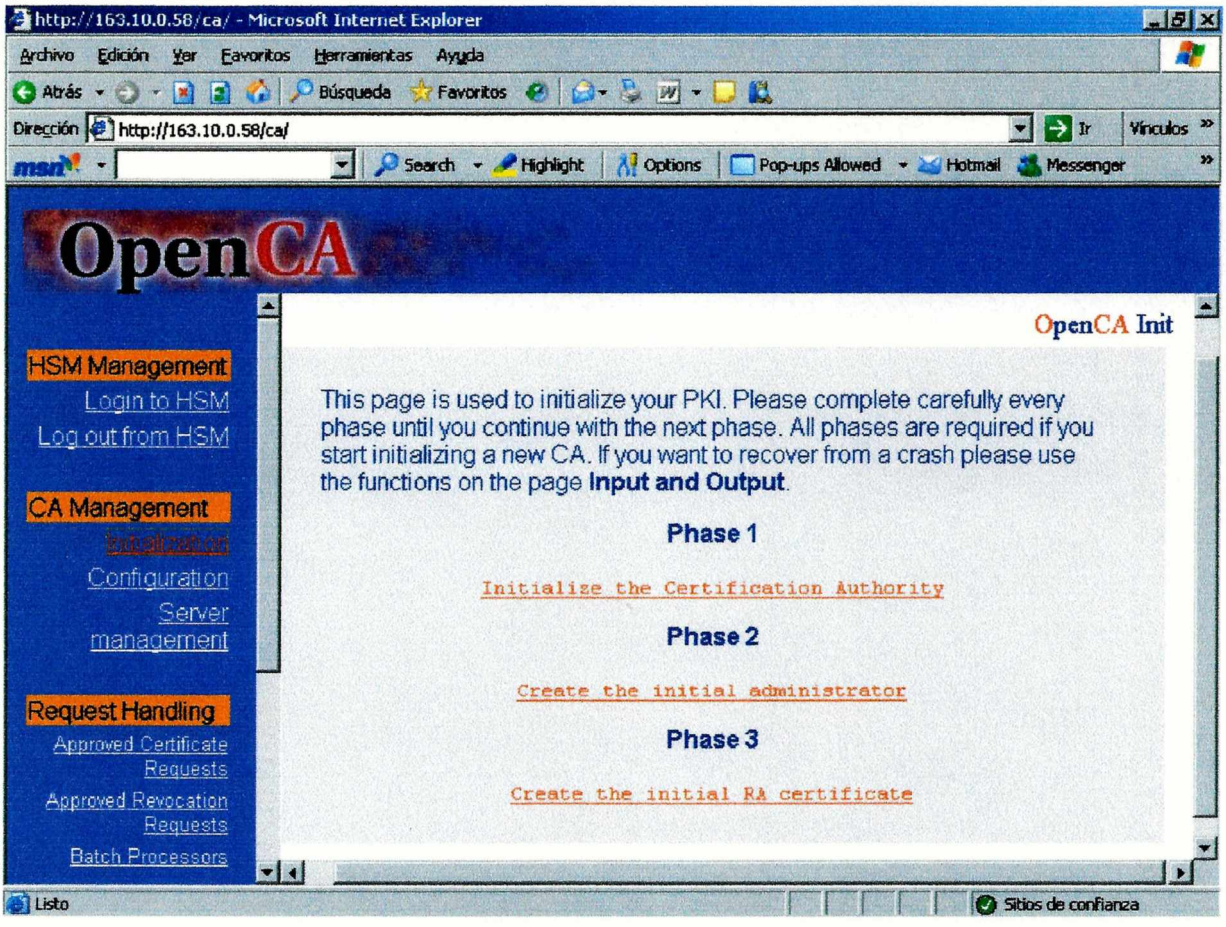
La operación de generación de un requerimiento está compuesta por varios procesos, entre los cuales están la carga de los datos propios del certificado, el

autofirmado del certificado por la CA (Selfsigned CA-Certificate), reconstrucción de la cadena de la CA (Rebuilt Ca Chain) y la exportación de la configuración (Export configuration), todos los procesos se ejecutaron correctamente excepto el último. En el próximo capítulo veremos la solución, ya que este proceso realiza la integración de OpenCA con OpenLDAP.

Inicialización

Una vez instaladas todas las componentes se debe proceder a la inicialización de las mismas. En términos generales se deben realizar los siguientes pasos:

Desde la página web de la autoridad de certificación ingresar a la opción *Server Managent / Certification Authority / Initialization*, se mostrará lo siguiente



En la fase 1 de inicialización, se muestran las siguientes opciones, las cuales deben realizarse en el orden indicado:

DB Setup

- Initialize Database

  Esta opción debe realizarse sólo una vez, luego de la instalación de la CA

Key pair Setup

- Generate new CA secret key

  Acá se genera la clave privada de la CA, la misma es usada para aprobar los requerimientos de certificados de usuario.

Request Setup

- Generate new CA Certificate Request (use generated secret key)

En esta opción se genera el requerimiento para el certificado de la CA. Los valores por defecto son los siguientes:

- Algoritmo de encriptación (des,des3,idea):des3
- Algoritmo asimétrico (rsa, dsa):rsa
- Tamaño de la clave de la CA (in bits):4096

Se completan los parámetros como sean necesarios según la instalación, luego se debe confirmar el DN generado.

Certificate Setup

- Selfsigned CA-Certificate
  - Generate Self Signed CA Certificate (from altready generated request)

En este caso, el requerimiento generado en el paso anterior es firmado por la misma CA, para lo cual hay que indicar la clave privada.

- Signed by another CA
  - Export CA Certificate Request
  - Import CA certificate (approved by Root CA)

En este caso, el requerimiento generado en el paso anterior es firmado por otra CA, para lo cual hay que exportarlo y una vez que la otra CA lo firma hay que importar el certificado firmado.

Final Setup

- Rebuild CA Chain

Se debe obtener una respuesta que confirme el éxito de la operación

- Export Configuration

En la fase 2 de inicialización, se crea el primer usuario de la PKI, el cual debe ser administrador (operador de CA), además debe crearse el certificado para el sitio web (web site). Esto se hace a través de los siguientes pasos:

1. Create a new request
2. Edit the request
3. Issue the certificate
4. Handle the certificate

En la fase 3 de inicialización, en esta etapa se genera el primer server-certificate de la PKI, este certificado debe ser usado por la RA (certificado del operador de la RA). Esto se hace a través de los siguientes opciones:

1. Create a new request
2. Edit the request
3. Issue the certificate
4. Handle the certificate

Descripción del uso de una CA por parte de usuarios finales

Supongamos que Dalila quiere comunicarse con Viviana en forma segura. Se describirán los pasos necesarios para establecer la comunicación usando la CA.

Creación del par de claves y el CSR

Dalila crea un par de claves pública/privada usando un algoritmo público como, por ejemplo, RSA [Ref.2]. Luego, crea un CSR, el cual es el requerimiento de certificado de la CA. En el certificado, hay información acerca de la identidad del usuario, como el nombre, dirección, número de teléfono y dirección de correo electrónico. Los certificados pueden ser usados para autenticar no sólo personas sino entidades en general, tal como un servidor web o un agente de software. En este caso, la información en el certificado podría ser la URL del servidor web, los detalles del administrador web, etc.. Finalmente, la clave pública se agrega al CSR.

La CA firma el CSR

El CSR es recibido por la RA y enviado a la CA para que lo firme. La RA verifica los datos de identidad incluidos en el requerimiento y determina si será aprobado o no. El resultado de la firma del certificado, es enviado de vuelta a Dalila a través de la RA. Para una mejor recolección de los certificados, los mismos son frecuentemente almacenados en un directorio del servidor.

Niveles de certificados

Los certificados pueden tener diferentes grados de seguridad, dependiendo de cuanta certeza tiene la CA de que el propietario del certificado es quien dice ser. En el caso de certificados de evaluación, como están disponibles para compañías como Verisign, la única prueba de identidad necesaria es la validación de la dirección de correo electrónico dada. La compañía envía una dirección de correo electrónico a la dirección dada y una vez que el usuario contesta, ella puede tener el certificado. Esto es aceptable para certificados de usuario que serán utilizados para servicios típicos de Internet. Para compañías que conducen sus negocios por Internet, deberán utilizarse certificados de mayor nivel de seguridad.

Cadenas de CA

Usando el certificado, Dalila puede solicitar que su clave pública sea confiable para cualquiera que pregunte. Viviana, para verificar el certificado de Dalila, necesita encontrar la clave pública de la CA que firmó el certificado de Dalila, y necesita hacerlo de manera segura. Si ambas utilizan la misma CA, ya la tiene. Sino, Viviana pide a su CA que contacte la otra CA para obtener la clave pública. Por cada pregunta de la CA de Viviana a otra CA, la primera necesita tener la clave pública de la otra, entonces la comunicación es segura. Si se puede encontrar una cadena que conduzca a la otra CA entonces se puede establecer una comunicación.

Comunicación híbrida

Teniendo la clave pública de los otros, hay muchos protocolos que pueden garantizar una comunicación segura entre las dos partes. La clave pública criptografiada no es muy apropiada para, por ejemplo, transferir una gran cantidad de datos, y debe ser usada una cifra simétrica para este propósito. La clave para realizar el cifrado simétrico (secreto compartido) puede ser transferida usando la clave pública de la otra parte.

Referencias

[Ref. 1]http://www.mail-archive.com/OpenCA-users@lists.sourceforce.net/msg02577.html, www.OpenCA.org

[Ref.2] Trabajo de grado "Utilizando firma digital", desarrollado por Paula Venosa y Verónica Fredes.

OpenLDAP

OpenLDAP como repositorio

Introducción

El proyecto OpenLDAP fue desarrollado por un equipo de voluntarios. Luego la Universidad de Michigan construyó la fundación LDAP[1].

OpenLDAP [Ref. 1] es principalmente una implementación open source multiplaforma para la suite LDAP. La suite incluye como componentes:

- Slapd: es un servidor LDAP stand-alone (daemon), que escucha y responde a conexiones LDAP. Es configurado usando slapd.conf
- Slurpd: es un servidor de replicación LDAP stand-alone.
- Librerías, las cuales implementan el protocolo LDAP.
- Utilidades, herramientas y ejemplos de clientes.

OpenLDAP posee entre sus características empresariales escalabilidad, replicación y capacidad de referencias; y entre sus características de seguridad incluye soporte para ACLs[2], SSL/TLS/SASL[3] y criptografía. LDAP [Ref. 2] es un conjunto de protocolos para acceso a un servicio de directorios sobre Internet, similar a DNS[4]. El paquete OpenLDAP contiene archivos de configuración, librerías, y documentación para OpenLDAP.

Servicio que brinda

El software OpenLDAP ofrece principalmente un servicio de directorios, para lo cual brinda lo siguiente:

Herramientas para la creación y mantenimiento de la base de datos

Hay dos formas de crear una base de datos. En la primera, se crea la base de datos on-line usando LDAP. Con este método, simplemente se comienza a ejecutar slapd y se agregan entradas usando un cliente de LDAP. Este método es apropiado para bases de datos relativamente chicas. El segundo método para la creación de la base de datos es hacerlo off-line usando utilidades especiales provistas con slapd. Este método es mejor si se trata de una base de datos grande, es decir, con muchas miles de entradas.

Crear una base de datos on-line

Con este método se usa un cliente LDAP (p.e. Ldapadd()) para agregar entradas, luego que la base de datos ha sido creada. Para realizar esto hay que setear las siguientes opciones antes de comenzar a ejecutar slapd en el archivo de configuración:

```
suffix <dn>
```

Esta opción define que entradas son almacenadas en la base de datos. Se debe setear el DN de la raíz del subárbol que se quiere crear, por ejemplo:

```
suffix "dc=example,dc=com"
```

y especificar el directorio en donde los archivos serán creados:

```
directory <directory>
```

Por ejemplo:

```
directory /usr/local/var/openldap-data
```

---

[1] *LDAP*: **L**ightweight **D**irectory **A**ccess **P**rotocol.

[2] *ACLs*: **A**ccess **C**ontrol **L**ist.

[3] *SSL/TLS/SASL*: **S**ecure **S**ocket **L**ayer/ **T**ransport **L**ayer **S**ecurity/ **S**imple **A**uthentication and **S**ecurity **L**ayer. Proporciona sus servicios de seguridad cifrando los datos intercambiados entre el servidor y el cliente, con un algoritmo de cifrado.

[4] *DNS*: **D**omain **N**ame **S**ystem, es un sistema de nombres que permite traducir de nombre de dominio a dirección IP y viceversa.

hay que crear este directorio con los permisos adecuados para que *slapd* pueda escribir en él y configurar *slapd* para conectarse a él como un usuario con permiso de agregar entradas. Se puede crear un super usuario o un usuario raíz con este propósito, esto se hace a través de las siguientes opciones en la definición de la base de datos:

```
rootdn <dn>
rootpw <passwd>
```

Por ejemplo:

```
rootdn "cn=Manager,dc=example,dc=com"
rootpw secret
```

estas opciones especifican el DN (nombre de dominio) y la clave que son usadas para autenticarse como super-usuario de la base de datos.

Finalmente, se deben definir los índices que se quieren mantener sobre la base de datos:

```
index {<attrlist> | default} [pres,eq,approx,sub,none]
```

Por ejemplo:

```
index cn,sn,uid pres,eq,approx,sub
index objectClass eq
```

Una vez que ésta todo configurado se ejecuta *slapd*, se conecta con un cliente LDAP y se comienza a agregar entradas. Por ejemplo, para agregar las entradas *organization* y un *organizational* role usando la herramienta *ldapadd*, se puede crear un archivo LDIF llamado entries.ldif que contenga lo siguiente:

```
# Organization for Example Corporation
dn: dc=example,dc=com
objectClass: dcObject
objectClass: organization
dc: example
o: Example Corporation
description: The Example Corporation

# Organizational Role for Directory Manager
dn: cn=Manager,dc=example,dc=com
objectClass: organizationalRole
cn: Manager
description: Directory Manager
```

y luego se usa el siguiente comando para crear la entrada:

```
ldapadd -f entries.ldif -x -D "cn=Manager,dc=example,dc=com" -w
secret
```

Crear una base de datos off-line

Las herramientas para crear una base de datos off-line leen el archivo de configuración de *slapd* y un archivo de entrada que contiene la definición de las entradas a ser agregadas.

Nuevamente con este método es muy importante tener seteadas las opciones del archivo de configuración de *slapd* previamente a la creación de la base de datos.

Replicación

Para ciertas configuraciones, una simple instancia de *slapd* puede ser suficiente para manejar un conjunto de clientes requiriendo servicio de directorio

vía LDAP. Muchos consideran necesario ejecutar más de una instancia de *slapd*. En muchos sitios, hay varios servidores *slapd* para una misma instancia: uno maestro y uno o más esclavos. Esto brinda una forma simple y efectiva de incrementar la capacidad, la disponibilidad y la confiabilidad.

Slurpd brinda la capacidad de que *slapd* maestro propague los cambios a las instancias esclavas de él mismo, implementando un esquema de replicación maestro/esclavo. *Slurpd* corre sobre el mismo host en el cual está corriendo *slapd* maestro.

Slurpd provee el servicio de replicación "in band". Esto es, usa el protocolo LDAP para actualizar las bases de datos esclavas desde la maestra.

Cuando *slapd* es configurado para generar un archivo de replicación (logfile), éste escribe los cambios a los registros en el archivo en formato LDIF. El log de replicación nos da el sitio de la réplica, el timestamp, el DN de la entrada que está siendo modificada y una serie de datos que indican los cambios realizados.

Las opciones de línea de comando más comúnmente usadas por *slurpd* son:

```
-d <level> | ?
```

Esta opción setea el nivel de debug para *slurpd*. Los niveles posibles son.

Level	Description
4	heavy trace debugging
64	Configuration file processing
65535	enable all debugging

```
-f <filename>
```

Esta opción especifica un archivo de configuración de *slapd* alternativo. *Slurpd* no tiene que tomar su propio archivo de configuración. Toda la información de configuración es leída desde el archivo de configuración de *slapd*.

```
-r <filename>
```

de este modo, se especifica un archivo de log alternativo para la replicación *slapd*.

Construcción de un servicio de directorio distribuido

Para muchos sitios, ejecutar uno o más slapd y tener el árbol entero de datos es suficiente. Pero a menudo es deseable tener un slapd con referencias a otro servicio de directorio para cierta parte del árbol.

Slapd soporta conocimiento de información subordinada y superior. El conocimiento de la información subordinada es mantenido a través de los objetos *referral*. Esta información puede ser proporcionada para delegar un subárbol. Los objetos *referral* actúan como punto de delegación, encolando dos servicios juntos. Este mecanismo permite construir servicios de directorios jerárquicos.

Un objeto *referral* es una clase de objeto estructurada y tiene el mismo DN que el subárbol delegado en él. Generalmente, proveerá una clase de objeto auxiliar denominada *extensibleObject*. Esto permite que la entrada contenga valores apropiados para el RDN. Por ejemplo, si el servidor a.example.net mantiene dc=example, dc=net y desea delegar el subárbol ou=subtree, dc=example, dc=net al servidor b.example.net, el siguiente objeto *referral* debe ser agregado a a.example.net:

```
dn: dc=subtree,dc=example,dc=net
objectClass: referral
```

```
objectClass: extensibleObject
dc: subtree
ref: ldap://b.example.net/dc=subtree,dc=example,dc=net
```

El servidor usa esta información para generar las referencias y buscar continuaciones en los servidores subordinados.

