OpenLDAP Baseline Security Analyzer

Parte 1: Ataques y amenazas a un servidor LDAP

Parte 2: Criterios de seguridad

Versión: 0.03

Autor: Inmaculada Bravo (inma@usal.es)

Fecha: 11/Noviembre/09

Resumen del proyecto

El propósito de este proyecto es elaborar un listado de criterios que se deberían tener en cuenta para realizar el plan inicial de securización del directorio.

Se documentan brevemente los principales ataques y las amenazas a las cuales nos enfrentamos los administradores y, a continuación, se listan los criterios de seguridad que se deben considerar, con una explicación de cada uno, su valoración, como implementarlo y con la etiqueta [OJO] se marcan los errores mas comunes que hay que tratar de evitar.

Los criterios que se plantean se han recogido de diferentes fuentes que puedes consultar en la sección de referencias y de la experiencia de administradores LDAP de las instituciones afiliadas a RedIRIS coordinados a través de la lista IRIS-LDAP.

Se utilizará una herramienta gráfica para interrogar al administrador por el cumplimiento de cada cada uno de los criterios de seguridad propuestos.

Esta herramienta es OCIL Interpreter, (escrito en java ,software libre), para lo cual, estos criterios se codifican en XML respetando el Schema OCIL.

Se diseña un sitio web para dar visibilidad al proyecto y que sea accesible por el mayor número de usuarios posible.

El objetivo será doble: provocar en los administradores actuales un reanálisis de la securización del servicio de directorio y, además, que pueda servir de guía para los administradores que se enfrenten a esta tarea por primera vez.

Índice de contenido

Justificación

El ciclo de vida del servicio de directorio en nuestras organizaciones suele ser, como he comprobado por mi propia experiencia y de otros colegas, como sigue:

El directorio suele nacer como un servicio secundario asociado al correo electrónico, un lugar donde depositar las credenciales de los usuarios.

Al ser un servicio que no se accede desde el exterior, solo desde la protegida red interna y desde muy pocas máquinas, las amenazas a las que se enfrenta en un principio parecen controladas por lo que su securización no es vital.

Con el tiempo otros equipos de los propios servicios informáticos comienzan a consultarlo para hacer búsquedas; se crean nuevos servicios que requieren autenticación y necesitan realizar búsquedas de algún otro dato o atributo como categoría, DNI.. ..etc, para acabar convirtiéndose en el punto de entrada de la mayor parte de las aplicaciones tales como accesos a la red inalámbrica vpn, docencia virtual, acceso a portales institucionales , incluso para aplicaciones tan sensibles como modificar las actas de notas de alumnos, caso de nuestra institución.

Es en este momento cuando parece realmente necesario replantearse profundamente la seguridad, con la complejidad añadida de reconducir todos los accesos actuales a través de estos nuevos criterios de seguridad.

En otros servicios que se ofrecen a la comunidad universitaria ya existen unas recomendaciones precisas sobre la seguridad, como en el servicio de correo electrónico y otros servicios de comunicaciones.

Pero al iniciar el proceso de securización del directorio solo contamos con unos escuetos manuales y el sentido común de cada administrador.

Herramientas parecidas

RACEv2

Red de Calidad de Correo Electrónico Criterios de calidad para la creación de una red de confianza . http://www.rediris.es/race/

Los criterios de calidad expuestos en el documento RACEv2, y las recomendaciones asociadas a los mismos, han sido extraídas de la experiencia adquirida a lo largo del tiempo por el conjunto de responsables de los Servicios de Correo Electrónico en la Comunidad Académica Española, a través del Grupo de Trabajo de RedIRIS: IRIS-MAIL

Objetivo exponer las mejores recomendaciones para diseñar, configurar y gestionar un Servicio de Correo Electrónico desde el punto de vista de la excelencia en la calidad del servicio ofrecido tanto a los usuarios locales de una organización, como al resto de entidades con las que se intercambia tráfico SMTP.

Estas recomendaciones se estructuran en criterios de calidad, clasificados según su ámbito de aplicación y asignados a diferentes niveles, que permitirán medir la calidad del Servicio de Correo Electrónico de una organización, y promover la mejora del mismo .

MBSA

Microsoft Baseline Security Analyzer (MBSA) es una herramienta fácil de usar diseñada para los profesionales de TI que ayuda a las pequeñas y medianas empresas a determinar su estado de seguridad según las recomendaciones de seguridad de Microsoft y ofrece orientación de soluciones específicas. Mejore el proceso de administración de seguridad utilizando MBSA para detectar los errores más comunes de configuración de seguridad y actualizaciones de seguridad que falten en sus sistemas informáticos.

Tecnologías

OpenLDAP-BSA tendrá tres partes:

a) Elaboración de un informe de los criterios de seguridad que se van a evaluar, explicando en cada uno de los criterios, su valoración , los motivos por los que se realiza

su recomendación, posibles ataques o amenazas que evita, entornos en los que no es imprescindible, como se implementa y como se realiza su comprobación. Podrá servir de guía como plan de base de securización del directorio.

b) Una interfaz gráfica de usuario que irá interrogando al administrador si cumple o no los criterios propuestos, se mostrará el listado de criterios que se irán desplegando y resaltando según se vayan respondiendo.

c) Al finalizar se obtendrá un informe con las respuestas del administrador y como se han valorado.

Las dos últimas partes se basan en el lenguaje **OCIL (Open Checklist Interactive Language)** , y como interfaz gráfica se utiliza el **OCIL Interprete** escrito en java para lo cual se codifican los criterios de seguridad propuestos en **XML** respetando el **Schema OCIL.**

OCIL es desarrollado en el seno del National Institute of Standards and Technology (NIST) define un marco para expresar una serie de preguntas para ser presentado a un usuario y procedimientos para interpretar las respuestas a estas preguntas.

En la seguridad de las IT, las organizaciones trabajan con las políticas de seguridad, detallando que información debe ser protegida y los requisitos de seguridad que deben cumplirse para garantizar que dicha información está realmente protegida.