La información de nivel superior puede ser especificada usando la directiva *referral*. El valor es una lista de URIs[5] referenciando a un servicio de directorio de nivel superior. Para servidores sin superiores inmediatos, tal como a.example.net del ejemplo anterior, el servidor puede ser configurado para usar el servicio de directorio con conocimiento global, tal como *OpenLDAP Root Service* (www.OpenLDAP.org/faq/index.cgi?file=393)

```
referral        ldap://root.openldap.org/
```

sin embargo a.example.net es el superior inmediato de b.example.net, el cual debe ser configurado como sigue:

```
referral        ldap://a.example.net/
```

Agregar, modificar y borrar objetos referral es generalmente hecho usando ldapmodify() o herramientas similares.

Configuración e Instalación

Para instalar y configurar la herramienta hay que tener en cuenta los siguientes hitos.

Pre-requisitos de software

OpenLDAP software necesita paquetes de software distribuidos por terceras partes. Dependiendo de las características que se necesiten usar, se deben bajar e instalar paquetes de software adicionales.

Transport Layer Security

Los clientes y servidores OpenLDAP requieren la instalación de librerías OpenSSL o TLS para proveer servicios de seguridad en la capa de transporte. Los sistemas operativos proveen estas librerías como parte de su sistema base o como una componente de software adicional. OpenSSL [Ref. 3] a menudo requiere una instalación separada.

OpenLDAP Software no compilará al menos que detecte una instalación disponible de OpenSSL.

Servicio de autenticación Kerberos[6]

Los clientes y servidores OpenLDAP soportan servicio de autenticación basado en Kerberos. En particular, OpenLDAP soporta los mecanismos de autenticación SASL/GSSAPI usando cualquiera de los siguientes paquetes Heimdal [Ref. 4] o MIT

---

[5] *URIs*: **U**niversal **R**esource **I**dentifier, corresponden a una forma de encapsular un nombre en un espacio de nombres registrados, y etiquetarlo con el espacio de nombres, produciendo un miembro del conjunto universal

[6] *Kerberos*: es un protocolo de seguridad para realizar servicios de autenticación en la red, el cual usa criptografía basada en claves secretas para la seguridad de las contraseñas en la red.

Kerberos V [Ref. 5]. Si se decide usar autenticación Kerberos SASL/GSSAPI, debe instalar cualquiera de estos dos paquetes: Heimdal o MIT Kerberos V.

Es recomendable usar servicios de autenticación fuerte, tal como provee Kerberos.

Autenticación simple y capa de seguridad

Los clientes y servidores OpenLDAP requieren la instalación de las librerías Cyrus's SASL [Ref. 6] para proveer autenticación simple y capa de seguridad.

Cyrus SASL usará las librerías de OpenSSL y Kerberos/GSSAPI, las cuales deberán estar preinstaladas.

Software de base de datos

Slapd es el backend principal de base de datos de OpenLDAP, BDB, el cual requiere Sleepycat Software Berkeley DB [Ref. 7], versión 4. Si no está disponible al momento de la configuración, no podrá ser capaz de construir *slapd* con este backend principal de base de datos.

Slapd es el backend LDBM. OpenLDAP soporta una variedad de administradores de base de datos incluyendo Berkeley DB y GDBM [Ref. 8].

Threads

OpenLDAP fue diseñado para tomar ventaja de los threads. Soporta POSIX *pthreads*, Mach *CThreads*, y otras variedades.

TCP Wrappers

Slapd soporta TCP Wrappers (filtro de control de acceso a nivel IP) si están preinstalados. Usar TCP Wrappers u otro filtro de acceso a nivel IP (tal como los provistos por un IP-level firewall) es recomendable para el contenido de la información no pública de los servidores.

Los pasos generales para la instalación son:

1. Obtener el software: desde el sitio http://www.openldap.org/software/download/, se obtuvo el siguiente archivo: openldap-2.0.25.tgz

2. Desempaquetar el software: se utilizó el siguiente comando:
   gunzip -c openldap-2.0.25.tgz | tar xvfB -

3. Leer la documentación: se deberá leer los siguientes documentos COPYRIGHT, LICENSE, README e INSTALL .

4. Ejecutar el comando configure: se necesitará ejecutar el script configure provisto para construir el sistema. Este script acepta muchas opciones en la línea de comandos, para ver una lista de ellas se puede hacer:
   ./configure --help

5. Construir el software: este paso tiene dos partes, primero se construirán las dependencias y luego se compilará el software, con los siguientes comandos, los cuales se deberán completar sin error:

make depend
make

6. Testear lo construído: para asegurarse que todo resultó bien, se debe ejecutar el test suite:
   make test

7. Instalar el software: el ambiente ya está preparado para instalar el software; ésto usualmente requiere privilegios de súper usuario:

   root -c 'make install'

   Todo deberá ser instalado en el directorio /usr/local o en el prefijo de instalación que fue usado en el configure.

8. Editar el archivo de configuración: se debe editar el archivo *slapd.conf* (usualmente se encuentra instalado en /usr/local/etc/openldap/slapd.conf) para obtener la definición de la base de datos de la siguiente forma:
   database bdb
   suffix "dc=<MY-DOMAIN>,dc=<COM>"
   rootdn "cn=Manager,dc=<MY-DOMAIN>,dc=<COM>"
   rootpw secret
   directory /usr/local/var/openldap-data

   Reemplazar <MY-DOMAIN> y <COM> con el dominio apropiado

Experiencia en la implementación

En primer lugar se instaló OpenLDAP 2.0.25 sobre SUSE en la PC 163.10.10.2, para lo cual se siguieron los pasos de instalación requeridos, sin configurar nada en particular. Durante dicho proceso los pasos fueron:

Los archivos de configuración de OpenLdap son: sladp.conf y LDAP.conf. slapd.conf fue configurado de la siguiente manera:

```
###slapd.conf###
# $OpenLDAP: pkg/ldap/servers/slapd/slapd.conf,v 1.8.8.7 2001/09/27
20:00:31
kurt Exp $
#
# See slapd.conf(5) for details on configuration options.
# This file should NOT be world readable.
#

/*EN ESTA LINEA SE INDICA CUAL ES EL ESQUEMA QUE SE VA A UTILIZAR PARA
EL DISEÑO DE ARBOL */

include /usr/local/etc/openldap/schema/core.schema

# Define global ACLs to disable default read access.

# Do not enable referrals until AFTER you have a working directory
# service AND an understanding of referrals.

#referral       ldap://root.openldap.org
```

```
#pidfile         /usr/local/var/slapd.pid
#argsfile        /usr/local/var/slapd.args

# Load dynamic backend modules:
# modulepath    /usr/local/libexec/openldap
# moduleload    back_ldap.la
# moduleload    back_ldbm.la
# moduleload    back_passwd.la
# moduleload    back_shell.la

#
# Sample Access Control
#       Allow read access of root DSE
#       Allow self write access
#       Allow authenticated users read access
#       Allow anonymous users to authenticate
#

/*EN ESTA LINEA SE INDICA CUAL ES EL PERMISO DE ACCESO QUE VAN A TENER
TODOS LOS USUARIOS*/

access to * by * read

#access to dn="" by * read
#access to *
#       by self write
#       by users read
#       by anonymous auth
#
# if no access controls are present, the default is:
#       Allow read by all
#
# rootdn can always write!

#######################################################################
# ldbm database definitions
#######################################################################

database        ldbm

/*EL SUFFIX INDICA SUPUESTAMENTE CUAL ES EL NOMBRE DE LA BASE DE
DATOS*/

suffix "dc=example,dc=com"

/*SUPUESTAMENTE EN ESTE DIRECTORIO SE ENCUENTRA LA BASE DE DATOS*/

directory /usr/local/var/openldap-ldbm

#suffix         "dc=my-domain,dc=com"
#suffix         "o=My Organization Name,c=US"

rootdn          "cn=Manager,dc=example,dc=com"

#rootdn         "cn=Manager,o=My Organization Name,c=US"
# Cleartext passwords, especially for the rootdn, should
# be avoid.  See slappasswd(8) and slapd.conf(5) for details.
# Use of strong authentication encouraged.

/*CLAVE DE ACCESO DEL ADMINISTRADOR*/
```

```
rootpw          secret

replogfile /usr/local/var/openldap-ldbm/slapd.replog

replica host=slave1.example.com:389
        binddn="cn=Replicator,dc=example,dc=com"
        bindmethod=simple credentials=secret

# The database directory MUST exist prior to running slapd AND
# should only be accessible by the slapd/tools. Mode 700 recommended.
#directory      /usr/local/var/openldap-ldbm
# Indices to maintain

/*INDICA CUALES SON LOS INDICES CREADOS*/
index uid pres,eq
#index cn,sn,uid pres,eq,approx,sub
index objectClass eq

/*INDICA LOS DISTINTOS NIVELES DE ACCESO*/

access to attr=userpassword
       by self write
       by anonymous auth
       by dn="cn=Admin,dc=example,dc=com" write
       by * none
access to *
       by self write
       by dn="cn=Admin,dc=example,dc=com" write
       by users read


###LDAP.conf###
```

cambiamos el archivo ../openldap/LDAP.conf, en él modificamos la BASE =
y HOST = 163.10.10.60:386
####

Antes de poder realizar alguna acción sobre la base de datos es necesario ejecutar el daemon que nos da el servicio (usr/local/etc/libexec/slapd) para que se cree el directorio. Una vez hecho ésto se comenzaron a probar las operaciones brindadas por OpenLdap sobre dicho directorio.

Se agregaron entradas al directorio a través del comando *ldapadd* ejecutado sobre el mismo servidor, utilizando archivos LDIF. Por ejemplo, se crearon los archivos entries.ldif, el cual contiene lo siguiente:

Entries.ldif

dn: o=UNLP, c=AR
objectclass: top
objectclass: organization
o: UNLP
postaladdress: 50 y 115 La Plata
postalcode: 1900
telephonenumber: +54 221 4226666

Para poder comprobar el correcto funcionamiento del comando anterior se realizó un volcado del directorio. Dicho volcado se realiza a través del comando

slapCat y genera un archivo LDIF con el contenido del directorio al momento de ejecutarse, el archivo generado contiene los datos tanto de entries.ldif, como entries2.ldif.

Para comprobar que todo lo realizado desde en el servidor puede ser visto por un usuario LDAP conectado al mismo, se utilizó el cliente GQ. Al comienzo surgieron inconvenientes para configurar la herramienta correctamente, luego de tener experiencia con el uso de la misma se logró configurar el software para que se conecte con el servidor LDAP (*VER Anexo GQ*). La configuración se debe realizar tomando como base el archivo slapd.conf de OpenLdap [Ref.9].

Al quedar el cliente LDAP bien configurado se pudo conectar con el servidor y realizar sobre él las operaciones antes mencionadas a través de su interfaz gráfica.

Para poder almacenar certificados en el directorio se agregaron en el archivo slapd.conf los esquemas correspondientes, como puede observarse a continuación:

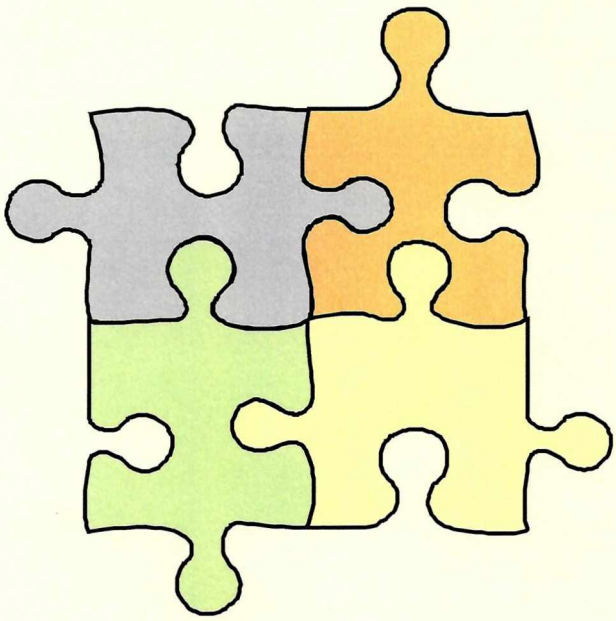
```
###slapd.conf###
# $OpenLDAP: pkg/ldap/servers/slapd/slapd.conf,v 1.8.8.7 2001/09/27
20:00:31
kurt Exp $
#
# See slapd.conf(5) for details on configuration options.
# This file should NOT be world readable.
#

/*EN ESTA LINEA SE INDICA CUAL ES EL ESQUEMA QUE SE VA A UTILIZAR PARA
EL DISEÑO DE ARBOL */

include /usr/local/etc/openldap/schema/core.schema
include /usr/local/etc/openldap/schema/cosine.schema
include /usr/local/etc/openldap/schema/inetOrgPerson.schema
```

Referencias

[Ref. 1] www.linuxmafia.com/presentations/OpenLDAP-svga/text10.html
[Ref. 2] Linux.about.com/cs/linux101/g/OpenLDAP.html
[Ref. 3] http://www.openssl.org/.
[Ref. 4] http://www.pdc.kth.se/heimdal/
[Ref. 5] http://web.mit.edu/kerberos/www/.
[Ref. 6] http://asg.web.cmu.edu/sasl/sasl-library.html
[Ref. 7] http://www.sleepycat.com/download.html.
[Ref. 8] ftp://ftp.gnu.org/pub/gnu/gdbm/
[Ref. 9]http://fc.kuh.kumamoto-u.ac.jp/~jsato/ldapmemo/ldapsetup.htm
[Ref. 10] http://www.ietf.org/rfc/rfc2587.txt
http://www.zvon.org/tmRFC/RFC2587/Output/chapter3.html
http://www.tldp.org/HOWTO/LDAP-Implementation-HOWTO/certificates.htm

[Ref. 11] http://www.sentrisystems.com/products/whitepapers/sentrisystems_vdc.pdf
http://www.aboveground.cx/~rjmooney/projects/misc/clientcertauth.html

Integración de los componentes



Integración de los componentes

Montado de la estructura

Para poder montar la infraestructura necesaria para la realización de este trabajo de tesis, se utilizaron distintas componentes de software, como se ha mencionado anteriormente. Las versiones utilizadas de dichas componentes son:

OpenLDAP 2.0.25
OpenSSL 10.9.7
OpenCA 0.9.1
Servidor web: apache
Perl 5.8.4

Introducción

Una vez instaladas y funcionando OpenLDAP y OpenCA, comenzó la etapa de integración para que trabajen en conjunto. Esto es, utilizar OpenLDAP como repositorio de los certificados emitidos y aprobados por OpenCA.

La idea general para el funcionamiento en conjunto de estas herramientas es la siguiente:

1. el usuario realiza, a través de la interfaz pública de la CA, un requerimiento de certificado
2. luego ese requerimiento es tomado desde la RA, para ser aprobado por la misma
3. una vez realizado lo anterior, el requerimiento puede ser tomado por la CA, quien emite el certificado y lo publica en OpenLDAP.

A continuación se detallarán los distintos pasos realizados y las dificultades encontradas para lograr este objetivo.

Experiencia

Como se indicó en los capítulos referidos a OpenCA y OpenLDAP, ambas componentes están instaladas e inicializadas para que funcionen juntas. Esto se debe a que OpenLDAP se instaló con anterioridad a OpenCA y en la instalación y configuración de OpenCA se indicaron los parámetros necesarios para que reconozca dicha instalación.

Los archivos de configuración que debieron ser modificados son: slapd.conf y ldap.conf pertenecientes a OpenLDAP y ca.conf, ca_node.conf, pub.conf y ra.conf pertenecientes a OpenCA. A consecuencia de ésto se deduce que los siguientes parámetros de los diferentes archivos de configuración, deben contener el mismo valor:

En slapd.conf

```
# The base of your directory in database #1
suffix        "o=Linti,c=AR"

# Where the database file are physically stored for database #1
directory     "/var/lib/ldap"
rootdn              "cn=root,o=Linti,c=AR"
rootpw              syperca
```

En ldap.conf:

```
## Now the LDAP default base dn
Basedn "O=Linti,C=AR"

## Let's define the privileged Account Allowed to Modify the LDAP entries
ldaproot "CN=root, O=Linti, C=AR"
ldappwd  "syperca"

LDAP_CRL_Issuer   "dalila_r@info.unlp.edu.ar"
LDAP_CA_DN        "O=Linti,C=AR"
```

En ra.conf

```
# if you have more than one OU simply add them
# this works for all possible attributes
# DN_TYPE_SPKAC_ELEMENTS "EMAIL" "CN" "OU" "OU"
DN_TYPE_SPKAC_BASE_1 "Linti"
DN_TYPE_SPKAC_BASE_2 "AR"

# if you have more than one OU simply add them
# this works for all possible attributes
DN_TYPE_IE_BASE_1 "Linti"
DN_TYPE_IE_BASE_2 "AR"
```

En ca.conf

```
# if you have more than one OU simply add them
# this works for all possible attributes
DN_TYPE_BASIC_BASE_1 "Linti"
DN_TYPE_BASIC_BASE_2 "AR"
```

En pub.conf:

```
# if you have more than one OU simply add them
# this works for all possible attributes
DN_TYPE_TOKEN_BASE_1 "Linti"
DN_TYPE_TOKEN_BASE_2 "AR"

# if you have more than one OU simply add them
# this works for all possible attributes
DN_TYPE_SPKAC_BASE_1 "Linti"
DN_TYPE_SPKAC_BASE_2 "AR"

# if you have more than one OU simply add them
# this works for all possible attributes
DN_TYPE_IE_ELEMENTS "emailAddress" "CN" "OU"
DN_TYPE_IE_NAME     "Basic User Request"

DN_TYPE_IE_BASE_1 "Linti"
DN_TYPE_IE_BASE_2 "AR"
```

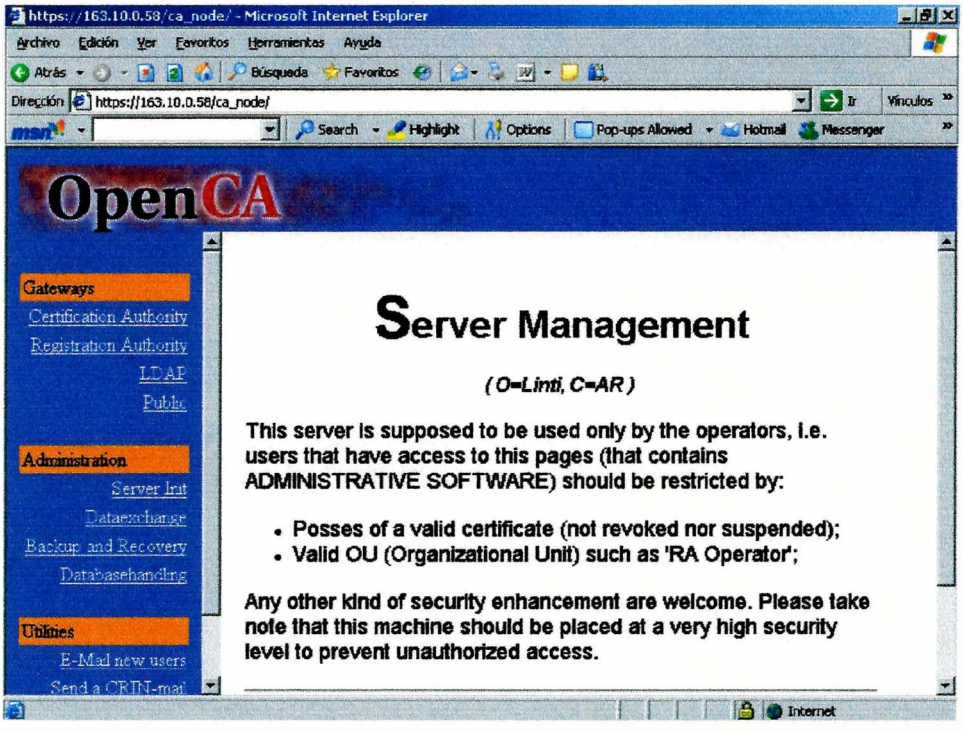
En ca_node.conf:

```
## Now the LDAP default base dn
basedn "O=Linti, C=AR"
```

```
## Let's define the privileged Account Allowed to Modify the LDAP entries
ldaproot "CN=root, O=Linti, C=AR"
ldappwd  "syperca"

## Let's define some Directory Env
ldapbasedir "/usr/local"
```

En el resto de los archivos no se configuró nada en particular.

Como primer paso se realizó un volcado desde OpenCA hacia OpenLDAP, ésto se hace a través de la opción *ServerManagement/ LDAP*, como se ve en la siguiente pantalla:



Al cliquear sobre el link LDAP dio el siguiente error:

Not found

Esto fue solucionado agregando la siguiente línea en el http.conf

```
ScriptAlias /cgi-bin/ldap/ "/usr/local/openca-online/apache/cgi-bin/ldap/"
```

A continuación, se procedió con la operación de volcado y, como resultado de l misma, se deberían visualizar los certificados del Operador de la CA, operador de la RA y del sitio web, los cuales fueron emitidos durante las fases de inicialización de OpenCA; en cambio, ninguno de estos certificados pudo ser visualizado. Al ingresar a la opción *LDAP/ CACertificate*, la misma indicaba que necesitaba el objectClass 'pkiCA', por lo tanto se agregó el esquema **pki.schema.**, en el archivo de configuración de OpenLDAP (slapd.conf), como resultado de esto y al realizar nuevamente el volcado, se muestra lo siguiente:

Adding valid CA-certificates to the LDAP server ...

Certificate 0 FAILED (error 1: LDAP-bind failed: I/O Error Resource temporarily unavailable)

Por lo tanto, se chequearon los permisos de lectura y escritura sobre las distintas componentes, hasta indicar los permisos a nivel de atributo:

```
access to    attribute=caCertificate
    by    *    write
```

Esto se hace en el archivo de configuración de OpenLDAP. Al probar nuevamente la operación, se obtuvo el mismo error.

Se modificaron los archivos ra.conf, ca.conf, ca_node.conf y pub.conf en la línea

##EXPORT_IMPORT_LOCAL_DEVICE "/dev/fd0"

por EXPORT_IMPORT_LOCAL_DEVICE "/tmp/openca.tar"

Luego se produjo el siguiente error:

Importing CA-Certificates into ldap ...
Cannot write CA-Certificate 2209dd510b12efd1dc0cb973e8a7a8e1 to LDAP

Fue solucionado utilizando el mail [Ref. 2], en el cual recomienda hacer lo siguiente:

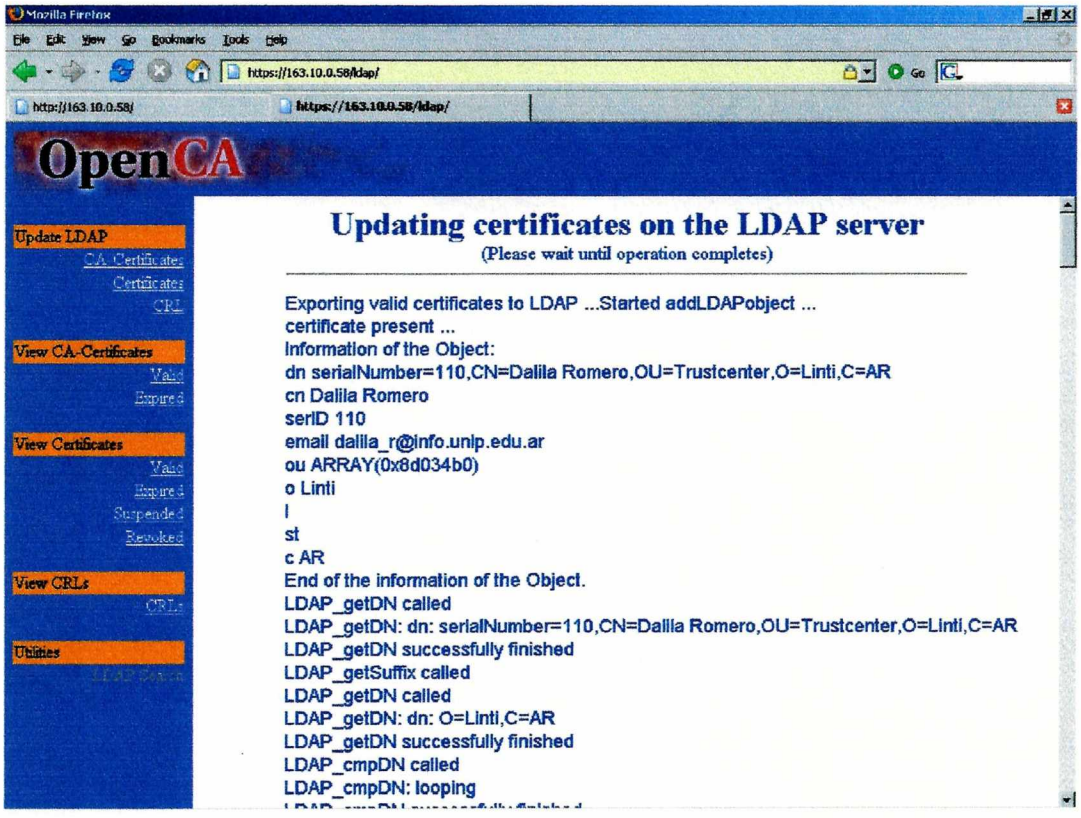
En el método connect de OpenLDAP, ubicado en la librería `LDAP.pm`, **se debieron comentar las líneas de Timeout:**

```
sub _connect {
  my ($ldap, $host, $arg) = @_;

  $ldap->{net_ldap_socket} = IO::Socket::INET->new(
    PeerAddr => $host,
    PeerPort => $arg->{port} || '389',
    Proto    => 'tcp',
#    Timeout  => defined $arg->{timeout}
#                ? $arg->{timeout}
#                : 120
  );
}
```

Finalmente, con los seteos realizados antes, se logró la correcta integración de OpenCA con OpenLDAP.

Para actualizar OpenLDAP se debe ejecutar la operación correspondiente desde la CA.

Ahora, al hacer el volcado aparecen todos los nodos con los certificados, como lo muestra la siguiente pantalla:


Mozilla Firefox
File Edit View Go Bookmarks Tools Help
https://163.10.0.58/ldap/
http://163.10.0.58/
https://163.10.0.58/ldap/
OpenCA
Update LDAP
CA Certificates
Certificates
CRL
View CA-Certificates
Valid
Expired
View Certificates
Valid
Expired
Suspended
Revoked
View CRLs
CRLs
Utilities
Updating certificates on the LDAP server
(Please wait until operation completes)
Exporting valid certificates to LDAP ...Started addLDAPobject ...
certificate present ...
Information of the Object:
dn serialNumber=110,CN=Dalila Romero,OU=Trustcenter,O=Linti,C=AR
cn Dalila Romero
serID 110
email dalila_r@info.unlp.edu.ar
ou ARRAY(0x8d034b0)
o Linti
l
st
c AR
End of the information of the Object.
LDAP_getDN called
LDAP_getDN: dn: serialNumber=110,CN=Dalila Romero,OU=Trustcenter,O=Linti,C=AR
LDAP_getDN successfully finished
LDAP_getSuffix called
LDAP_getDN called
LDAP_getDN: dn: O=Linti,C=AR
LDAP_getDN successfully finished
LDAP_cmpDN called
LDAP_cmpDN: looping

Caso práctico

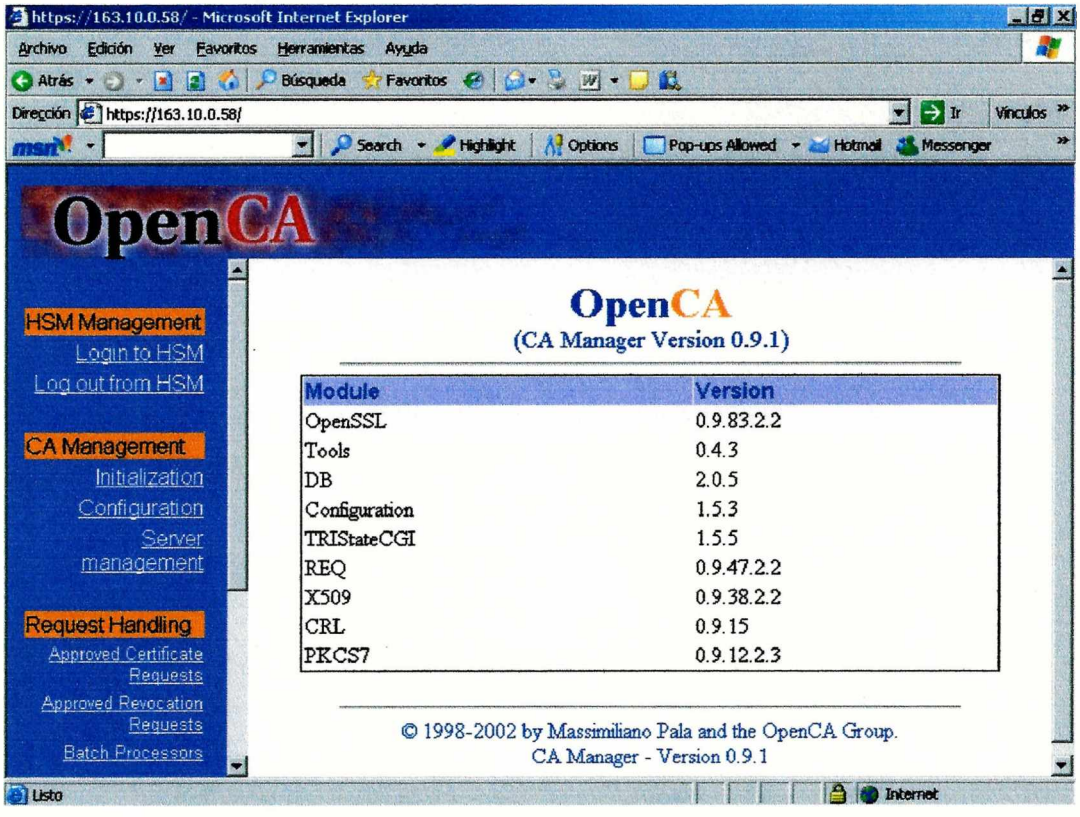
A continuación se detallarán cada uno de los pasos desde que un usuario hace un requerimiento de certificado hasta que la CA lo emite y lo publica en un servidor LDAP.

Los pasos generales son los detallados al comienzo de este capítulo. Una vez que hay un requerimiento de usuario pendiente, la CA es la encargada de aprobar y firmar el certificado (lo emite), y la RA es la que se encarga de realizar las tareas administrativas, tales como verificar los datos del usuario, el cuál debe hacerse presente en las instalaciones de la RA con sus documentos en mano.

Cómo obtener un certificado

El procedimiento para que un usuario o entidad pueda obtener un certificado es el siguiente:

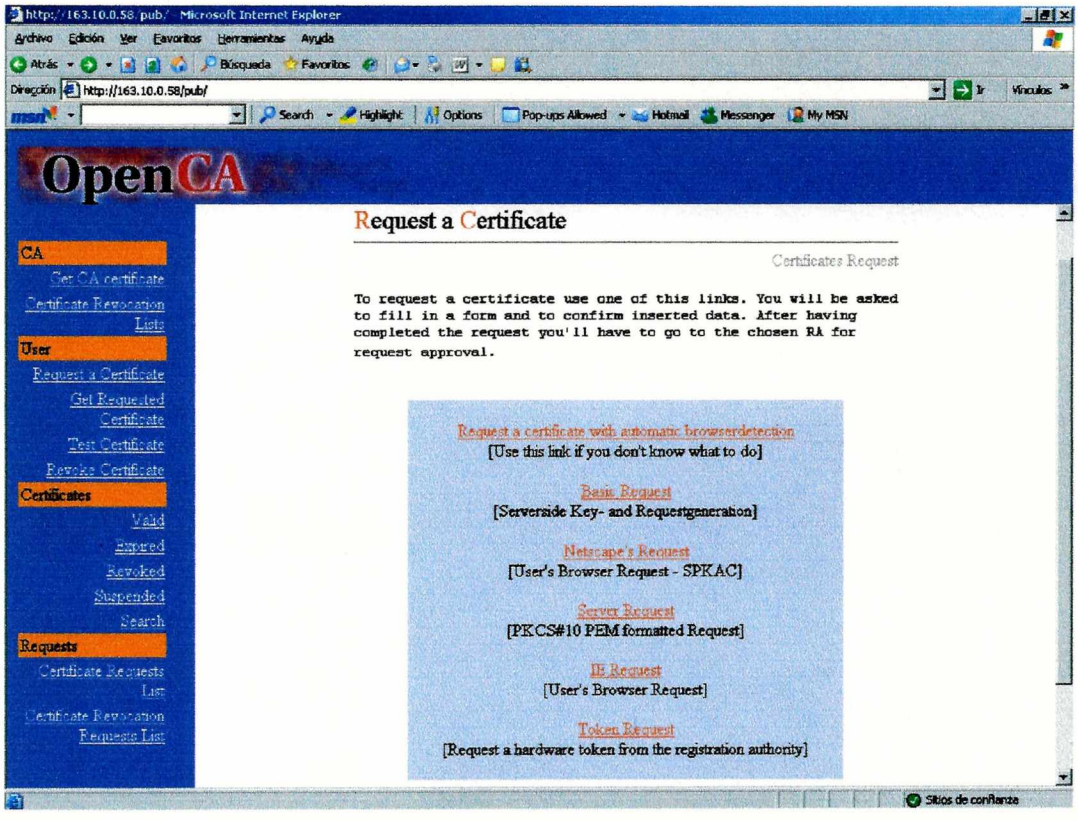
1) Ingresar a través de Internet al sitio de la autoridad de certificación (CA). Se mostrará una pantalla similar a la siguiente:



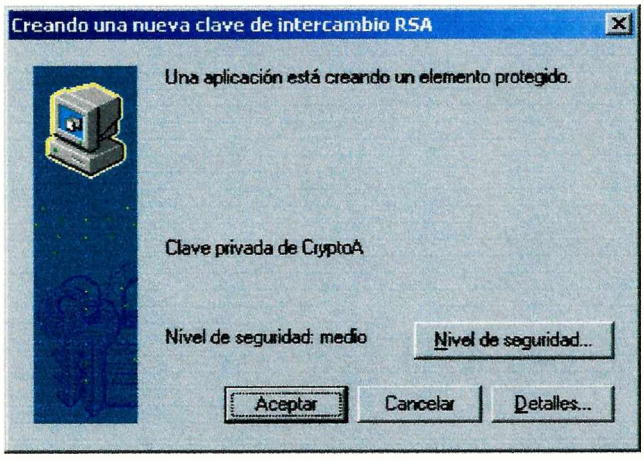
2) Descargar el certificado de la CA en su computadora, esto se hace a través de la opción *Get CA Certificate*. Luego configurar su browser[6] para que éste confíe en la CA.

[6] *Browser:* Utilitario que se usa para navegar sobre una red de computadoras.