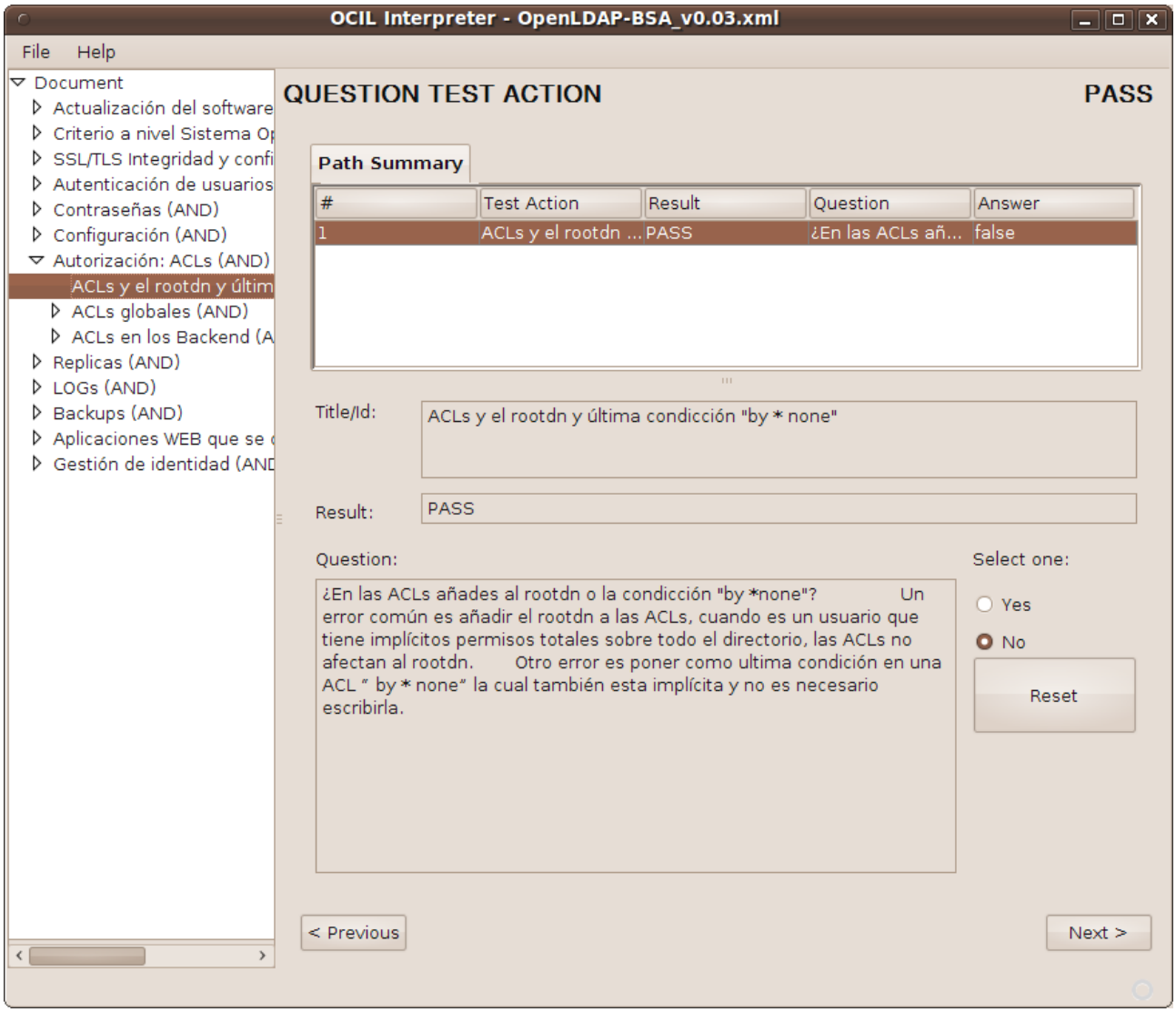
Para verificar el cumplimiento de los requisitos de seguridad, las agencias federales de Estados Unidos han puesto en práctica tecnologías de seguridad que se apoyan en SCAP (Security Content Automation Protocol) iniciativa del gobierno americano acogida en Mitre que pretende poner fin a los problemas de automatización en la gestión de seguridad en cualquier entorno.

OCIL se considera una nueva especificación, por lo que no está incluido actualmente en SCAP. Sin embargo, OCIL puede ser usado conjuntamente con las especificaciones SCAP XCCDF .

En resumen, OCIL proporciona un enfoque normalizado para expresar y evaluar manualmente los controles de seguridad. La siguiente lista define las características soportadas por OCIL:

* Capacidad para definir las preguntas (del tipo Boolean, Choice, numérico o de cadena)
* Capacidad para definir las posibles respuestas a una pregunta y que el usuario puede elegir
* Capacidad para definir las acciones que deben adoptarse como consecuencia de la respuesta de un usuario
* Capacidad de enumerar el conjunto de resultados

La siguiente imagen muestra el aspecto de la aplicación en funcionamiento:

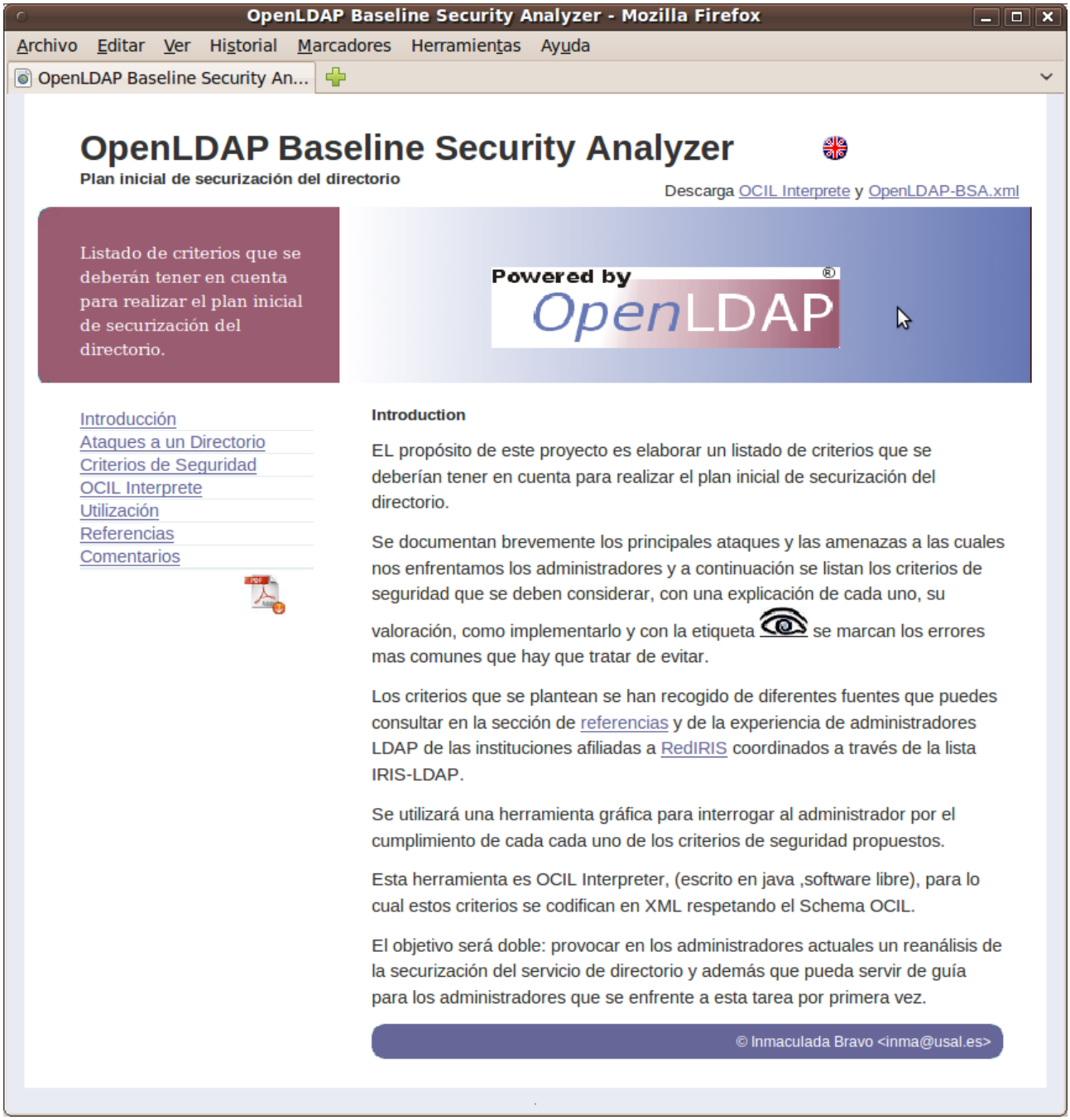


OpenLDAP-BSA en la WEB

Para dar visibilidad al proyecto y que pueda ser de utilidad al máximo número de personas posibles, se publica el proyecto en la Forja de RedIRIS.