3) Requerir un certificado, para lo cual se debe elegir la opción correspondiente según el browser que se tenga, o bien usar la opción de detectar automáticamente el browser. Se mostrará una pantalla similar a la siguiente:

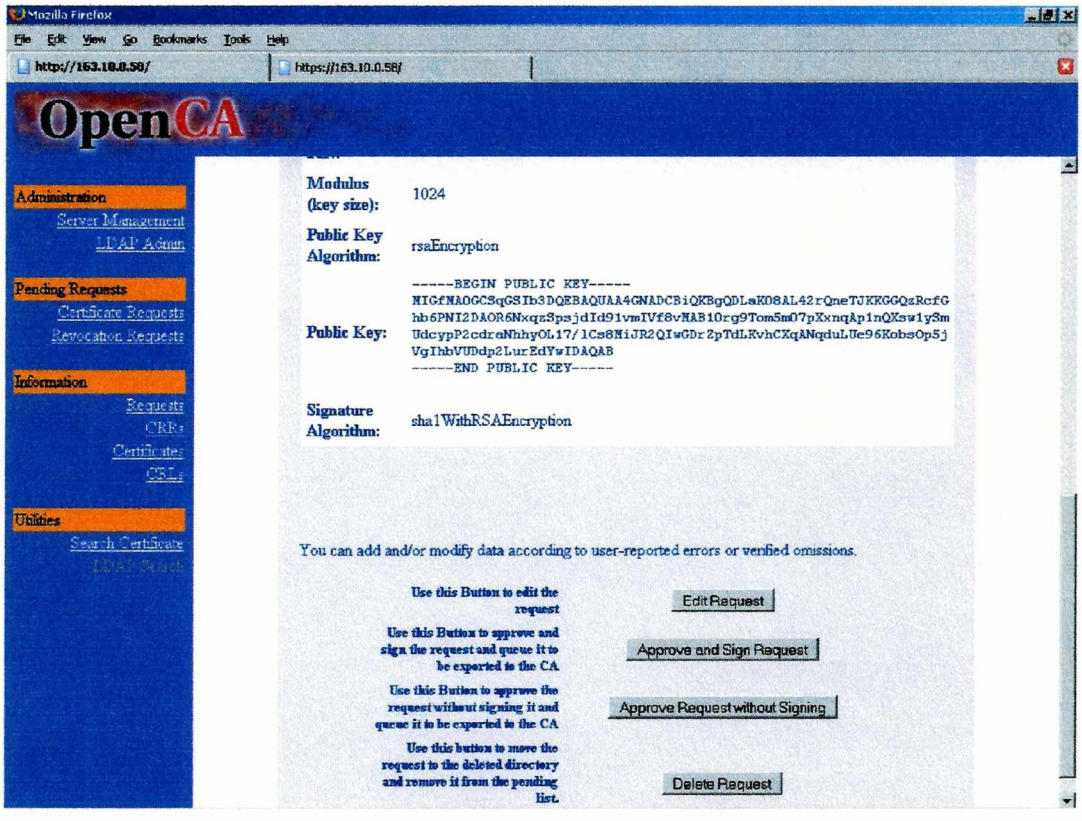


Para completar la solicitud se deben indicar los datos pedidos por la misma, tales como, nombre, correo electrónico, clave pública, etc. Una vez completados estos pasos se ejecutará un proceso, el cual creará la clave privada.

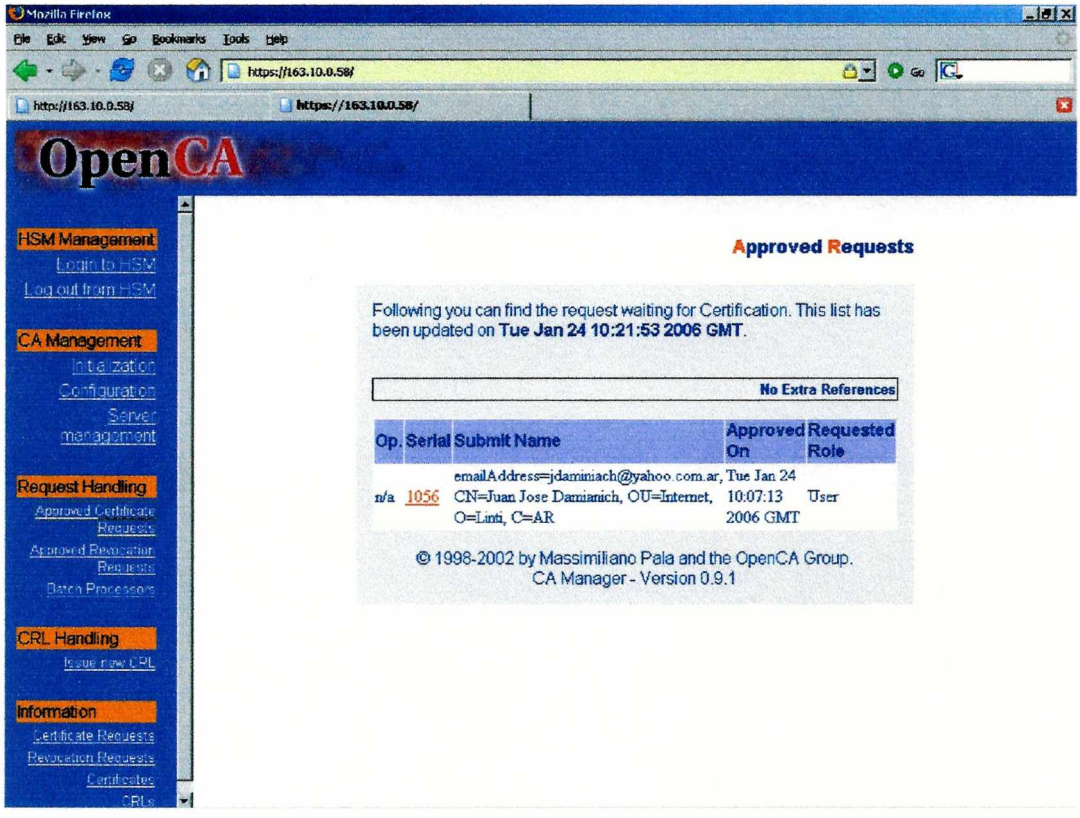


4) Dirigirse personalmente a la autoridad de registración (RA), con la documentación requerida por ésta. Con dicha documentación se llevará a cabo la identificación y solicitará el certificado.

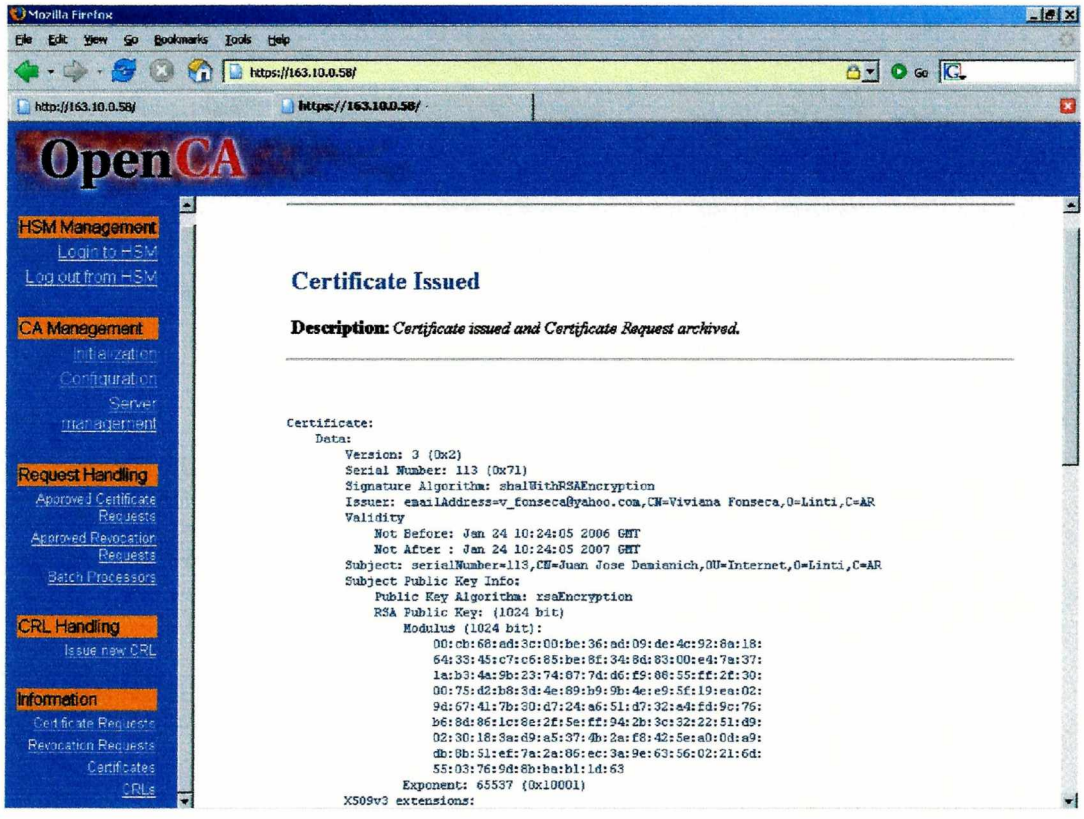
5) Comprobar que la solicitud de certificado sea correcta y firmarla en presencia del agente de registro. La autoridad de registración usa su clave privada para firmar. De este modo la RA aprobará dicha solicitud:



6) La CA pasa a tener el certificado aprobado por la RA para emitirlo:



Se selecciona el requerimiento a ser emitido, para lo cual se debe ingresar la clave de la CA y como resultado se obtiene el certificado emitido:



7) Una vez emitido el certificado, éste es publicado en LDAP y el usuario podrá utilizarlo.

Cómo revocar un certificado

Para poder revocar un certificado, el usuario debe dirigirse a la autoridad de registración, acompañado de la documentación requerida por esta.

El agente de registro verificará la identidad y solicitará la revocación.

La revocación del certificado puede ser solicitada, por ejemplo, cuando:

1) Se pierde, se rompe o sea robada la clave privada.
2) Alguien no autorizado haya tenido acceso a la clave privada.

Período de validez de los certificados

Todos los certificados tienen un período de validez, que consta de una fecha de comienzo y otra de finalización. Si la fecha de nuestra computadora está fuera de este intervalo de fechas, el browser[1] considerará que el certificado es inválido porque ha prescripto.

[1] *Browser*: herramienta para navegar en la web. Presenta una interfaz gráfica

Referencias

[Ref. 1] http://www.geocrawler.com/archives/3/10403/2002/11/0/10238445/
[Ref. 2] http://www.mail-archive.com/openca-users@lists.sourceforge.net/msg02981.html
[Ref. 3] http://sources.redhat.com/ml/cygwin/2001-09/msg01665.html

Servidor Web

Servidor web

Apache

Es el servidor http (Protocolo de Transferencia de HyperTexto) más utilizado en Internet, según las grandes empresas de estadística de la red: "El 80% de los servidores web de Internet utilizan Apache o derivados del mismo. El 20% restante esta conformado por software de otras empresas....".

Apache es un servidor web poderoso, flexible, rápido, eficiente, altamente configurable y extensible a través de módulos, que podemos obtener en Internet o escribir nosotros mismos. Gracias a ser modular se han desarrollado diversas extensiones entre las que destaca PHP, un lenguaje de programación del lado del servidor.

Es Open Source y consecuentemente, nos provee todo su código fuente y documentación. Es un esfuerzo enteramente voluntario, completamente fundado por sus miembros, y no por objetivos comerciales. Se desarrolla en forma abierta e incentiva la realimentación de los usuarios, obteniendo nuevas ideas, informes de fallos y parches para la solución de los mismos.

Su licencia es no-restrictiva, es multiplataforma, es decir funciona en la mayoría de los sistemas tipo-Unix, y también en Windows NT/9x/ME, netware 5.x y OS/2. Posee las siguientes funciones:

- Autenticación de usuarios a través de bases de datos DBM
- Configuración de mensajes de error y problemas
- Generación dinámica de archivos index.html en base a scripts
- Capacidad de manejar alias ilimitados a páginas
- Negociación de contenidos en base a lo que un navegador le solicite
- Administración de hosts Virtuales (o sea, diferentes sitios en un mismo sistema)
- Avanzadas técnicas de administración del registro de uso del sistema.

Es una de las pocas aplicaciones servidoras que no necesita de mayores modificaciones después de su instalación a fin de funcionar. De hecho sólo se necesita que se inicie el servicio y se publiquen las páginas correspondientes.

Apache ha mostrado ser substancialmente más rápido que muchos otros servidores.

Justificación

Para realizar nuestro trabajo es necesario que diferentes aplicaciones openSource interactúen entre sí. Como la autoridad de certificación es implementada utilizando OpenCA y nos brinda sus operaciones a través de una interfaz web, utilizaremos Apache como Servidor para poder acceder a dichas operaciones.

Configuración

Al instalar OpenLDAP, se realizaron las siguientes modificaciones en la configuración de Apache:

En /usr/local/apache/conf # vi httpd.conf, se modificaron las siguientes entradas:

.....
<VirtualHost 163.10.10.60:443>
General setup for the virtual host
ServerName ra.linti.unlp.edu.ar
DocumentRoot "/usr/local/openca-online/apache/htdocs/ra"
ServerAdmin root@ca.linti.unlp.edu.ar
ErrorLog /usr/local/apache/logs/errorra_log
TransferLog /usr/local/apache/logs/accessra_log
Alias /ra/ "/usr/local/openca-online/apache/htdocs/ra/"

Se agregó el alias /ra_node/ "/usr/local/openca-online/apache/htdocs/ra_node/"

Alias /pub/ "/usr/local/openca-ca/apache/htdocs/ca/"

Alias /ldap/ "/usr/local/openca-online/apache/htdocs/ldap/"
ScriptAlias /cgi-bin/ra/ "/usr/local/openca-online/apache/cgi-bin/ra/"
ScriptAlias /cgi-bin/pub/ "/usr/local/openca-online/apache/cgi-bin/ra_node/"

ScriptAlias /cgi-bin/ca/ "/usr/local/openca-ca/apache/cgi-bin/ca/"
.....
<VirtualHost 163.10.10.60:81>
ServerName ca.linti.unlp.edu.ar
ServerAdmin pvenosa@info.unlp.edu.ar

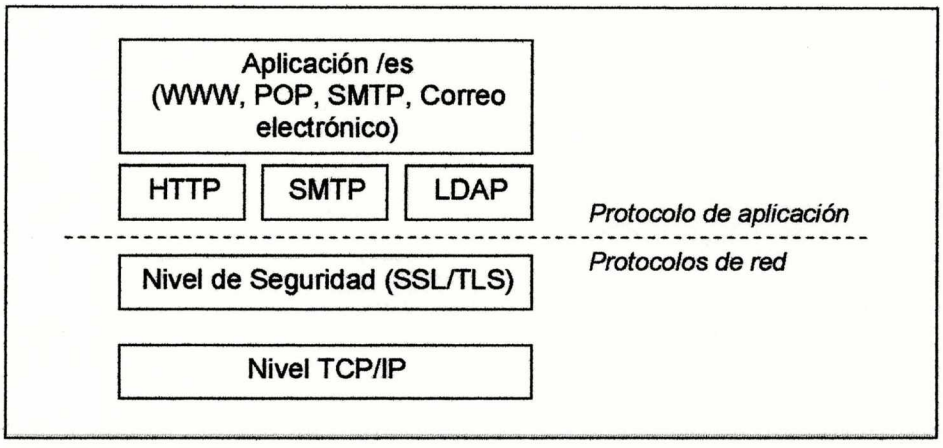
DocumentRoot /usr/local/openca-ca/apache/htdocs/ca
ErrorLog /usr/local/apache/logs/errorca_log
TransferLog /usr/local/apache/logs/accessca_log
Alias /ca/ "/usr/local/openca-ca/apache/htdocs/ca/"

Alias /ca_node/ "/usr/local/openca-ca/apache/htdocs/ca_node/"

ScriptAlias /cgi-bin/ "/usr/local/openca-ca/apache/cgi-bin/ca/"
<Directory "/usr/local/openca-ca/apache/cgi-bin/ca">
AllowOverride None
Options None
Order allow,deny
Allow from all
</Directory>

SSL y TLS

El protocolo SSL fue diseñado para proveer tanto autenticación como seguridad de datos. Se encapsula en un socket TCP/IP y básicamente, todas las aplicaciones TCP/IP pueden usarlo para tener una comunicación segura.



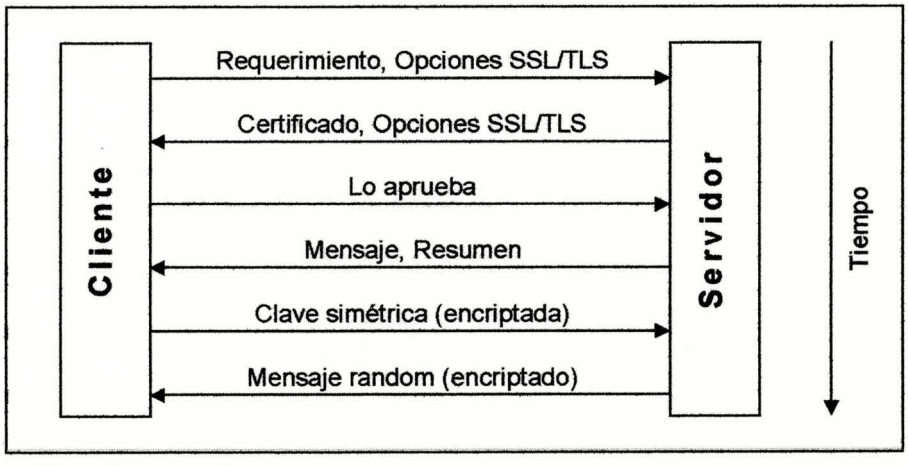
SSL/TLS en relación con otros protocolos

TLS es un estándar abierto en evolución. Está basado en SSL con el cual solamente tiene diferencias menores. Se asume que TLS reemplazará a SSL.

SSL y TLS soportan autenticación del servidor, autenticación del cliente, o autenticación mutua. Además, proveen privacidad por la encriptación de datos enviados por la red.

SSL y TLS utilizan el método de clave pública para asegurar la comunicación y autenticar la contraparte en la sesión. Esto es logrado con un par de claves pública/privada. Ellas operan como funciones inversas una de la otra, lo cual significa que los datos encriptados con la clave privada pueden ser desencriptados con la clave pública y viceversa. El par de claves pertenecientes al servidor son usualmente generadas cuando se hace el seteo del mismo.

El intercambio entre un cliente y un servidor en la negociación para la conexión SSL/TLS es explicada a continuación:



SSL/TLS handshake

SSL/TLS es utilizado para autenticar un servidor a un cliente usando su certificado y su clave privada y para negociar la clave secreta más tarde para la encriptación de datos usada.

El módulo MODSSL provee criptografía fuerte para el servidor de Web Apache vía el protocolo SSL v2/v3 y TLS[1] v1.

Las principales características de mod_ssl son:

- Es un software de libre distribución.
- Provee criptografía de 128 bits.
- Soporta los protocolos SSL v2/v3 y TLS v1.
- Soporta cifrados RSA y Diffie-Hellman.
- Se integra en forma dinámica con Apache a través de un API Extendido (EAPI).
- Provee comandos para la generación de certificados X.509v3.
- Soporta listas de revocación de certificados (CRLs).
- Brinda autenticación basada en certificados X.509 para cliente y servidor.

Este módulo[2] incluye al módulo SSL y un conjunto de actualizaciones para Apache agregando la EAPI (API extendida), la cual es un requisito esencial para mod_ssl.

Mod_ssl combina la flexibilidad de Apache y la seguridad de OpenSSL[3].

[1] *TLS* (Transport Layer Security): El protocolo SSL 3.0 es la base del TLS (Transport Layer Security) desarrollado por el IETF. Se podría decir que es SSLv3.1 ya que es compatible hacia atrás con todas las versiones de SSL y es SSL 3 pero mejorado. Difiere de SSL en que usa un conjunto un poco más amplio de algoritmos criptográficos.

[2] Para más información dirigirse a: http://www.modssl.org

[3] Para más información dirigirse a: http://www.openssl.org

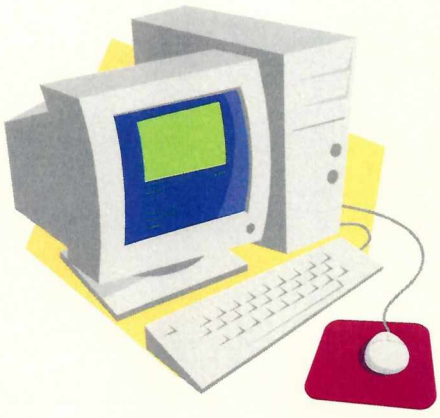
Referencias

[Ref. 1] http://httpd.apache.org/docs/howto/auth.html
[Ref. 2] http://httpd.apache.org/docs/misc/FAQ.html
[Ref. 3] http://www.freeos.com/articles/4121/

Módulo 3

Implementación

Sistema SIU-Guaraní



Sistema SIU-Guaraní

Introducción

En esta sección se va a describir, en líneas generales, el sistema que se toma como base para la implementación práctica de lo investigado en este trabajo de grado. Básicamente se describirá en que consiste el proyecto SIU, los diferentes módulos del sistema y su arquitectura.

¿Qué es el SIU?

El SIU[1] desarrolla soluciones informáticas y brinda servicios para el Sistema Universitario Nacional. Su objetivo es contribuir a mejorar la gestión de las instituciones, permitiéndoles contar con información segura, íntegra y disponible, optimizar sus recursos y lograr que el software sea aprovechado en toda su potencialidad.

Las soluciones y servicios del SIU están en permanente evolución. Los productos evolucionan en varias dimensiones: en eficiencia, en nuevos servicios, en actualizaciones tecnológicas, en la integración con otros sistemas, y en respuesta a las estrategias definidas por la SPU[2].

La realidad del sistema universitario es compleja (heterogeneidad de tamaño y población, diversidad geográfica, cultural, social, etc.) para incorporar soluciones informáticas que sean valoradas y utilizadas por todos sus actores. Este desafío llevó al desarrollo de una forma de trabajo nueva en el Estado, involucrando la participación de todos los actores en las actividades a través de comunidades de práctica donde se intercambian conocimientos y experiencias (desde la definición de las mejoras a los sistemas, hasta las actividades de capacitación y las propuestas de nuevos servicios, compartir experiencias, etc.).

Los sistemas desarrollados por el SIU toman los nombres de tribus indígenas de distintas regiones de la Argentina. Estos nombres transmiten una noción de integridad entre comunidades diversas y a su vez remiten a una identidad común.

El SIU desarrolla e implementa distintos sistemas, los cuales se aplican a diferentes áreas, como por ejemplo, liquidación de sueldos (SIU-Pampa), Tesorería (SIU-Comechingones), etc.

[1] SIU: Sistema de Información Universitario
[2] SPU: Secretaría de Políticas Universitarias

SIU-Guaraní

El SIU-Guaraní es un sistema de gestión de alumnos que registra y administra todas las actividades académicas de la universidad, desde que los alumnos ingresan como aspirantes hasta que obtienen el diploma. Fue concebido para administrar la gestión de alumnos en forma segura, con la finalidad de obtener información consistente para los niveles operativos y directivos.

El sistema brinda servicios para alumnos, docentes, usuarios administrativos y autoridades, ya que pueden explorar los datos y obtener información como soporte para toma de decisiones.

El SIU-Guaraní es un *sistema de información* y a la vez un proyecto inserto en un escenario cultural heterogéneo y complejo. Para lograr el éxito del proyecto, el SIU acompaña a las instituciones durante todo el proceso de implementación, poniendo en práctica una metodología de trabajo de carácter colaborativo en red. De esta manera se busca *crear espacios participativos y sentido de pertenencia al proyecto.*

El sistema permite mejorar el tratamiento de la información y agilizar los mecanismos de gestión académica. Además, la instalación del SIU-Guaraní en las unidades académicas permite homogeneizar sus procedimientos. Los aspectos relevantes de la implementación del sistema son:

- Facilitar la información para la gestión de Secretaría Académica y para los alumnos.
- Aligerar los trámites académicos, evitando la carga burocrática sin perder el control de los mismos.
- Brindar a las autoridades herramientas de control sobre el manejo de la gestión académica y la posibilidad de contar con la información en el momento y lugar oportuno.
- Brindar a los docentes el acceso a información sobre sus cursos (actas, inscriptos, etc.).
- Fomentar el trabajo en equipo, estableciendo un flujo de operaciones entre los distintos sectores.
- Carga de datos en la fuente de información, evitando de esta manera los errores por pasos sucesivos en forma manual.
- Posibilidad de consulta y validación de los datos ingresados.

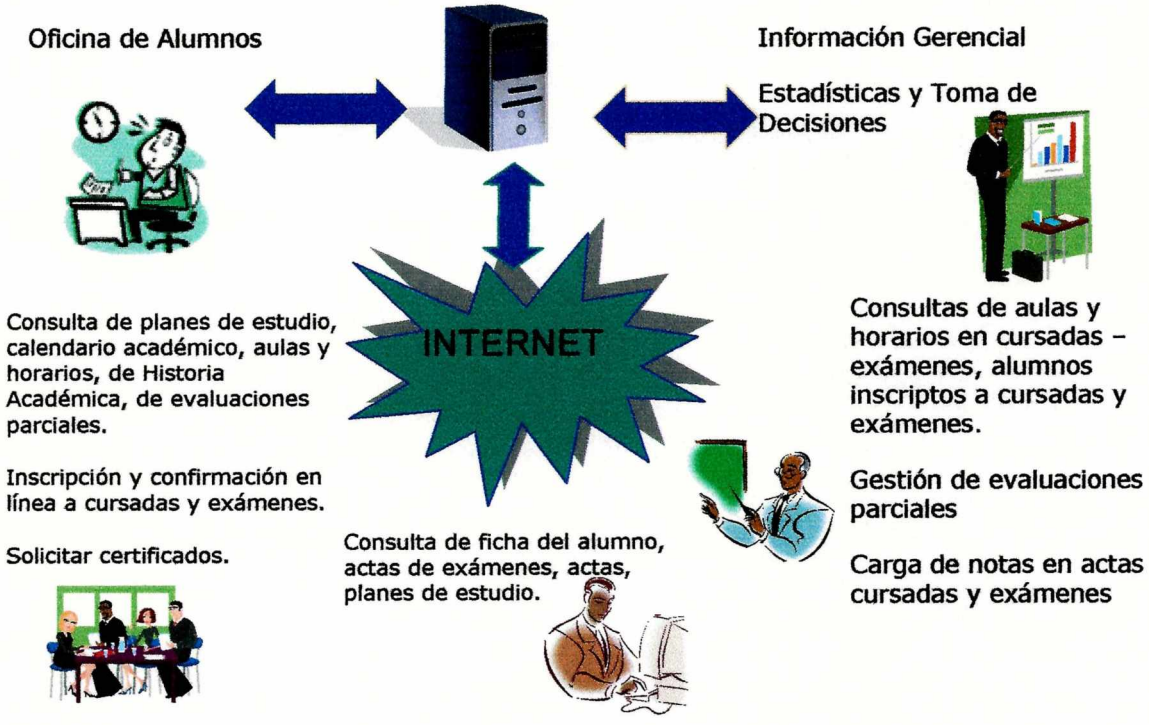
El sistema tiene una arquitectura que le permite adaptarse a las realidades que existen en las distintas Universidades.

El sistema cuenta con controles dinámicos y parámetros generales que permiten definir su comportamiento. Los parámetros generales son variables que al momento de la implementación toman un valor en función de ciertas necesidades que tenga la Unidad Académica. Los distintos valores que pueden tomar estos parámetros implican una forma diferente de operar del SIU-Guaraní, esto es parte de lo que hace flexible al sistema ante distintas realidades.

Otra herramienta para adaptarse a diferentes necesidades son los controles dinámicos. Estos controles son validaciones que se asocian a una operación (ítem de menú) y que al momento de ejecutarse la misma son disparados. La forma de comportarse de estos controles para cada operación a la que están asociados, se determina al implementar el sistema junto con los parámetros generales. El sistema provee un conjunto de controles que puede ser ampliado por la Unidad Académica sin necesidad de modificar el sistema. Ej.: Pueden agregar controles para usar en la inscripción a examen.

El cambio de cualquiera de estas definiciones (parámetros o controles) implica una modificación en la forma de funcionamiento del sistema, que debe estar respaldada por un cambio en la forma de trabajo de la Unidad Académica y debe ser manejada con precaución.

El sistema brinda para su acceso tres módulos diferentes: Gestión, autogestión y Web. El módulo de Gestión es el utilizado por los empleados de la dirección de enseñanza, y los módulos de autogestión y Web son utilizados por los alumnos y docentes, según muestra el gráfico siguiente:



Estos módulos ofrecen diferentes prestaciones, las cuales se detallan a continuación:

Módulo de gestión

- Gestión de carreras y planes.
- Planificación.
- Gestión de matrícula.
- Gestión de cursado.
- Gestión de aulas.
- Gestión de exámenes.
- Gestión de equivalencias.
- Gestión de egresados.
- Administración y emisión de certificados.
- Mensajería a casillas de e-mail o celulares (docentes/alumnos/autoridades).
- Gestión de encuestas para alumnos.
- Interfaces
  - Estadísticas generales (interfaz con sistema SIU-Araucano).
  - Generación de datos para Coneau.
  - Información de personal docente (interfaz con el sistema de gestión de personal SIU-Pampa).
  - Interfaz con el SIU-Kolla (sistema de seguimiento de egresados).
  - Interfaz con Data Warehouse (herramientas para el análisis institucional y la toma de decisiones).
  - Interfaz con SIU-Quilmes (sistema de gestión de facturación y cuenta corriente).
  - Interfaz con Moodle (software libre para gestión de educación a distancia).
- Accesos: Gestión (utilizada por el personal administrativo/docente), Autogestión (servicios a los docentes y a los alumnos a través de ventanillas electrónicas), Internet (acceso al sistema a través de Internet) y Ecófono (acceso telefónico, sin necesidad de una computadora).

Prestaciones para las autoridades, a través del módulo Web

- Consulta de ficha del alumno: carreras, regularidades, historia académica, títulos, promedios, sanciones, certificados solicitados, pérdidas de regularidad, readmisiones, etc.
- Consulta de actas de examen, actas de regulares y promociones.
- Consulta de planes de estudio.

Prestaciones para los docentes, a través del módulo Web

- Consulta de agenda de clases: comisiones asignadas y alumnos inscriptos.
- Consulta de agenda de mesas de exámenes, calidad de alumnos inscriptos (libre, regular).
- Alta y baja de evaluaciones parciales.
- Ingreso y consulta de notas de evaluaciones parciales.
- Carga de notas en actas de examen, cursado y promoción.
- Recepción y envío de mensajes.
- Creación de cursos en Moodle.

Prestaciones para los alumnos, a través del módulo Web

- Inscripción a exámenes y cursadas.
- Reinscripción a carrera.
- Consulta de créditos.
- Consulta de inscripciones, plan de estudios e historia académica.
- Consulta de cronograma de evaluaciones parciales.
- Notas de evaluaciones parciales.
- Materias regulares.
- Agenda de clases.
- Solicitud de certificados.
- Actualización de datos censales.
- Recepción de mensajes.
- Acceso a Moodle.

Descripción técnica del sistema

El sistema SIU-Guaraní, está diseñado con una arquitectura cliente-servidor. Se utilizó Power Builder 7 como herramienta de desarrollo para la parte cliente e Informix IDS 9.x como servidor de base de datos. La interfaz Web está desarrollada en PHP. Gran parte de las reglas de negocio están escritas en forma de procedimientos almacenados dentro de la base de datos.

La aplicación permite crear usuarios del sistema con perfiles particulares en donde cada usuario debe tener asignado un conjunto de operaciones que serán las únicas que puede realizar. El usuario debe tener un nombre que lo identifique y una clave para asegurar su identidad.

El sistema registra la operatoria realizada (datos modificados y su estado previo), el usuario que la realizó y el momento en que se llevó a cabo (fecha y hora). Esto permite seguir el comportamiento de un usuario determinado.

La información se encuentra almacenada en un motor de base de datos relacional que asegura la consistencia de los datos y brinda mecanismos para realizar tareas de respaldo (back up). Ante algún imprevisto que cause la caída del sistema, deben contemplarse con celeridad, las distintas opciones que brinda el sistema para reconstruir el contenido de la base de datos. La tarea de back up es responsabilidad de la Unidad Académica.

Referencias

[Ref. 1]: http://www.siu.edu.ar
[Ref. 2]:http://www.guarani-laplata.unlp.edu.ar

Desarrollo

Desarrollo

Introducción

Una vez concluida la etapa de armado y testeo de la arquitectura propuesta, en la cual se integran distintas componentes de software para poder brindar en un mismo producto seguridad en la transmisión de datos, autenticidad[1] de cada usuario, y una manera clara y consistente de organizar y manipular información a través de un servicio de directorios, se adaptará el módulo de docentes del sistema SIU-GUARANI, agregando al mismo la funcionalidad necesaria para facilitar la interacción a través de Internet, entre los docentes de la facultad y la oficina de alumnos en forma segura.

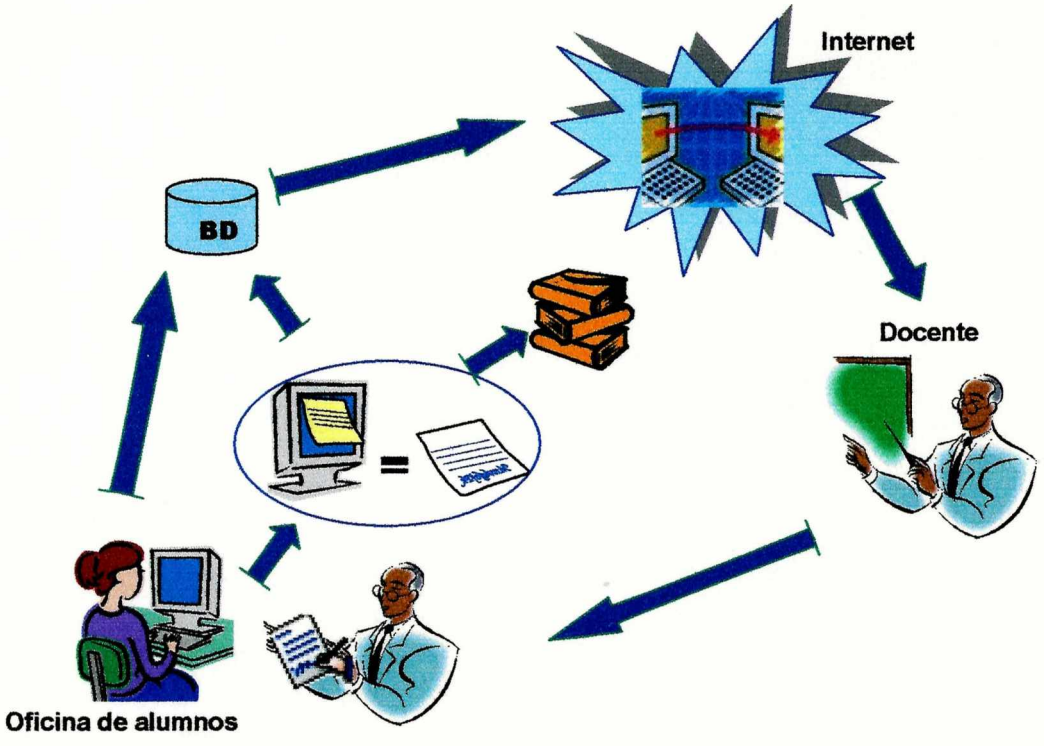
En la actualidad en cualquier carrera universitaria, los docentes de cada asignatura deben obtener de la oficina de alumnos el acta con los alumnos que cursan su asignatura, como así también las actas con los alumnos que se inscribieron para rendir el examen final en cada mesa de examen. Luego completan dichas actas manualmente, firmándolas de puño y letra, para entregarlas en la oficina de alumnos y se proceda al cierre definitivo de las mismas.