https://forja.rediris.es/projects/openldap-bsa/

Diseño el sitio web donde se muestra toda la información, explicando detalladamente el proyecto. El sitio está también en ingles. http://openldap-bsa.forja.rediris.es/



Bibliografía

BlackHat Europa 2008 J.M Alonso y J. Parada LDAP Injection & LDAP Blind Injection

Welcome to IEEE Xplore 2.0: LDAP injection techniques

Ataques a LDAP

Jugando con LDAP

OWASP LDAP_injection

Ldap-injection-blind-ldap-injection.html

Buscando archivos ldif en Google

Security Focus

OpenLDAP, Main Page

Libro: Mastering OpenLDAP: Configuring, Securing and Integrating Directory Services by Matt Butches

ZYTRAX: LDAP for Rocket Scientists

Adam Tauno Williams Presentaciones sobre LDAP

Directory-based Authentication

ESTADISTICAS script en perl analizador logs de OpenLDAP:

Eating Security: OpenLDAP Security

Linux LDAP Tutorial: Add a login, password protection and security to your OpenLDAP directory

ModSecurity: Open Source Web Application Firewall

Howard chu openldap.ppt Free Presentation PowerPoint

SYMAS: Managing Password Policies in the Directory

PerLDAP

IBM Redbooks | Understanding LDAP - Design and Implementation

A world of LDAP and RADIUS

UKUUG Spring Conference 2009

OCIL - The Open Checklist Interactive Language - The Security Content Automation Protocol (SCAP) - NIST

SourceForge.net: OCIL Interpreter: Files

S21sec SCAP - Automatización en seguridad

Parte 1.- AMENAZAS Y ATAQUES A UN SERVIDOR LDAP

En esta sección se recogen brevemente las amenazas a las que nos enfrentamos los administradores de un LDAP para empezar a plantearnos que será lo mas importante a la hora de planificar la securización del sistema.

1. Bugs en el software

Consultando en Security Focus los expedientes de seguridad de OpenLDAP, desde sus inicios hasta la actualidad, descubrimos que se han detectado y corregido 20 vulnerabilidades.

2. Accesos al sistema de archivos del servidor

Si el sistema no está convenientemente securizado se podrían dar accesos no deseados a archivos críticos de la máquina. Más concretamente al archivo de configuración de Ldap (que puede tener configurado el rootpw), la base de datos (sobre todo si no está cifrada), los log de la base de datos y los archivos ldif.

3. Interceptación de la comunicación

Las técnicas que se comentan a continuación utilizan como técnica previa lo que se denomina hombre en medio (Man in the Middle), utilizando envenenamiento ARP, que es un mecanismo que hará creer al cliente que la dirección del servidor está asociada a la MAC del atacante y viceversa, es decir, hará creer al servidor que la IP del cliente está asociada a la MAC del atacante. Esto le permite al atacante ponerse en medio de la comunicación entre el cliente y el servidor.

Esta técnica se puede implementar fácilmente con herramientas como Cain&Abel. Una condición imprescindible es que el atacante debe estar en la misma red que el cliente o que el servidor.

Downgrading de autenticación

Si se utiliza el protocolo SASL como método de autenticación entre el cliente y el servidor, primero tiene lugar una negociación entre ambos para decidir que mecanismo SASL van a utilizar. En este caso el hombre en medio tratará de conseguir negociar como sistema de autenticación Plain Text, aunque tanto el cliente como el servidor soporten mecanismos mas fuertes.

Ataque de Replay

Esta técnica requiere realizar previamente un downgrade del sistema para conseguir negociar como sistema de autenticación preshared key. El atacante utiliza al cliente como generador de las respuestas ante desafíos emitidos por el servidor, para conseguir una conexión autenticada sin tener la clave compartida.

Hijacking de sesiones LDAPs

Ni siquiera estamos completamente seguros con la capa de seguridad TLS.

Esta capa realiza un intercambio de certificados entre el cliente y el servidor y cifrará los datos transmitidos en las conexiones.

Si con la misma técnica de hombre en medio, enviamos al cliente un certificado falso, al cliente le llegará una alerta de CA desconocida, pero posiblemente lo aceptará sin más.

4. Acceso a los datos transmitidos

Mediante un sniffer se puede capturar la información transmitida tanto en redes wireless como en redes cableadas.

Es habitual que servidores web o de correo electrónico accedan al directorio para autenticar usuarios. Las credenciales de los usuarios viajarán por la red primero desde el ordenador del usuario hasta el servidor web o de correo electrónico y en segundo lugar de estos servidores hasta el directorio.

Los administradores de LDAP podemos tener mas control sobre esta segunda etapa:

evitaremos que el servidor LDAP esté expuesto directamente a internet y para incrementar la seguridad podemos forzar que la comunicación viaje cifrada mediante el uso de LDAP-s o TLS. Pero solo en este tramo de la comunicación, donde no es tan imprescindible, ya que los servidores posiblemente estén en la misma red, dentro de un entorno controlado.

Sin embargo, desde un cliente en cualquier punto de internet y el servidor web o de correo electrónico la conexión se realiza de un modo absolutamente impredecible e incontrolado. Es en este entorno donde la exigencia del cifrado de datos mediante una capa SSL es fundamental.

5. Conseguir credenciales utilizando "fuerza bruta"

Cualquier servicio que solicite credenciales de usuario, es decir usuario y contraseña, es susceptible de ser atacado por fuerza bruta, es decir meter en un bucle intentos de validación usando todas las palabras de un diccionario o combinaciones de caracteres. Si estas aplicaciones no prevén mecanismos de control y bloqueos ante intentos fallidos de validación, la seguridad de las credenciales dependerá de la fortaleza de las contraseñas de los usuarios, pero en muchos casos será solo cuestión de tiempo obtenerlas.

6. Inyecciones de código en aplicaciones web

El directorio se suele utilizar para autenticar a usuarios y también para obtener información sobre ellos o sobre otras identidades que almacenamos.

Ambos servicios se suelen ofrecer a través de aplicaciones web.

Si estas aplicaciones aceptan entradas desde los clientes y ejecutan las consultas sin realizar validaciones previas de dichas entradas, los atacantes pueden crear sus propias consultas y extraer información sensible del LDAP.

Para realizar una inyección de código a LDAP es necesario que el filtro original sea múltiple, es decir que tenga un operador OR o AND. En un filtro unitario no es posible realizar una inyección.

LDAP-Injection

Por ejemplo, un modo de validar a un usuario puede ser conectarse al directorio con un usuario privilegiado con permiso para leer el userPassword, y ejecute una consulta con el siguiente filtro:

(&(rfc822MailMember=$usuario)(userPassword=$passwd))

donde $usuario y $passwd son los valores introducidos por el cliente.

Si en alguno de ellos se permite uso de asteriscos, se conseguiría una validación automática del usuario.

(&(rfc822MailMember=$usuario)(userPassword=*))

Si no se permite el asterisco se podría inyectar en ambos campos caracteres suficientes para poder modificar el filtro del programador:

(&(rfc822MailMember= **admin)(!(&(|**)(userPassword= **any))**))

La segunda parte del filtro es siempre TRUE.

El atacante solo necesita escoger un alias válido, no necesita conocer la contraseña para validarse.

Blind LDAP Injection

Los ataques de inyección LDAP a ciegas consisten en que, aunque la aplicación no muestre los datos que devuelve la consulta, la respuesta es diferente si el filtro inyectado genera una respuesta valida o si no genera ninguna respuesta.