[1] *Autenticidad*: garantiza que el mensaje proviene de la persona que efectivamente lo está enviando.

Cambios en la estructura y en los procesos de aplicación

Como se indicó anteriormente, una vez concluído el período de evaluación los docentes, deben informar a la oficina de alumnos el listado de personas que aprobaron la cursada de la materia y, después de cada mesa de exámenes finales, deben entregar las actas con la información de aprobados, desaprobados y ausentes.

Todo ésto se hace en la actualidad a través de las operaciones brindadas por el módulo de docentes del sistema SIU-GUARANI, el cual es accedido a través de Internet, de la siguiente forma: el personal de la oficina de alumnos emite las actas correspondientes a través del módulo de gestión del sistema SIU-GUARANI, a partir de ese momento los docentes tienen acceso a ver las actas de cursada y/o final según corresponda. Luego, cada docente puede cargar el detalle de cada una de las actas de las materias que tiene a cargo, y una vez que considera que no va a realizar más modificaciones debe dirigirse a la oficina de alumnos, con el acta impresa y firmada de puño y letra, para que el personal de dicha oficina coteje el acta impresa con el acta cargada digitalmente, para asegurar que la información es verídica y se proceda al cierre definitivo del acta. De este modo, se hace efectiva la nota para los alumnos involucrados en la misma. El acta en papel es foliada en el libro de actas correspondiente.

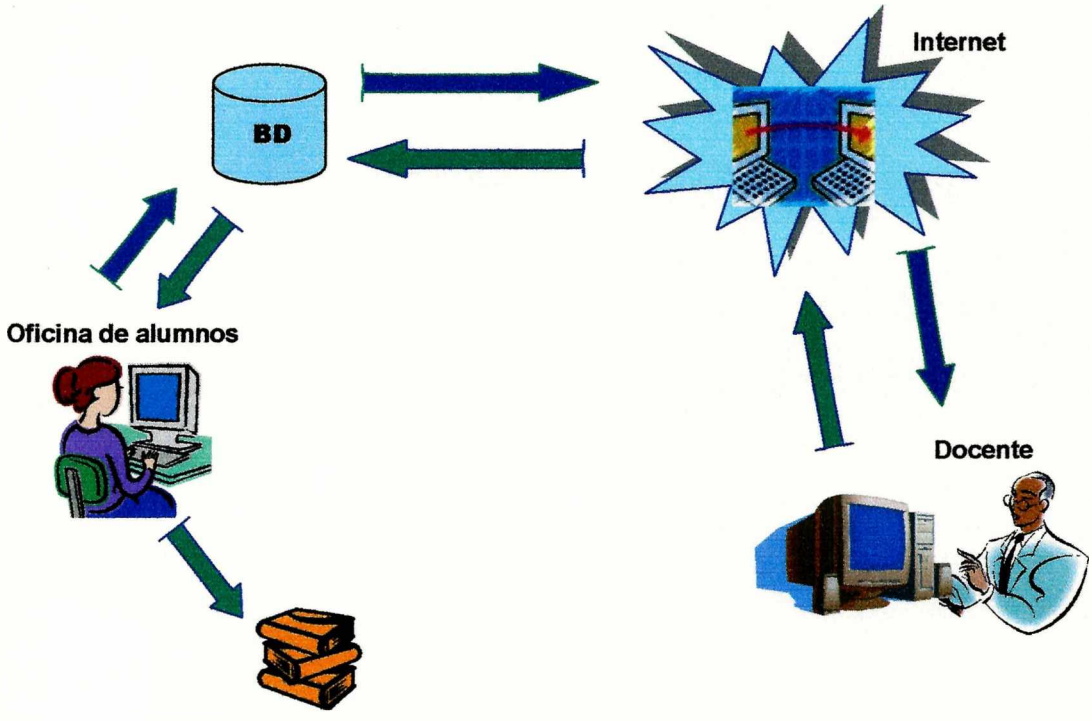


Con el fin de agilizar los trámites que el docente debe realizar ante la oficina de alumnos, se propone evitar la instancia de que el docente se presente personalmente ante dicha oficina con el acta firmada de puño y letra. Con tal objetivo, ampliaremos la funcionalidad del módulo de docentes del sistema SIU-GUARANI, para que los docentes firmen digitalmente las actas con la seguridad

necesaria para que todas las partes involucradas tengan la certeza que la información no ha sufrido alteraciones por agentes mal intencionados.

El hecho de firmar digitalmente un acta modifica el circuito administrativo actual, debido a que, la firma involucra el cierre definitivo del acta y ya no es necesario que lo realice el personal de la oficina de alumnos.

Otra forma de implementar la firma digital de actas, sería requerir que el acta también sea firmada por la oficina de alumnos antes de ser cerrada totalmente. Esta opción implicaría que se genere nuevamente el documento firmado digitalmente para luego compararlo con el que generó el docente. La comparación de documentos firmados digitalmente corresponde a los procesos de auditoria, por eso no se eligió implementar dicha opción.



En cuanto al cierre digital de un acta, surgen dos posibilidades, cierre parcial o cierre total. Cerrar parcialmente un acta implica que cada renglón del acta deba ser firmado digitalmente y almacenado en la base de datos documental[2], por lo tanto, por cada acta se tendrán n documentos firmados. El cierre total consisten en firmar el acta completo, por consiguiente se genera un documento firmado digitalmente por cada acta.

Actualmente, se realiza el cierre total de actas, debido a esto, se utilizará esta opción.

Los documentos firmados digitalmente son almacenados en una base de datos documental. Dicha base de datos tendrá la misma funcionalidad que los libros de actas actuales, es decir, permitirá buscar y agregar actas, pero NO modificar ni eliminar las que ya están almacenadas.

[2] *Base de datos Documental:* es la base de datos en la cual se van a guardar los documentos firmados digitalmente

Especificación

A continuación se explicará en detalle cual es la funcionalidad del módulo desarrollado e implementado.

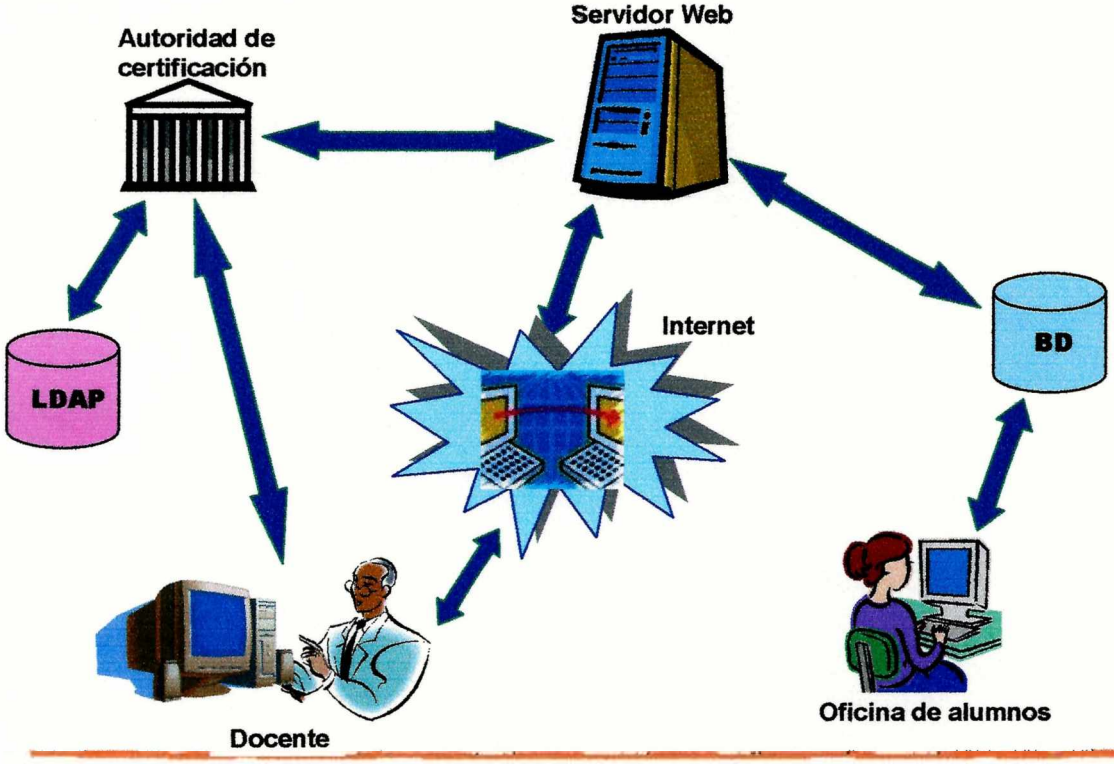
1. Los docentes deben solicitar su certificado a la autoridad de certificación desarrollada anteriormente.

2. La oficina de alumnos de la unidad académica emite el acta correspondiente a través del módulo de gestión del sistema SIU-GUARANI. En esta instancia no se requiere que se firme dicho documento, debido a que el acceso a este módulo sólo lo tienen personas autorizadas a tal efecto. Además está instalado sobre estrictas normas de seguridad implementadas por el equipo SIU-GUARANI La Plata [Anexo 1].

3. El docente, a través del módulo de docentes del sistema, puede ver las actas que fueron emitidas (un docente solo ve las actas de las materias que tiene a cargo). Como está previsto por el sistema, el docente puede cargar y/o modificar las notas de cada uno de los renglones del acta cuantas veces quiera. Una vez que el acta está cargada en forma definitiva, el docente deberá firmarla, lo cual produce el acta.

   Al firmar el acta, el docente está avalando su contenido y el mismo NO podrá ser modificado. Para modificarlo se deberá generar un acta rectificativa, tal cual como se maneja hasta el momento.
   Una vez firmada el acta, el documento digital generado (acta + firma) es almacenado en la base de datos documental.

Arquitectura del sistema



La solución implementada esta compuesta por las siguientes partes:

- **Autoridad de Certificación**: Entidad encargada de emitir los certificados para crear el contexto legal de la Firma Digital.
- **Base de datos documental**: en el cual se almacenan los documentos firmados digitalmente.
- **Interfaz de usuario**: basada en PHP, accesible desde cualquier navegador web.
- **Servidor de WEB:** el cual contendrá las páginas y scripts que implementarán la interfaz y relacionarán las partes del sistema. Dichos scripts serán los encargados de interactuar con la PKI y con el sistema SIU-Guarani.
- **Arquitectura del Sistema SIU-Guaraní.**

Desarrollo

Para el desarrollo de este modelo se debió estudiar, analizar y probar distintas herramientas. Los pasos realizados fueron:

- La primera tarea que se realizó fue la instalación y configuración de una Autoridad de Certificación, la cual publica los certificados que emite en un servidor LDAP, como hemos explicado en capítulos anteriores. Dicha Autoridad de Certificación será utilizada para que los usuarios de nuestra aplicación requieran sus certificados. Durante la fase de configuración se puso mayor atención en lo que se refiere a: requerimiento y emisión de certificados, políticas de revocación de certificados y procedimientos para la verificación de los datos.

- Dado que se va a agregar funcionalidad a un sistema ya implementado, se adoptó el software sobre el cual está desarrollado, en este caso PHP.

- Para poder implementar todos los aspectos relacionados con criptografía se buscaron librerías que proveyeran las operaciones necesarias.

- Como servidor web se utilizará IIS, ya que actualmente el sitio web del sistema SIU-Guaraní se encuentra corriendo sobre dicha herramienta.

- Para contar con un **repositorio de datos** se instaló una base de datos Informix. La selección de dicha base se centró en que el sistema SIU-Guaraní está desarrollado utilizando dicho motor, por lo tanto se eligió continuar con la misma línea de trabajo, para que sea fácilmente incorporado por las distintas implementaciones del mismo.

Funcionalidad obtenida

La incorporación de firma digital a la aplicación modificó y adicionó su funcionalidad.

Se modificó la funcionalidad debido a que, sin la incorporación de firma digital, una vez que los docentes entregaban las actas firmadas de puño y letra a la oficina de alumnos, se procedía al cierre definitivo de las mismas a través de las operaciones correspondientes del módulo de gestión. El uso de firma digital hace posible que el cierre del acta sea realizado por el mismo docente desde el módulo web a través de las operaciones de firma implementadas.

Se adicionó funcionalidad ya que, al utilizar firma digital, la aplicación administra también documentos firmados, debido a que genera un documento firmado por cada acta que se firme digitalmente y ese documento debe ser almacenado en una base de datos documental.

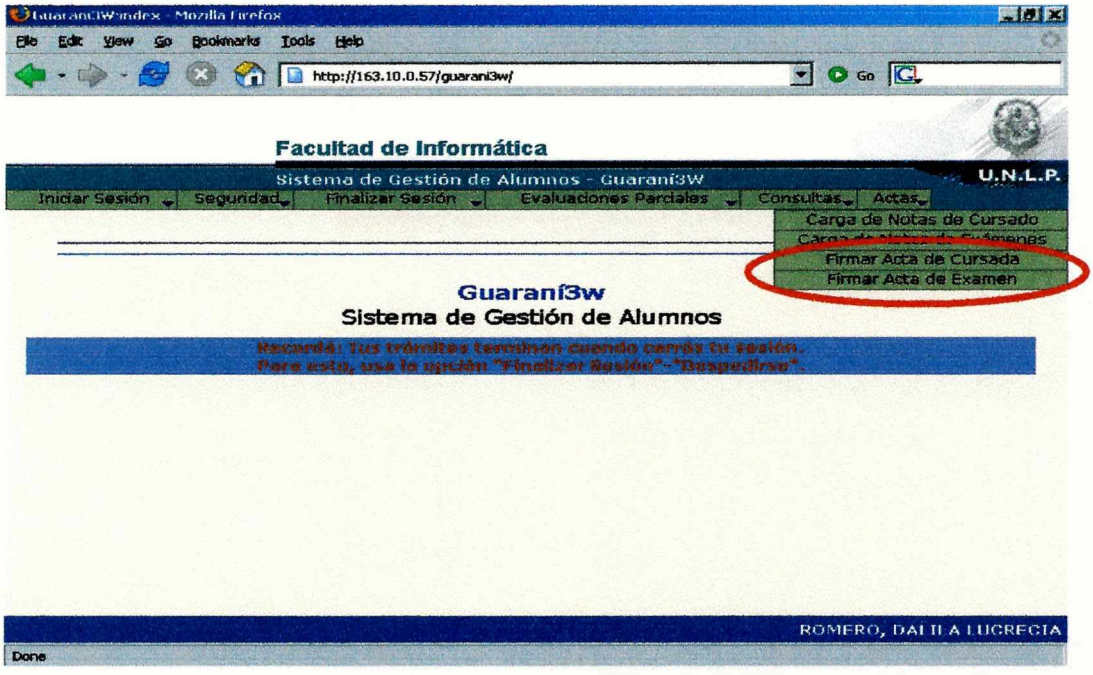
Operaciones incorporadas al sistema

Para implementar firma digital se agregaron dos operaciones al módulo de docentes del Sistema SIU-Guaraní:

1. Firmar Acta de Cursada, esta operación permite firmar el acta de cursada seleccionada.
2. Firmar Acta de Examen, esta operación permite firmar el acta de examen seleccionada.

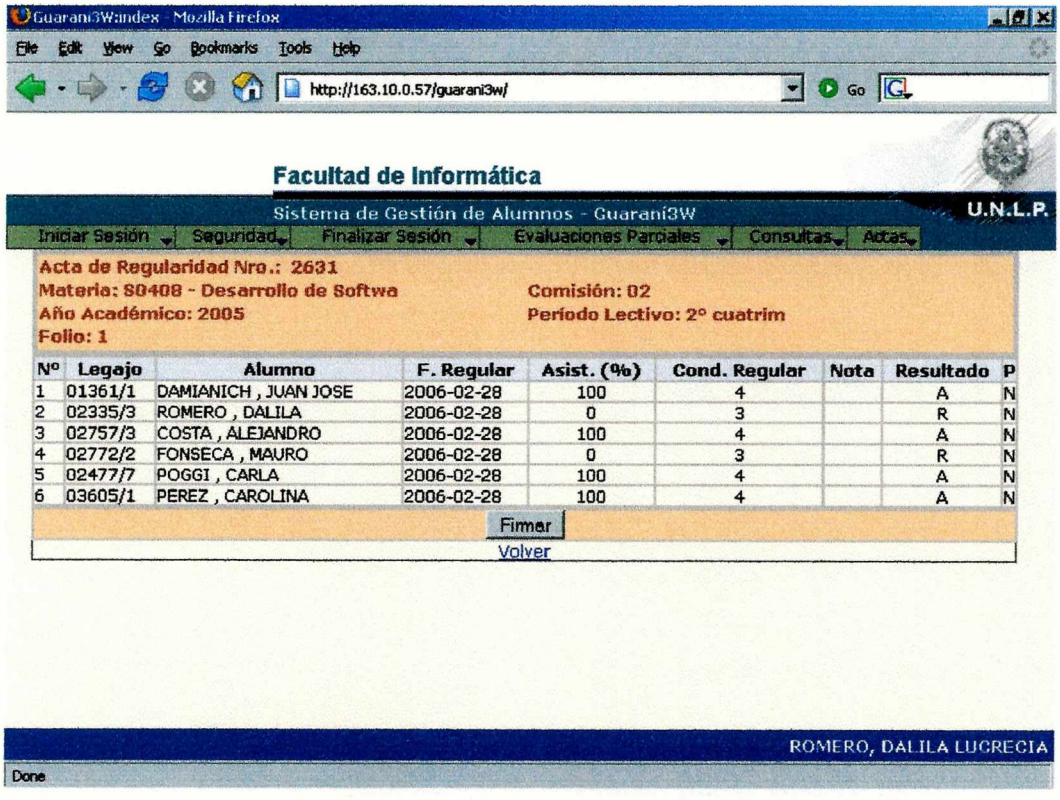
El docente accede a la aplicación de la forma habitual, indicando usuario y contraseña.