En el caso anterior, si incluimos:

(&(rfc822MailMember= **admin)(objectClass=eduPerson)(!(&(|**)(userPassword= **any))**))

Si se valida al usuario, inferimos que el usuario admin tiene como valor del atributo objectClass eduPerson. Si se utilizan búsquedas por patrones y herramientas que automaticen esta tarea, descubrir la estructura de un árbol y el valor de sus datos es relativamente sencillo.

7. Modificación de datos

Hasta ahora los ataques vistos tienen como objetivo obtener información, pero más grave aún es la modificación de los datos.

Es habitual que un usuario pueda cambiar sus datos, sin embargo si esto no se controla con suficiente precisión, quizá podría cambiar algún dato con el que obtenga incremento de privilegios.

Por ejemplo si puede cambiar su propio gidNumber, puede que al obtener una shell en un sistema prefiera pertenecer al grupo de administradores.

Por otro lado, si en alguno de los ataques anteriores se obtienen credenciales de un usuario privilegiado para escribir en una rama o en todo el árbol del directorio, y además no se controlan desde donde se pueden realizar sus conexiones, ni bajo qué medidas de seguridad se pueden realizar, el problema se convertiría en catástrofe.

8. Denegación de Servicio

Aplicaciones mal programadas o ataques intencionados: Si una aplicación realiza consultas a un árbol LDAP filtrando por atributos no indexados, la búsqueda se lleva a cabo de modo secuencial recorriendo todo el árbol hasta encontrar las posibles coincidencias. Estas búsquedas son muy lentas, resultando una aplicación poco eficaz y además puede sobrecargar al servidor LDAP si no se tienen limitados los tiempos de respuesta.

Estas consultas también pueden resultar de ataques intencionados tratando de descubrir información.

Por otro lado aplicaciones o ataques que por algún motivo no cierran las conexiones, pueden generar un número de conexiones suficiente para saturar el nivel de aceptación de las mismas por parte del sistema operativo, provocando que conexiones legítimas sean rechazadas.

9. Google y ficheros olvidados en un servidor web

Un error del administrador puede ser dejar archivos sensibles olvidados en algún directorio que es accedido por un servido web.

Como no está enlazado desde ninguna de nuestras páginas es difícil que alguien descubra el olvido .... pero google si.

Puedes unir las búsquedas en Google y los datos en árboles LDAP buscando las exportaciones de datos del directorio en ficheros LDIF. También existen herramientas que por fuerza bruta son capaces de extraer cualquier contenido accesible.

Así, buscando por filetype:ldif salen muchos ficheros con datos.
Por ejemplo:
- Ficheros Ldif con datos de gente de Madrid, Barcelona, París o lo que quieras.
- Ficheros con datos con gente que se apellide como Chema.
- Ficheros con contraseñas de usuarios.
- Ficheros de Backup LDIF o dumpeo.

Parte 2.- CRITERIOS DE SEGURIDAD

Introducción

En este documento se listan los criterios de seguridad que se deberían tener en cuenta para realizar el plan inicial de securización del directorio.

En esta sección se recogen de manera breve los criterios clasificados en bloques.

Sobre algunos de ellos se ha realizado una primera valoración en cuanto a su relevancia:

IMPORTANTE

RECOMENDABLE

VALORAR .- Dependerá de la propia instalación, se consultará al administrador

Las frases del archivo de configuración slapd.conf están en azul: ej:

include /etc/ldap/schema/core.schema

En el documento aparecerá la etiqueta [OJO] para resaltar errores comunes a tratar de evitar, no solo a nivel de seguridad sino también a nivel de configuración general.

1. Software actualizado

Valoración: IMPORTANTE

Comprobar que el software está actualizado a su última versión estable.

Evitaremos bugs corregidos y disfrutaremos de las mejoras introducidas.

[OJO] A partir de la versión 2.4 no hay soporte para la base de datos lmdb como backend; En su lugar se aconseja migrar a Berkeley DB (bdb). Tampoco tiene soporte para el sistema de mantenimiento de réplicas slurp, por tanto es necesario migrar al syncrep. Estos dos procesos son sencillos de realizar.

2. Consideraciones a nivel del Sistema Operativo

En este apartado se analizan una serie de reglas básicas a nivel de sistema operativo, cómo se ejecuta el demonio slapd, permisos sobre ficheros y directorios y la conveniencia de cifrar la base de datos.

2.1 Reglas básicas

La securización del sistema operativo excede el alcance de este documento, aparte de las siguientes reglas básicas:

2.1.1 Actualización del sistema operativo.

Valoración: IMPORTANTE

Se debe tener un plan de actualización periódica del sistema operativo y aplicar regularmente los parches de seguridad que se publican.

2.1.2 Sistema dedicado y redundante

Valoración: RECOMENDABLE

Es recomendable que el directorio se ejecute en un sistema dedicado, en el que solo se instalen un servidor ssh para acceso a su administración y, en su caso, otros sistemas que dependen absolutamente del directorio como un radius.

Además para dar un servicio de alta disponibilidad se debe redundar en uno o varios servidores, utilizando el sistema de replicación syncrepl.

2.1.3 Firewall interno.

Valoración: VALORAR

Configurar y arrancar un Firewall en el servidor para filtrar las conexiones desde determinadas IPs y hacia determinados puertos; es importante para evitar accesos indebidos.

En este criterio puede que haya discrepancias: si solo se habilitan los servicios

necesarios, si está dentro de una red interna protegida, si ya se dispone de un firewall de frontend...

En todo caso es preferible Iptables que TCP wrappers; este último consume mas recursos de procesado ya que requiere aceptar las conexiones antes de denegarlas.

2.2 Ejecución del demonio slapd

El demonio slapd se puede lanzar utilizando diversos parámetros que analizamos a continuación.

[OJO] Parar el demonio con el kill -9 deja la base de datos inconsistente, en su lugar utilizar kill -INT.

2.2.1 Usuario y Grupo con el que se ejecuta el demonio

Valoración: IMPORTANTE

Crear un usuario y grupo especifico no privilegiado con el que se arrancará el demonio, con las opciones -u ldap -g ldap

2.2.2 Controlar en que interfaces y puertos se ejecuta

Valoración: VALORAR

Por defecto el demonio SLAPD escucha en todas las interfaces de la maquina en el puerto 389. Se puede forzar que solo escuche en ciertas interfaces, puertos y socket con el parámetro “-h protocolo://interfaz:puerto”

2.2.3 Entorno enjaulado

Valoración: BAJO

Ejecutar slapd en un entrono enjaulado CHROOT es una medida extra de seguridad que evitaría que un atacante, aprovechando alguna vulnerabilidad del OpenLDAP, accediera a todo el servidor; sólo podría acceder a la zona enjaulada. Si es un servidor dedicado este criterio no parece imprescindible. Se arranca el demonio con el parámetro “-r directorio”.

2.2.4 Evitar exponer directamente el servidor LDAP a internet

Valoración: RECOMENDABLE

El servidor LDAP suele convertirse en punto de entrada de la mayoría de los aplicativos de la institución, por lo que solo debería estar accesible a los servidores de la red interna.

2.3 Permisos sobre los directorios y archivos del servidor LDAP

Valoración: IMPORTANTE

Los directorios y archivos deben tener los mínimos privilegios para evitar ser accedidos por usuarios no autorizados.