Luego, se accede a la opción Actas del menú, donde se pueden seleccionar las opciones para acceder a la operación para firmar el acta correspondiente, como se muestra a continuación:

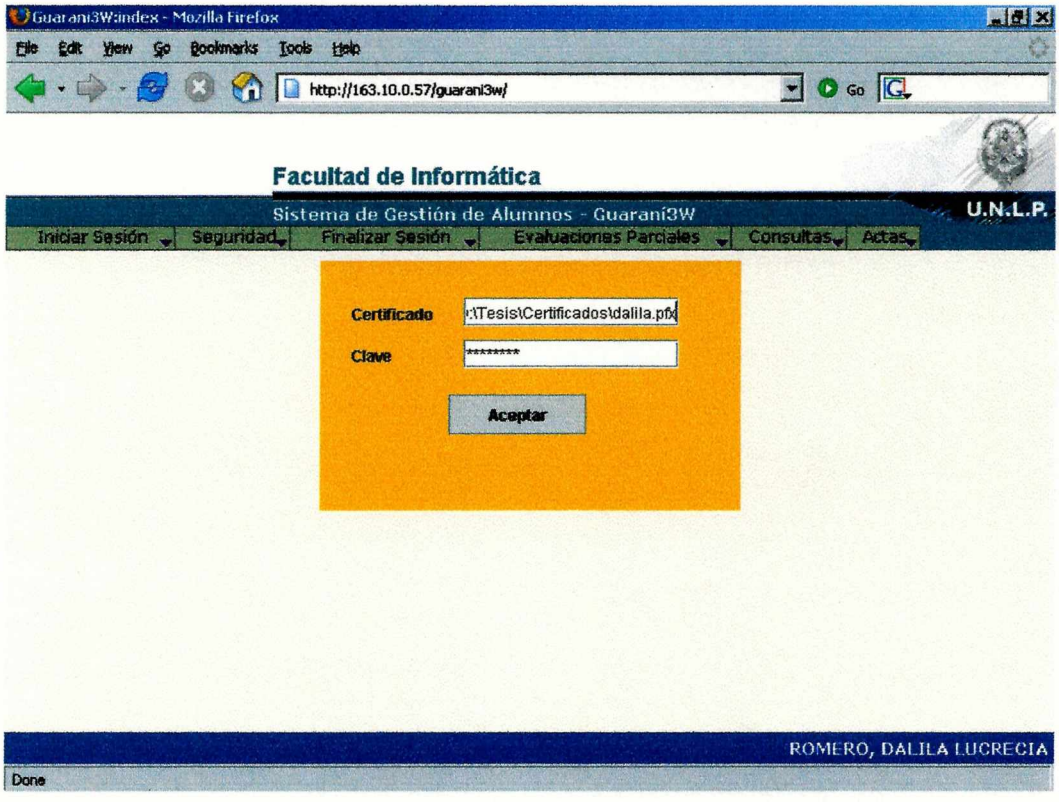


Luego de esto se desencadenan las siguientes acciones:

1. La página mostrará el detalle del acta según ha sido modificada por el docente, pero en esta instancia no le permite realizar cambios.



2. Una vez que el docente chequea el contenido del acta, para hacer efectiva la firma debe cliquear el botón "*Firmar*"; a partir de esta acción se producen los siguientes pasos:
   a. Se le solicitará el passphrase proporcionado por la CA.
   b. Se toma el certificado del docente desde donde éste lo indique.
   c. Se genera el documento firmado con dicho certificado y la clave privada.
   d. El documento es enviado al servidor, donde se valida la firma (LDAP). Si todo es correcto, el documento digital es almacenado.



3. Por último, se informa al docente el resultado de la operación.

Implementación

La firma de cada operación es realizada en el cliente a través de un applet, esto se debe a que el passphrase y la clave privada del certificado no debe ser transmitido a través de la red, independientemente que sea sobre un canal seguro (https). El applet, se encarga de generar el documento con el acta y de firmar dicho documento. Luego, la firma junto con el documento es enviada al servidor para realizar su validación.

Finalmente en el servidor, el php receptor se encarga de verificar la firma y dependiendo del resultado de dicha verificación, las acciones correspondientes son registradas en la base de datos y se informa al usuario el resultado de la misma.

Para implementar lo explicado anteriormente se desarrolló un applet llamado appFirmaActa. Este applet, implementa el método *firmar*, el cual utiliza las operaciones brindadas por el paquete iaik[3] para generar la firma correspondiente. Es invocado a través de la página *819firmarActaCurs.php*.

El método firmar del applet recibe tres parámetros, la dirección en donde se encuentra el archivo a firmar, la dirección en donde esta el certificado en formato pkcs12[4] y el passphrase para desencriptar la clave del certificado. En su implementación, el método firmar utiliza las operaciones brindadas por el paquete iaik para crear un objeto que implementa un algoritmo de firma, en nuestro caso SHA1/RSA[5], luego se obtiene la clave privada del certificado y se aplica el algoritmo de firma con esa clave privada; por último se invoca la página validarFirma.php, que se encarga de realizar la validación del lado del servidor.

Fue implementada una clase *VerificarFirma*, que es utilizada en el servidor para realizar la verificación de la firma, la misma se conecta con LDAP para realizar dicha verificación. Esta clase se encarga de certificar que efectivamente el documento que llegó al servidor no haya sido alterado durante su transferencia desde el cliente. Si la verificación fue exitosa, el acta y la firma son almacenadas en la base de datos.

Qué usarían los docentes para guardar la firma (clave privada)

Es importante tener presente que la clave privada debe ser guardada en forma segura para evitar la pérdida u olvido de la misma, jamás debe ser guardada en formato de texto plano.

Los docentes del sistema hacen uso de su clave privada a través de certificados digitales, por consiguiente éste no conoce cual es su clave privada, pero lo que si conoce es el passphrase para obtenerla de su certificado. Como principal medida de seguridad, es muy importante que ese passphrase esté bien resguardado y que no sea de fácil deducción. El mecanismo más simple sería almacenar dicho passphrase en una computadora que no sea accedida por nadie o en un medio removible que lleve consigo y este protegido para su acceso. Otro mecanismo, quizá menos aplicable, sería utilizar un software para encriptar dicho passphrase y de ese modo guardarlo en donde quede cómodo. Otro mecanismo simple, es armar el passphrase con por ejemplo, las iniciales de una frase que sea fácil de recordar y no escribirla en ningún lado.

[3] Iaik: Institute for Applied Information Processing and Communication
[4] Pkcs12:
[5] MD5/RSA:

Código Fuente de la implementación

A continuación se presenta todo el código fuente de lo desarrollado.

```
package firmadigital;

import java.applet.*;
import java.awt.*;
import java.awt.event.*;
import java.io.*;
import java.net.*;
import iaik.x509.X509Certificate;
import java.security.SignatureException;
import java.security.NoSuchAlgorithmException;
import java.security.NoSuchProviderException;
import java.security.InvalidKeyException;
import java.security.Security;
import iaik.security.provider.IAIK;
import java.security.PrivateKey;
import java.security.Signature;
import iaik.pkcs.PKCSException;
import iaik.pkcs.pkcs12.PKCS12;
import iaik.pkcs.pkcs12.KeyBag;

import netscape.security.*;
import javax.swing.*;
import com.borland.jbcl.layout.*;

public class AppFirmaActa extends Applet {
  boolean isStandalone = false;

  private String pathArchi;
  public String archiFirmado="Todavia no soy el archi firmado";

  XYLayout xYLayout1 = new XYLayout();
  JTextField jTextFCert = new JTextField();
  JPasswordField jPassword = new JPasswordField();
  JTextPane jTextPane1 = new JTextPane();
  JTextPane jTextPane2 = new JTextPane();
  JButton jButtonAceptar = new JButton();

  /**Get a parameter value*/
  /*public String getParameter(String key, String def) {
    return isStandalone ? System.getProperty(key, def) :
      (getParameter(key) != null ? getParameter(key) : def);
  }
*/
  /**Construct the applet*/
  public AppFirmaActa() {

    String x;
    try {
      jbInit();
   //   x=this.firmar("c:\\t.txt", "c:\\dalila.pfx", "anajulia");
      // x=this.firmar();
    }
    catch(Exception e) {
      e.printStackTrace();
    }
```

```
}

/**Initialize the applet*/
public void init() {
String param;
  try {
    /*pathArchi = this.getParameter("pathArchi");*/
    param = getParameter("pathArchi");
    /*param = getParameter("Target");*/
            if (param != null)
                    pathArchi = param;
            else
             pathArchi="pepe";
    jbInit();
  }
  catch(Exception e) {
    e.printStackTrace();
  }
}
/**Component initialization*/
private void jbInit() throws Exception {
  jTextFCert.setToolTipText("");
  this.setLayout(xYLayout1);
  jTextPane1.setText("Certificado");
  jTextPane2.setText("Clave");
  xYLayout1.setWidth(335);
  xYLayout1.setHeight(168);
  jButtonAceptar.setText("Aceptar");
  jButtonAceptar.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(ActionEvent e) {
      jButtonAceptar_actionPerformed(e);
    }
  });
  this.setBackground(new Color(152, 198, 130));
  this.setForeground(Color.white);
  this.add(jPassword,     new XYConstraints(160, 60, 113, 22));
  this.add(jTextFCert, new XYConstraints(159, 28, 113, 22));
  this.add(jTextPane1,  new XYConstraints(21, 29, 123, 22));
  this.add(jTextPane2, new XYConstraints(21, 61, 123, 22));
  this.add(jButtonAceptar, new XYConstraints(80, 105, 149, 36));
}
/**Start the applet*/
public void start() {
}
/**Stop the applet*/
public void stop() {
}
/**Destroy the applet*/
public void destroy() {
}
/**Get Applet information*/
public String getAppletInfo() {
  return "Applet Information";
}

public String[][] getParameterInfo()
   {
            String[][] info =
```

```
            {
                    /*{ "pathArchi", "String", "Target Frame" },*/
                  { "pathArchi", "String", "Camino del archivo"},
            };
            return info;
    }

  //public String firmar(String pathArchi,String pathPrivKey,String realPasswd ) {
  public String firmar(String pathPrivKey,String realPasswd ) {
 //pathArchi: es la direccion en donde se encuentra el archivo a firmar
 //pathPrivKey: es la direccion en donde esta el certificado en pkcs12
 //realPasswd: es el passfrase para desencriptar la clave del certificado

 PrivilegeManager.enablePrivilege("UniversalFileAccess");

 java.security.Security.addProvider(new iaik.security.provider.IAIK());


 PKCS12 archi = null;  //certificado

 Signature rsa = null; //algoritmo de firma

 byte[] realSig = null;  //firma


  try {
    archi = new PKCS12( new FileInputStream(pathPrivKey));
    }

  catch (PKCSException xxx) {}
  catch (FileNotFoundException fex) {}
  catch (IOException iox) {}

  try {
    // creo la firma
    rsa = Signature.getInstance("MD5/RSA", "IAIK");
    }
  catch (NoSuchAlgorithmException aaa) {}
  catch (NoSuchProviderException bbb) {}

    // Desencripto la clave privada???
    char[] passwd = realPasswd.toCharArray();

  try {
    archi.decrypt(passwd);
    }
  catch (PKCSException ppp) {}

    // Obtengo la clave privada
    iaik.pkcs.pkcs12.KeyBag clave=archi.getKeyBag();

  java.security.PrivateKey privKey=clave.getPrivateKey();

  try {
   rsa.initSign(privKey);
    }
  catch (InvalidKeyException ccc) {}
  catch (java.lang.NullPointerException c) {}
```

```
        // cargo el contenido del archivo que quiero firmar
      FileInputStream fis = null;

    PrivilegeManager.enablePrivilege("UniversalFileRead");

      try {

        fis = new FileInputStream (pathArchi);
      }
      catch (FileNotFoundException ffe){}

        BufferedInputStream bufin = new BufferedInputStream(fis);
        byte[] buffer = new byte[1024];
        int len;

      try {
        while (bufin.available() != 0) {
            len = bufin.read(buffer);
            rsa.update(buffer, 0, len);
        };
        bufin.close();

        realSig=rsa.sign();

        }
      catch (SignatureException xxx) {}
      catch (IOException iox) {}


      String lafirma=new String(realSig);
      //archiFirmado=new String(realSig);
  //Comunicacion del Applet con el Servlet para enviarle la firma **************

      URL servletURL=null;
      URL phpUrl=new URL("http://163.10.0.57/guarani3w/validarFirma.php");

      URLConnection phpConnection = null;
      phpConnection = phpUrl.openConnection();

      String idfirma=null;

    try {
        phpConnection.setDoOutput(true);
        phpConnection.setAllowUserInteraction(false);
        phpConnection.setDoInput(true);
        phpConnection.setUseCaches(false);
        phpConnection.setRequestProperty("Content-type","application/binary");
        DataOutputStream dos = new
  DataOutputStream(phpConnection.getOutputStream());
       dos.writeBytes(lafirma);
       int tamanio = dos.size();
       dos.close();
       phpConnection.connect();

       // Lee la respuesta de la pagina
        InputStream in = phpConnection.getInputStream();
        StringBuffer respuesta = new StringBuffer();
        int chr;
        while ((chr=in.read())!=-1) {
```

```
            respuesta.append((char) chr);
          }
          in.close();
          idfirma=respuesta.toString();

        } catch (IOException e) { }

       return idfirma;

    }


     void jButtonAceptar_actionPerformed(ActionEvent e) {

      String pathPrivKey= jTextFCert.getText();
      String realPasswd= String.valueOf(jPassword.getPassword()) ;
      this.firmar(pathPrivKey, realPasswd);
     }

    }


    import java.security.*;
    import java.util.*;
    import iaik.security.rsa.*;
    import iaik.security.provider.IAIK;
    import iaik.pkcs.pkcs12.*;
    import iaik.pkcs.*;

    import iaik.x509.X509Certificate;
    import java.security.cert.CertificateException;
    import java.io.*;
    import java.lang.*;


    public class VerificarFirmaStr {


    public boolean verificar (String infoAfirmar, String clicert, String idfirma) throws
    Exception
     {

        String firma = null;
        boolean verifies=false;

         IAIK.addAsProvider(true);

    try {

          byte[] realSig=null;

    // tomo la firma del archivo correspondiente
        String dirFirma = "//usr//local//firmas//";
        String archivoFirma = "firma-"+idfirma.substring(0,idfirma.length()-1)+".enc";

        FileInputStream fis3 = new FileInputStream (dirFirma+archivoFirma);
        BufferedInputStream bufin3 = new BufferedInputStream(fis3);
        byte[] buffer3 = new byte[128];
```

```
        int len3 = bufin3.read(buffer3);
        String str_buffer3=new String(buffer3);

          realSig = str_buffer3.getBytes();
        X509Certificate cert = null;

    try {
      cert = new X509Certificate(clicert.getBytes());
      }
    catch (CertificateException vvv) {}


      // obtener la clave publica
      PublicKey pub = cert.getPublicKey();

      Signature sig = null;

    try {
      sig = Signature.getInstance("MD5/RSA","IAIK");
      }
     catch (NoSuchAlgorithmException eee) {}
     catch (NoSuchProviderException iii) {}

    try {
      sig.initVerify(pub);
      }
    catch (InvalidKeyException yyy) {}


    // verifico la firma

    try  {
       byte[] buffer = infoAfirmar.getBytes();
       int len = buffer.length;
       sig.update(buffer, 0, len);

       verifies = sig.verify(buffer3);
       }
     catch (SignatureException ppp) { }

    }catch (java.lang.NullPointerException npe) {}

     return verifies;
  }


  public boolean verificarDoc (String pathDoc, String clicert, String firmaDoc) throws
  Exception
   {

      String firma = null;
      boolean verifies=false;

      //IAIK es agregado como proveedor

      IAIK.addAsProvider(true);

  // tomo la firma del arcfirmaDoc.enc
```

```
        String dirFirma="//usr//local//firmas//";
        FileInputStream fis3 = new FileInputStream (dirFirma+firmaDoc);
        BufferedInputStream bufin3 = new BufferedInputStream(fis3);
        byte[] buffer3 = new byte[128];

        int len3 = bufin3.read(buffer3);
        String str_buffer3=new String(buffer3);


    //*************************************************************

       X509Certificate cert = null;

     try {
       cert = new X509Certificate(clicert.getBytes());
        }
     catch (CertificateException vvv) {}


       // obtener la clave publica
       PublicKey pub = cert.getPublicKey();

       Signature sig = null;

     try {
      sig = Signature.getInstance("MD5/RSA","IAIK");
       }
     catch (NoSuchAlgorithmException eee) {}
     catch (NoSuchProviderException iii) {}

     try {
      sig.initVerify(pub);
       }
     catch (InvalidKeyException yyy) {}


     // ***************** verifico la firma

       FileInputStream fis = new FileInputStream (pathDoc);
       BufferedInputStream bufin = new BufferedInputStream(fis);
       byte[] buffer = new byte[1024];
       int len = 0;
     try {
       while (bufin.available() != 0) {
           len = bufin.read(buffer);
           sig.update(buffer, 0, len);
        };
       bufin.close();

       verifies = sig.verify(buffer3);
       }
     catch (SignatureException ppp) { }

      return verifies;
     }

  }
```