2.3.1 Ficheros de configuración

slapd.conf y ldap.conf

-rw-r----- root ldap sladp.conf

-rw-r--r-- root ldap ldap.conf

2.3.2 Schemas

Los Schemas, pertenecerán a root y podrán leerlo todos

drwxr-xr-x root root Schemas/

-rw-r—r-- root root *.schema

2.3.3 Bases de datos

Base de datos: solo debe ser accedida por el usuario ldap

drwx------ ldap ldap database/

-rw------- ldap ldap *.db?

2.3.4 Ficheros log de la base de datos

Ficheros log de la base de datos: solo deben ser accedidos por el usuario ldap

-rw------- ldap ldap log.00*

Se debe definir una política de rotación de estos log, por la seguridad de eliminarlos del sistema y evitar ocupar espacio inútilmente.

Incluir la directiva DB_LOG_AUTOREMOVE en el fichero DB_CONFIG

No se podrá recuperar la base de datos ante desastres, pero es mas habitual recuperarla de backups almacenados en ficheros ldif.

2.3.5 Ficheros ldif

Estos ficheros son creados para realizar modificaciones o como backups del directorio. Solo deben pertenecer a root. Se debe definir una política para su creación y custodia y para la eliminación de los archivos ldif obsoletos.

2.4 Cifrar la base de datos bdb

En entornos con servidores no dedicados donde el acceso restringido al sistema de archivos no se puede garantizar, una opción para securizar los datos del directorio es cifrar la base de datos, teniendo en cuanta la penalización en la velocidad de respuesta.

3. SSL/TLS Integridad y confidencialidad

En este apartado se analiza la necesidad de evitar que los datos sensibles, como las credenciales de los usuarios, viajen en claro por la red, entre el cliente y el servidor.

Con SSL/TLS se provee de dos elementos importantes de seguridad: por un lado demuestra al cliente la autenticidad del servidor y por otro permite la confidencialidad en la comunicación al cifrar la transmisión de datos.

Se pueden configurar ambas opciones: SSL requiere un puerto diferente para el tráfico

cifrado que suele ser el 636; TLS es un refinamiento de SSL más flexible y permite que los clientes que se conecten al puerto estándar 389 de LDAP puedan escoger entre transmisiones en claro o cifradas; Se negociará **StartTLS** al inicio de la comunicación entre el servidor y el cliente.

Es preferible utilizar TLS a SSL excepto en dos casos: que algún cliente no lo soporte, o que se requiera realizar en un firewall convencional un filtrado por puertos del tráfico cifrado y del no cifrado.

Pueden configurarse ambos protocolos simultáneamente.

La instalación de TLS es sencilla: se requieren la clave privada del servidor, su certificado firmado y el certificado de la entidad certificadora.

Se especifica su ubicación en el archivo de configuración slapd.conf, en el bloque de las directivas globales con las siguientes directivas respectivamente:

TLSCertificateKeyFile
TLSCertificateFile
TLSCACertificateFile

Se puede comprobar si se ha instalado con éxito con el comando ldapsearch y el flag -ZZ.

Para obligar que ciertas operaciones o conexiones se realicen con TLS, se utiliza SSF tanto en las ACLs de los backend como en la configuración global con la directiva security.

Para habilitar también SSL en el mismo servidor, arrancar el demonio con el flag -h

#slapd -h “ldap:/// ldaps:///”

Usar certificado en el lado del cliente y configurar el demonio SLAPD para que solicite al cliente este certificado antes de realizar esta conexión. Esto evitaría la posible suplantación de IPs, (IP spoofing) . Esto no se puede hacer con ACLs . Se puede utilizar la directiva “TLSVerifyClient Demand” en el archivo de configuración ldap.conf.

4. Autenticación de usuarios

Uno de los principales usos de un directorio es la autenticación de usuarios desde diferentes aplicativos.

4.1 Autenticación Simple

La autenticación simple requiere un DN y una password. Hay tres tipos:

anonymous: habilitada por defecto. El DN estará vacío. Este bind se asocia a una sesión anónima. Para deshabilitar la posibilidad de realizar un bind anónimo:

disallow bind_anon

unauthenticated : no habilitada por defecto. Se recomienda no habilitarla. Es muy similar a la anónima, solo requiere un nombre de usuario pero no su password y resulta en una sesión anónima.

user/password: Es la preferida. La password se envía en claro por lo que se recomienda utilizar TLS para cifrar la comunicación entre el cliente y el servidor.

[OJO] Un bind fallido se convierte en una sesión anónima.

Para evitar que se permitan sesiones anónimas:

required auth

La autenticación se puede realizar de dos maneras:

-*Bind rápido:* el cliente provee un DN completo y una password; el usuario anónimo debe tener permisos de “auth” en el atributo userPassword. Si la comprobación es correcta, SLAPD conecta al usuario y le permite realizar otras operaciones.

-*Bind lento:* (BIND → SEARCH → REBIND) el cliente no conoce el DN del usuario pero sí otro atributo, como su uid, o mail. Hay que hacer un bind previo con un usuario privilegiado (con el usuario anónimo no es recomendable), buscar el DN del usuario y hacer un REBIND con el DN del usuario y la password aportada.

4.2 Autenticación SASL

La autenticación SASL es la opción por defecto. Tiene como ventaja que transmite cifradas las password y como desventaja que se deben guardar en claro en el directorio.

Soporta variedad de esquemas de almacenamiento de la password , pero no son estándar. Desventaja: problemas de interoperabilidad.

4.3 Modo correcto de realizar la autenticación de usuarios

Valoración: IMPORTANTE

El objetivo es evitar malos diseños de aplicaciones, que se conviertan en aplicaciones incompatibles, que no se pueda controlar los login fallidos, que se generen problemas de seguridad...

Estos 6 pasos estándar se podrán utilizar contra cualquier servidor LDAP, con cualquier estructura, cualquier tecnología de cifrado o cualquier otro criterio:

1.- Conseguir alias y password del usuario (habitualmente a través de un formulario web).

2.- Enlazar con el LDAP (hacer un bind) con la cuenta de usuario privilegiado creada para la aplicación. También se podría utilizar el usuario anónimo pero es mejor deshabilitar los bind anónimos.

3.- Buscar en el directorio el DN asociado al alias del usuario

4.- Si devuelve una entrada y solo una, éste es el DN del usuario buscado. Si hay cero o más de una entrada se devolverá usuario no encontrado.

5.- Re-bind al LDAP con el DN devuelto en el paso 4 y la password del paso 1

6.- Si LDAP permite el BIND el login ha tenido éxito, si no devuelve “password no válida”

[OJO] Modo incorrecto de realizar la autenticación de usuarios: Sustituir los pasos 3, 4 y 5 por una búsqueda, utilizando un filtro con el alias del usuario y su contraseña.

Es fundamental realizar la autenticación correctamente, no solo para que las políticas de

bloqueos de contraseñas ante binds fallidos sean efectivas, sino también para evitar ataques de **ldap injection**.

Para forzar esto, lo mejor es poner una ACL que evite que ningún usuario pueda leer las password de los usuarios.

4.4 Crear Cuentas Administrativas y Grupos

Valoración: IMPORTANTE

Crear cuentas administrativas para cada aplicación o para delegar funciones, controlando que tengan el mínimo acceso posible tanto a partes del directorio como desde IPs o dominios permitidos.

Crear grupos para definir distintos roles y simplificar la lista de control de accesos definiendo permisos por grupos. De este modo será mas dinámico incluir o eliminar un usuario en un grupo con determinados privilegios, que reescribir ACLs para eliminar o conceder privilegios a un usuario determinado.