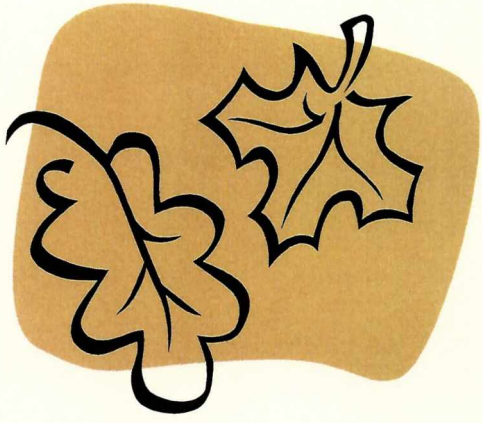
```
package firmadigital;

import java.util.*;
import java.io.*;
import javax.naming.*;
import javax.naming.ldap.*;
import javax.naming.directory.*;

public class accesoLdap {

final static String ldapServerName = "163.10.0.58:389";
final static String rootdn = "cn=root,o=Linti,c=AR";
final static String rootpass = "syperca";
final static String rootContext =  "o=Linti,c=AR";

InitialDirContext ctx = null;


 public void depura(String cadena) // codigo para unificar salidas
 {
    System.out.println(cadena);
 }

 public static void main(String[] args) // punto de entrada a la aplicacion
 {
    accesoLdap instancia = new accesoLdap();
    instancia.ejecuta(); // evitamos instranciacion estatica de los metodos
 }


 public void ejecuta()
 {
    String target = "";
    Properties env = System.getProperties();
    env.put(Context.INITIAL_CONTEXT_FACTORY, "com.sun.jndi.ldap.LdapCtxFactory");
    //env.put( Context.PROVIDER_URL, "http://" + ldapServerName);
    env.put(Context.PROVIDER_URL, "ldap://163.10.0.58:389/o=Linti,c=AR");
    env.put( Context.SECURITY_PRINCIPAL, rootdn );
    env.put( Context.SECURITY_CREDENTIALS, rootpass );

    //env.put(Context.PROVIDER_URL,
"ldap://163.10.0.58:389/ou=front,o=desarrollo,o=casa");
  try
  {
    ctx = new InitialDirContext(env);
    depura ("El DN es: " + ctx.getNameInNamespace());
    muestraLista(target, ctx.list(target));

    ctx.close();
  }
  catch (Exception e)
  {
    depura("Excepcion EN BUCLE PRINCIPAL");
    e.printStackTrace();
  }
 }


 // para el DN que se pone en PROVIDER_URL, recorre todos los elementos
```

```
void muestraLista(String msg, NamingEnumeration nl)
{
    System.out.println("Sacamos lista de elementos para: " + msg);
    if (nl == null)
    {
      System.out.println("No hay Elementos en la lista");
    }
    else
    {
      try
      {
            // recorrer la enumeracion
            while (nl.hasMore())
            {
            Object objeto = nl.next();
            NameClassPair parNombre = null;
            depura("Detalle del Objeto" + objeto.getClass().getName());

            // nos aseguramos que es objeto del tipo adecuado
            if (objeto instanceof javax.naming.NameClassPair)
            {
                  // depura ("Es un javax.naming.NameClassPair");
                  parNombre = (NameClassPair) objeto;
            }
            else
            {
                  depura("No es un nombre");
                  return;
            }

            // Cojer el nombre
            String nombre = parNombre.getName();
            depura("El nombre recogido es " + nombre);

            // listaAtributos(ctx,nombre);
            }
      }
      catch (NamingException e)
      {
            e.printStackTrace();
      }
    }
}


void listaAtributos (DirContext localContext, String cadena)
{
    try
    {
        // se puede mejorar pasandole un array con el nombre de los atributos a recoger
        Attributes attr = localContext.getAttributes(cadena);

        // recuperamos una enumeracion con todos los atributos
        NamingEnumeration nl = attr.getAll();

        if (nl == null)
        {
            depura("lista de atributos nula");
            return;
```

```
            }

            while (nl.hasMore())
            {
                Object objeto = nl.next(); // recorremos todos los tributos

                if (objeto instanceof Attribute)
                {
                    // cojemos un atributo especifico
                    Attribute internalAttr = (Attribute)objeto;
                    depura("\tAtributo = " + objeto.toString());
                }
            }
        }
        catch (NamingException e)
        {
            e.printStackTrace();
        }
    }

     public accesoLdap() {
     }


    }
```

Módulo 4
Conclusiones

Conclusiones



Introducción

Durante el desarrollo de este trabajo de tesis tuvimos ciertas dificultades, las cuales llevaron a demorar, entre otras cosas, la conclusión del mismo. Dichas dificultades, fueron en cierta medida, marcando el camino a seguir.

Debimos familiarizarnos con varios lenguajes de programación, realizar la configuración e instalación de distintas herramientas y pruebas de integración entre ellas, dada la heterogeneidad de los componentes utilizados en este trabajo. Dicha integración hace posible brindar: seguridad en la transmisión de datos, autenticidad de cada usuario y organización consistente por el uso de servicios de directorios.

Es importante destacar que la realización de este trabajo de tesis nos llevó a descubrir que utilizar firma digital no significa solamente firmar un documento, ya que si ese documento firmado no es usado digitalmente la firma digital carece de sentido, convirtiéndose sólo en un adorno.

Ventajas y desventajas de la incorporación de firma digital al sistema SIU-Guaraní

Si hacemos una evaluación de la incorporación de firma digital al sistema SIU-GUARANI desde la perspectiva de los usuarios, debemos tener en cuenta que este sistema tiene tres tipos de usuarios, a saber, personal de la oficina de alumnos, docentes y alumnos.

Con respecto a los alumnos, la incorporación de firma digital al sistema no modifica en nada la funcionalidad con la que ya cuentan, pero, reduce el tiempo entre que el docente vuelca la nota en el acta y la misma es vista en la historia académica de los alumnos.

Con respecto al personal de la oficina de alumnos, la incorporación de firma digital al sistema hace que se modifique el circuito administrativo correspondiente para el cierre de actas de cursada y final por lo explicado anteriormente. Además, agrega la función de auditar que la firma digital sea válida para los elementos de la base de notas cuando se pide un diploma.

Con respecto a los docentes, la incorporación de firma digital al sistema hace que los docentes sientan cuenten con mas mecanismos para agilizar los tramites administrativos que deben realizar para el cierre de las actas que tienen a cargo.

Si hacemos una evaluación de la incorporación de firma digital al sistema SIU-GUARANI desde la perspectiva nuestra perspectiva, podemos hallar las siguientes ventajas y desventajas:

Ventajas

- Los docentes no necesitan concurrir físicamente hasta la oficina de alumnos para que las actas a su cargo sean cerradas definitivamente.
- Firmar las actas digitalmente facilita futuras funciones de auditoria.
- Ya no es imprescindible mantener un archivo de las actas en papel.

Desventajas

- Los docentes deberán gestionar su certificado digital para poder firmar digitalmente sus actas.

- Se debe permitir que los docentes que no quieran utilizar la firma digital de actas puedan continuar gestionando sus actas como antes.

Líneas futuras de trabajo

Se presenta como líneas futuras de trabajo resolver el problema de que un documento digital sea firmado por más de un profesor, además de crear los mecanismos necesarios para realizar la auditoría de la base de datos digital.

En lo referente a que un documento sea firmado por más de un profesor, habría que incorporar el concepto de firmas pendientes, con lo cual el proceso de firma no estaría concluido hasta que todos los profesores lo hayan firmado. Esto significa que un documento firmado va a tener asociado diferentes estados y más de una firma, según la cantidad de profesores que participen en el acta.

Para poder implementar esto debemos tener presente que el cierre del acta no será consecuencia directa de la firma, sino que se producirá en el momento en que el último profesor firme el acta. De este modo, adquiere más complejidad el proceso de firmar un acta, ya debe verificar: que la firma sea válida, como hasta el momento, y si falta la firma de algún profesor para generar el cierre del acta.

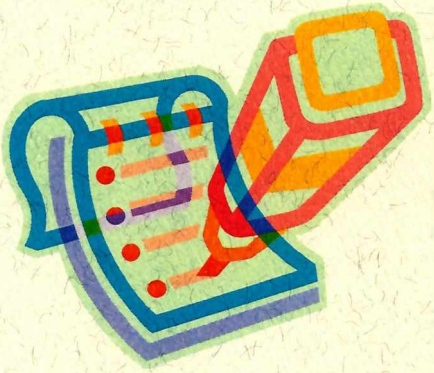
Con respecto a la auditoría, se deben implementar los mecanismos necesarios para auditar la base de datos documental, en la cual se almacenan los documentos firmados digitalmente.

Los procesos de auditoría deberán dar certeza de que el documento no ha sido alterado luego de su firma, que ha sido firmado por quien debió firmarlo y que la firma era válida al momento de ser utilizada. Estos procesos cobran vital importancia al momento de egresar a un alumno.

Bibliografía

- Trabajo de grado "Utilizando firma digital", desarrollado por Paula Venosa y Verónica Fredes
- Trabajo de grado "Implementando Firma Digital con J2EE" de Alejandro Falcone y Maria Clemens
- www.openldap.org
- www.redbooks.ibm.com
- www.opengroup.org
- www.openssl.org
- www.openldap.org
- www.openca.org
- httpd.apache.org
- www.siu.edu.ar
- www.programacion.com/java
- java.sun.com
- www.sun.com
- www.webestilo.com/php
- www.desarrolloweb.com/php
- www.php.net
- www.monografias.com/trabajos/tesisgrado/tesisgrado.shtml
- www.uv.es/~sto/cursos/seguridad.java/html/sjava-32.html#tblModSeg
- www.programacion.com/java/tutorial/security1dot2/1
- www.kriptopolis.org
- "Como se hace una tesis" Umberto Eco

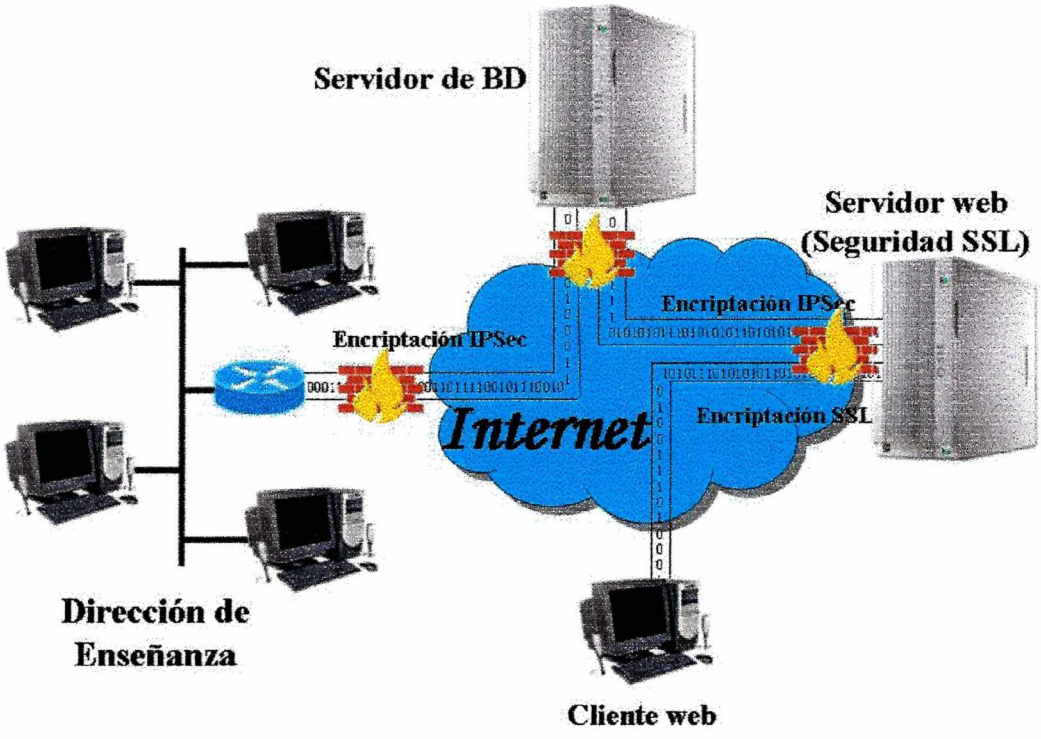
Apendice



Seguridad en Guaraní

A continuación se detallan las normas de seguridad implementadas para minimizar los riesgos[1] de intrusiones y demás riesgos concernientes a la seguridad de un servicio.

Topología



Los hitos claves para prevenir ataques se describen a continuación detallando las diferentes conexiones que se pueden llevar a cabo con una topología como la graficada y el servicio Guaraní (Gestión, Autogestión y Guaraní3w):

[1] Algunos de los riesgos en el caso de una intrusión pueden ser: que información de vital importancia sea vista por terceras partes no confiables, que la información sea alterada por terceras partes no confiables, que un intruso se haga pasar por un usuario legítimo, entre otros.

Políticas de seguridad implementadas en relación al acceso a los servicios:

Comunicación Servidor Web-Servidor de BD

El servidor de BD está configurado de manera tal de sólo permitir accesos de ciertos clientes. Dichos clientes son el servidor web, y los clientes de Gestión y Autogestión ubicados en la Dirección de Enseñanza de la facultad. Cualquier otra PC que desee conectarse con el servidor de BD será bloqueada.

Además, la comunicación entre el Servidor WEB y el Servidor de BD se establece a través de una sesión IPSec[2], donde ambos coinciden en armar un túnel. Esto significa que los datos viajan a través de Internet de manera encriptada y por lo tanto ilegible para cualquier extraño.

Cabe aclarar que el servidor de web sólo permite acceso a determinados puertos. Estos puertos son los que dan servicio a los clientes web, y aquellos que mantienen la conexión con el servidor de BD.

Comunicación Dirección de Enseñanza-Servidor de BD

Al igual que la comunicación que el servidor de BD establece con el servidor web, la conexión del servidor de BD con cada uno de los clientes se establecerá sólo si previamente se establece una sesión de IPSec entre ambos, es decir que entre ellos la información también viaja de manera encriptada e ilegible. Además el servidor de BD sólo permitirá la conexión con clientes preestablecidos, evitando de esta manera que un extraño pueda iniciar una conexión desde la Dirección de Enseñanza.

Además de esto, el router que permite la entrada/salida desde y hacia la red de la Dirección de Enseñanza, presenta una serie de filtros que permiten tráfico de entrada de paquetes sólo provenientes desde el servidor de BD bloqueando todo tipo de paquete proveniente de cualquier otro origen así como también permiten tráfico de salida sólo de los clientes preestablecidos. De esta manera se evita cualquier tipo de intrusión a la red de la Dirección de Enseñanza.

Comunicación Servidor Web- clientes web

Esta comunicación se establece en forma segura mediante el proceso de encriptación provisto por SSL[3] (Secure Socket Layer). Cada vez que un cliente se conecta a través de un Navegador al servidor web, este le comunica que está por establecer una conexión segura con dicho protocolo y le presenta la posibilidad de instalar el certificado (propio de la U.N.L.P) de manera tal que el cliente no tenga que aceptar la conexión cada vez que esta se establezca

[2] **IPSec** =*IP Security*. Conjunto de protocolos desarrollados por la **IETF** para soportar el intercambio seguro de paquetes a nivel IP [http://www.ietf.org/html.charters/ipsec-charter.html]

[3] **SSL** = *Secure Socket Layer*. Protocolo de seguridad desarrollado por Netscape que provee encripción de datos, autenticación del Servidor (garantizar autenticidad significa garantizar que la entidad sea quien dice ser), integridad de los mensajes (significa asegurar que los mismos no serán alterados en tránsito), y autenticación opcional de los clientes para conexiones TCP/IP.

Políticas de seguridad implementadas en los puestos de trabajo:

Las PCs que actúan como clientes de Gestión y Autogestión (ubicadas en la Dirección de Enseñanza) poseen *Microsoft Windows 2000 Professional* o bien, *Microsoft Windows XP*. Cualquiera de estos dos sistemas operativos de escritorio cuenta con varias herramientas que ayudan a lograr configuraciones más seguras que otros sistemas operativos de escritorio. Algunos aspectos con respecto a la seguridad de los clientes:

- *Discos con formato NTFS:* esto permite colocar permisos de acceso a nivel de archivo, especificando los usuarios que tendrán o no acceso a los mismos. Una configuración recomendada es dar permiso de sólo lectura tanto a los archivos del sistema Guaraní, como a los archivos del cliente *Informix* a los usuarios de la Dirección de Enseñanza. De esta manera se evita el borrado, ya sea accidental o intencional de archivos fundamentales para el servicio.

- *Instalación de patchs de seguridad:* Asegurarse de obtener las últimas actualizaciones de seguridad de los productos utilizados.

- *Creación de las políticas de seguridad IP (túnel con el servidor de BD):* estas políticas son las que permiten tanto al servidor aceptar las conexiones de los clientes y como a los clientes aceptar las respuestas del servidor de manera segura (encriptada). Estas políticas también evitan que los clientes puedan conectarse con otros equipos en forma innecesaria evitando posibles intrusiones en los clientes por un vecino infectado.

- *Desinstalación de aplicaciones innecesarias:* estas PCs sólo deberían ser usadas para uso de los módulos de Gestión y/ó Autogestión del sistema Guaraní. Para ello no es necesaria ninguna aplicación que pueda amenazar la seguridad de las mismas. Es pocas palabras, estas PCs **DEBEN** ser puestos dedicados.

- *Creación de perfiles de usuarios:* En cada una de las PCs se deberán crear perfiles personalizados para los usuarios que hagan uso de las mismas. Dichos perfiles deberán tener la menor cantidad de privilegios posibles.

- *Creación de políticas de seguridad mediante "Directivas de Seguridad":* Esta herramienta permite una importante variedad de configuraciones. Una buena práctica es denegar a los usuarios de la Dirección de Enseñanza cualquier tipo de acceso a las unidades de las PCs (discos, CD-ROM, disquetera) y permitiendo sólo la ejecución de las aplicaciones que le son de utilidad para llevar a cabo las tareas correspondientes. De este modo evitamos cualquier tipo de cambios en la configuración de los clientes (ya sea accidental o intencional).