5. Contraseñas

En este punto analizaremos la importancia del esquema de almacenamiento de las contraseñas y como implementar y mantener una política de contraseñas, exigiendo complejidad y cambios periódicos, y bloqueando temporalmente cuentas tras varias conexiones fallidas.

5.1 Esquemas de almacenamiento de las contraseñas

Valoración: IMPORTANTE

Evitar mantener guardadas las password en claro o en base64 y evitar crear otros atributos en esquemas personalizados para almacenar password.

Aunque se utilicen esquemas de almacenamiento cifrados, se deben proteger como si estuvieran en texto plano.

El atributo que se suele utilizar es userPassword; es multivaluado y cada valor puede estar almacenado con un formato diferente. Al validar al usuario openldap realiza una iteración con cada valor.

Esquemas de almacenamiento:

-SSHA es el más seguro

-SMD5 es como MD5 incluyendo salt, más seguro.

-MD5 es simplemente un hash en MD5 y se almacena codificado en base64; No incluye salt por lo que es vulnerable a ataques de diccionario.

-SHA es similar a MD5.

-CRYPT Unix-style se guarda un hash de 13 caracteres. Si el sistema cuenta con glibc2 se pueden generar hash de 32-bytes MD5; Ambos incluyen salt

[OJO] El atributo userPassword es multivaluado pero para usar el overlay ppolicy este atributo solo debe tener un valor.

5.2 Política de contraseñas: overlay PPOLICY

Crear una política de contraseñas es importante pero puede ser complejo de implementar y que los usuarios se adapten a ellas.

5.2.1 Complejidad de las contraseñas

Valoración: RECOMENDABLE

Controlar la fortaleza de las contraseñas: mínimo 6 caracteres con números, mayúsculas, minúsculas y algún carácter especial. Evitar que contengan el alias o que pertenezcan al diccionario.

Implementar este control se simplifica si solo se permite cambiar las contraseñas desde un solo aplicativo.

En caso de que se puedan realizar desde múltiples aplicativos los cambios de contraseña y, para asegurar que todos ellos implementan este control, habilitaremos la política de

complejidad de contraseñas con ppolicy.

5.2.2 Obligar a realizar cambios periódicos

Valoración: RECOMENDABLE

Al menos una vez para conseguir que las password sean fuertes. (Este paso va a ser muy criticado, hay usuarios con la misma password desde hace 10 años ..) Y luego decidir el periodo de caducidad, 6 meses o un año.

La complejidad en este caso está en el método de comunicar al usuario que debe cambiar su contraseña. Si para acceder a cualquier servicio existe un único punto de entrada es sencillo, pero no suele darse este esquema.

Como alternativa, registrar los cambios de contraseña con ppolicy y enviar mensajes al usuario recordándoles la antigüedad de sus contraseñas y solicitándole el cambio. Hacerlo obligatorio no es posible, porque no hay modo de asegurarse de que el usuario ha leído su correo antes de usar otros servicios que requieren su validación, como por ejemplo la red WIFI.

Una opción podría ser una vez al año (a principio de curso), se anuncia por correo electrónico que hay que cambiar la contraseña en un plazo generoso. La inmensa mayoría de la gente lee el correo al menos una vez al mes, y si no lo hace, se le restringe algún servicio por ejemplo el envío de correo electrónico.

5.2.3 Bloqueos de Cuentas

Valoración: RECOMENDABLE

Las consultas de autenticación al servicio ldap se realizan desde múltiples aplicaciones y plataformas, algunas de las cuales llevan implícitos mecanismos de bloqueos ante posibles ataques pero otras no.

Ppolicy ofrece la posibilidad de realizar estos bloqueos después de un número de intentos de bind fallidos dentro de un intervalo de tiempo dado y que este bloqueo sea temporal durante un tiempo especificado o permanentemente hasta que el administrador lo desbloquee.

Para que esta política sea efectiva, se debe utilizar el modo correcto de autenticación de usuarios explicado en el punto(4.3)

Ejemplo de política para realizar bloqueos:

```
//10 fallos en un minuto, se bloquea durante cinco minutos
pwdLockout: TRUE
pwdMaxFailure: 10
pwdFailureCountInterval: 60
pwdLockoutDuration: 300
```

No utilizar la directiva **ppolicy_use_lockout .** Poniendo esta directiva se enviará al cliente “cuenta bloqueada” lo que dará información al atacante. Por defecto, una cuenta bloqueda enviará al cliente “credenciales inválidas”.

5.2.4 Comprobar que cuentas se bloquean

Valoración: VALORAR

Es importante chequear las cuentas que se bloquean y comprobar desde que IPs se realizan los binds fallidos para crear un listado de IPs peligrosas y bloquearlas en el firewall. Se listan dos posibles métodos:

a)Analizando logs:

a1.)Añadir al loglevel actual el loglevel 4; Esto añade al fichero de log las modificaciones realizadas en el directorio. En cada bind fallido se añade a la entrada del usuario un atributo operacional pwdFailureTime. Si el número de pwdFailureTime supera el límite marcado por el atributo pwdMaxFailure y si está dentro del periodo de tiempo especificado en pwdFailureCountInterval se añadirá el atributo pwdAccountLockedTime.

Este es el log:

bdb_modify_internal: add pwdFailureTime

bdb_modify_internal: add pwdAccountLockedTime

Se puede analizar el archivo de log y controlar estas entradas para comprobar qué

cuentas se bloquean, desde que ips.. etc.

a.2)Si la carga del ldap es muy alta puede que no sea buena idea incrementar el nivel de log; En este caso podremos analizar en el log los errores “credenciales invalidas“ err=49

send_ldap_result: err=49 matched="" text=""

b) Buscar en el directorio las cuentas bloqueadas: realizar un script que hiciera una búsqueda cada cierto tiempo (podría ser el tiempo marcado en pwdLockoutDuration) del atributo pwdAccountLockedTime . Si se va a optar por esta opción es recomendable crear un índice pwdAccountLockedTime.

6. Configuración slapd.conf

La configuración del demonio se puede realizar de dos maneras:

-Con el archivo slapd.conf: es un archivo de texto plano, donde se especifican todas las opciones de configuración del servidor.

-Con el directorio slapd.d: es un directorio que almacena los parámetros de configuración en una base de datos con una estructura predefinida.

La ventaja de la segunda opción es que se pueden realizar cambios en la configuración sin tener que reiniciar el servidor.

Los nombres de los atributos son los mismos que los nombres de las directivas de configuración del archivo slapd.conf, con el sufijo olc.

Para modificar y acceder a estos atributos se pueden utilizar las herramientas clientes de OpenLDAP.

[OJO] En el archivo slapd.conf, las líneas que empiecen por algún espacio en blanco se consideran continuación de la línea inmediatamente anterior (también si la linea anterior es un comentario).

Las directivas se estructuran en dos bloques: Directivas Globales, al principio del

documento, y de backend o bases de datos escritas después de la sección de definición de cada backend. En un mismo directorio se pueden manejar varios backend. Las directivas especificadas en estos bloques sobreescribirán las directivas globales.

6.1 Evitar ldapv2

Valoración: IMPORTANTE

Por defecto ldapv2 no está habilitado. Algunas instalaciones lo habilitan para permitir compatibilidad con algunas aplicaciones antiguas. Se deberá evitar siempre que sea posible ya que perdemos las capacidades y mejoras que aporta ldapv3.

Con ldapv3:
- permite especificaciones de schemas más potente.
- Descubrimiento del schema
- Permite referrals
- SSL/TLS
- SASL
- Permite renombrar los objetos
- Unicode-> internacionalización
- Extensibilidad.

6.2 Evitar Accesos Anónimos

Valoración: RECOMENDABLE

Como ya se ha explicado en la sección de métodos de autenticación, los bind anónimos y las sesiones no autenticadas se deberían evitar.

disallow anonymous

requiere authc

[Polémica] Dependiendo del entorno, hay técnicos que valoran que no tiene sentido crear un usuario privilegiado para validar usuarios desde cientos de estaciones de trabajo (por ejemplo, en las aulas de informática donde los sistemas Windows se validan con pGina y los sistemas con Linux se validan PAM y NSS) donde las credenciales del usuario

privilegiado son difíciles de proteger, porque cientos de usuarios tienen acceso físicamente a dichas máquinas. En estos casos podría ser igual de práctico permitir utilizar el usuario anónimo.

6.3 Limites

6.3.1 sizelimit

Valoración: RECOMENDABLE

Es el número de entradas que será devuelto al realizar una consulta. Por defecto son 500. Se debería de limitar por ejemplo a 50 entradas, de esta manera se incrementa el trabajo para los atacantes con ldap injection y se evita cargar al servidor. Se devolverán todos los registros encontrados hasta llegar al límite establecido más un mensaje de error “size limit exceed”.

```
sizelimit 50
```

Se puede también especificar el número máximo de registros que pueden ser seleccionados como candidatos para realizar una búsqueda. Dicho número dependerá del tamaño del directorio

```
sizelimit size.uncheked=num
```

6.3.2 idletimeout

Valoración: RECOMENDABLE

Es el tiempo de espera para forzar la desconexión de sesiones idle, para reducir los recursos consumidos por estas sesiones y evitar posibles ataques de denegación de servicio. Una conexión idle está conectada al servidor pero sin realizar ninguna operación. Por defecto no está establecido.

```
idletimeout 90
```

6.3.3 timelimit

Valoración: RECOMENDABLE

Limites en los tiempos de búsqueda. Por defecto 3600 segundos.

Cuando se hacen búsquedas por atributos no indexados se hace un recorrido secuencial por todo el directorio, lo cual sucede por un error en la aplicación que realiza la búsqueda o por un intento de ataque con ldap injection.

```
timelimit  30
```

[OJO]También puede ser por una mala configuración de los índices: para resolver esto chequear de vez en cuando el log buscando entradas del tipo:

```
<= bdb_equality_candidates: (atributo) not indexed
```

si se repite con frecuencia plantearse indexar ese atributo .

Estos límites configurados en el bloque de directivas globales, pueden ser sobreescritos en las directivas de los backend. Por ejemplo en un entorno en el que el servidor LDAP se replica en uno o varios esclavos con syncrepl, el usuario que se utiliza en la replicación no debe ser afectado por estos limites:

```
limits dn="cn=replica,dc=ejemplo,dc=es" size=unlimited time=unlimited
```

6.4 SSF

Valoración: VALORAR

Security Strength Factor; con esta directiva se pueden controlar los criterios de integridad y confidencialidad que se requerirán para realizar ciertas tareas:

```
security ssf=1 update_ssf=112 simple_bind=112
```

En este ejemplo una mínima integridad es requerida en cualquier operación sobre el directorio. En operaciones de actualización de datos y para realizar un bind simple se requiere cifrado de la comunicación con una llave de cifrado de 112 bits como mínimo.

SSF se puede usar de un modo más granulado en las ACLs.

Como contramedida a la vulnerabilidad Downgrading de autenticación se debe evitar algoritmos que supongan un riesgo en la seguridad. También es posible establecer la directiva global security ssf (o de modo más granulado en las ACLs relativas a la autenticación de usuarios privilegiados) sobre todo para realizar cambios o modificaciones en el directorio.

6.5 Pasword del rootdn

Valoración: RECOMENDABLE

La directiva rootdnpw sirve para especificar la password del rootdn; es preferible no utilizarla en el archivo de configuración, y sobre todo no en claro.

Se puede crear una entrada en el directorio para el rootdn; de esta forma cuando se acceda con este usuario se realizará un bind y se podrá controlar desde donde se permitirán estas conexiones.

Para conseguir el cifrado de una password se puede usar el comando slappasswd.

Se debe tener una política de complejidad y cambio periódico de esta contraseña ya que nos es afectada por las políticas impuestas en ppolicy.

7. Autorización: ACLs

En OpenLDAP el mecanismo para autorizar o denegar accesos a ciertas partes del directorio son las ACLs (Listas de Control de Accesos). Su gran versatilidad y su potencia hacen imposible un análisis pormenorizado de todas las casuísticas, por lo cual, en este apartado solo se exponen una serie de consejos generales resaltando los errores más típicos a tratar de evitar.

[OJO]Un error común es añadir el rootdn a las ACLs ya que es un usuario que tiene implícitos permisos totales sobre todo el directorio. Las ACLs no afectan al rootdn.

Otro error es poner como última condición en una ACL “ by * none” la cual también esta implícita y no es necesario escribirla.

[OJO] En la documentación de OpenLDAP las configuraciones que muestra por defecto no son seguras, hay que tratarlas como ejemplos simples para echar a andar el directorio, pero siempre hay que planificar y elaborar una política de ACLs personalizada.

7.1 ACLs globales

Las ACLs en el bloque de las directivas globales, (las escritas antes de definir los backend), afectarán a todos los backend. Se debe proteger en este bloque solo las siguientes partes del directorio: rootDSE y cn=Subschema.

rootDSE : Es una entrada especial que provee información sobre el propio servidor, da información sobre qué controles, características y extensiones serán comprendidas por el servidor y cuales tiene habilitadas. Su DN es una cadena vacía. Se debe permitir su lectura a todo el mundo.

```
access to dn= "" by * read
```

cn=Subschema: Es un registro especial donde se almacena toda la información de los esquemas, incluyendo la definición de atributos y de ObjectClass, también las reglas de búsqueda, formatos, sintaxis y estructuras. Se debe permitir leer a los usuarios autenticados:

```
access to dn=”cn=Subschema” by user read
```

7.2 ACLs en los Backend

Las directivas especificadas en los backend sobreescribirán a las globales.

Origen de la conexión: En primer lugar se deben escribir las reglas relacionadas con el origen de la conexión, es decir limitar desde que IPs o nombre de dominio se pueden conectar ciertas cuentas administrativas o grupos. A continuación se muestran algunos ejemplos:

La cuenta administrativa con privilegios totales en todo es el backend es “el rootdn”. Si no

se establece la password en la directiva rootpw, se debe crear una entrada en el directorio, de esta forma cuando se acceda con este usuario se realizará un bind y se podrá controlar desde donde se permitirá:

```
access to dn=”cn=Manager,dc=ejemplo,dc=es”
        by peername.ip=”127.0.0.1” auth
```

Conexiones con cuentas del grupo de administradores solo se permiten desde la red interna.

```
access to dn.subtree=”group=administradores,dc=ejemplo,dc=es”
        by peername.ip=”10.50.1.0%255.255.255.0” auth
```

También se podría especificar con expresiones regulares:

```
        by peername.regex=”^IP=10\.50\.1\.[0-9]+$” auth
```

En las conexiones con una determinada cuenta administrativa se puede permitir acceso solo desde las máquinas que la utilizarán:

```
access to dn=”cn=courier,dc=ejemplo,dc=es”
        by peername.ip=”10.50.1.18” auth
        by peername.ip=”10.50.1.19” auth
```

Si se pretende que al directorio solo se tenga acceso desde la propia máquina y desde una determinada subred, se puede realizar con el control “break”, para que en caso de coincidencia continúe procesando las ACLs; si el origen no es ninguno de los dos especificados se rechaza la conexión.

```
access to *
        by peername.ip=”127.0.0.1” break
        by peername.ip=”10.50.1.0%255.255.255.0” break
```

[OJO] Se puede limitar el acceso a partes del directorio por IPs o por dominios; Por

dominios es menos confiable ya que se puede cometer el error de utilizar el CNAME, es decir un alias del nombre canónico, en cuyo caso se obtendrá un error en la resolución inversa en el DNs. Hay que asegurarse que se utiliza el A record domain name, para evitar errores.

```
access to dn.subtree=”ou=personas,dc=ejemplo,dc=es”
        by domain.exact=”sistemas.ejemplo.es” write
        by domain.sub=”ejemplo.es” read
```

Réplicas: El usuario utilizado para mantener las réplicas, debe tener acceso a todas las ramas que se van a replicar, o a todo el directorio si la réplica es completa. Además no debe ser afectado por los límites explicados en el punto 6.3.

En el proveedor o máster se debe controlar desde donde se podrá conectar dicho usuario, es decir solo desde los ldap consumidores y que las conexiones se realicen sobre TLS para asegurar la confidencialidad en la comunicación:

```
access to dn=” cn=replica,dc=ejemplo,dc=es”
        by peername.ip=”192.168.13.12” tls_ssf=128 auth
        by peername.ip=”192.168.13.13” tls_ssf=128 auth
```

Si se va a replicar todo el directorio, este usuario deberá poder leer en todo el directorio:

```
access to *
        by dn="cn=replica,dc=ejemplo,dc=es" read
        by * break
```

En los consumidores o esclavos solo se permite que el usuario réplica se conecte desde el localhost y debe poder escribir en todo el directorio.

```
access to dn=” cn=replica,dc=ejemplo,dc=es”
        by peername.ip=”127.0.0.1”
```

```
access to *
        by dn="cn=replica,dc=ejemplo,dc=es" =xw
        by *  break
```

[OJO] Estas ACLs deben escribirse antes de cualquier otra relacionada con accesos a partes del directorio, se deben poner o al principio o justo después de las ACLs de conexión.

La última frase “by * break” indica que si no se produce coincidencia en los “by” anteriores continúe procesando el resto de ACLs. Si no se pone esta frase se denegará completamente el acceso al resto de usuarios. Recordad que la última frase implícita en todas las reglas es “by * none”

Attributo userPassword: restringirlo lo mas posible, de hecho ningún usuario debería poder leerlo. Con el tipo de acceso =xw solo se permite autenticar contra el atributo y actualizarlo pero no leerlo.

```
access to attrs=userPassword,usalPassword
        by self =xw
        by anonymous auth
```

Si se permite a algún otro usuario acceso a este atributo, por ejemplo para administradores que creen nuevas cuentas o que tengan permiso para bloquearlas cambiando las password, debe estar controlado lo más posible.

```
        by dn=”usuario privilegiado” =xw
```

objectClass posixAccount : Esta clase se utiliza para permitir usar LDAP en lugar de NIS . En esta clase se requieren los atributos gidNumber, uidNumber, homeDirectory, uid y cn, y se permiten los atributos descripción gecos loginShell y userPassword. No se debe permitir a los propios usuarios modificar la mayoría de estos atributos.

```
access  to dn.subtree=”cn=personas,dc=ejemplo,dc=es”
                attrs= gidNumber, uidNumber, homeDirectory, uid, loginShell
        by group=”cn=administradores,ou=grupos,dc=ejemplo,dc=es” write
        by self read
```

En este ejemplo se permite modificar ciertos atributos de la clase al grupo de administradores y leer al propio usuario; el resto de usuarios no tienen acceso.

Accesos a grupos: se pueden dar permisos a todos los usuarios que pertenezcan a un grupo, como en el ejemplo anterior.

[OJO] Los grupos deben pertenecer al objectClass groupOfNames y sus miembros se especifican con el atributo “member”, en otro caso se debe especificar en la ACL. Pro ejemplo:

```
access to dn.subtree=”ou=personas,dc=ejemplo,dc=es”
    by group/grupoUsal/memberUid=”cn=adminUsal,ou=grupos,dc=ejemplo,dc=es” write
    by *  break
```

Expresiones Regulares: Se puede expresar ACLs con expresiones regulares. Además se pueden agrupar dichas expresiones entre paréntesis para poder ser utilizadas como variables en las frases “by”. Por ejemplo, si tuviéramos dos ramas una para personas y otra para máquinas, podríamos organizar dos grupos de administradores para gestionar cada una:

```
access to dn.regex=”dc=([^,]+),dc=ejemplo,dc=es”
    by group.exact,expand=”cn=admin,ou=$1,ou=grupos,dc=ejemplo,dc=es” write
    by * break
```

Donde $1 será personas o máquinas. Con una sola ACL se crean permisos para ambos administradores.

[OJO] Consejos para evitar errores frecuentes:

Es preferible “.+” que “.*” . Por ejemplo, en esta expresión dn.regex=”.*,dc=ejemplo,dc=es” , se abarca cualquier rama del dominio ejemplo.es; pero también el propio dominio ya que .* implica ninguno o más caracteres.

Utilizar marcas de principio y fin “^.+,dc=ejemplo,dc=es$”.

Utilizar ámbitos en lugar de expresiones regulares es más eficiente, en este caso dn.children=dc=usal,dc=es.

Es preferible usar la expresión “[^,]+” que “.*” ya que la segunda puede abarcar más RDNs de los esperados: ejemplo dn.regex=”cn=.*,dc=ejemplo,dc=es” podría resultar en cn=**personas**,dc=ejemplo,dc=es pero también cn=**admin,ou=grupos**,dc=ejemplo,dc=es.

Set: se pueden diseñar ACLs complejas utilizando set; permite operadores booleanos y acceso a valores de atributos. Permite crear reglas compuestas por condiciones unidas por los operadores unión “&” e intersección “|”.

Comando slapacl :(desde la versión 2.3) se utiliza para testear si las ACLs escritas tienen el comportamiento deseado. Este comando no se conecta con el servidor a través del protocolo LDAP, sino que arranca su propio SLAPD. Se le puede pasar cualquier nivel de log con el parámetro -d.

slapacl -x -D”que DN trata de acceder al directorio” -b”Recurso al que pretende acceder” “atributo/tipo de privilegio” -d nivel de log

slapacl -D"uid=inma,dc=ejemplo,dc=es" -b "uid=reyes,dc=ejemplo,dc=es" "userPassword/write”

Resultado:

authDN: "uid=inma,dc=ejemplo,dc=es” write access to userPassword: DENIED

8. Réplicas

Valoración: IMPORTANTE

Las ACLs no viajan con los datos por lo que es importante mantener actualizadas las mismas políticas de seguridad en todas las réplicas del directorio.

9. LOGs

Valoración: RECOMENDABLE

En algunas ocasiones con carga de trabajo muy altas es inevitable prescindir de los logs.

Se ofrecen diferentes niveles de log que se pueden usar solos o en combinación con otros simplemente sumándolos o separándolos con un espacio

ej: "loglevel 256 512" es lo mismo que "loglevel 768". Este es el loglevel recomendado.

Los mas útiles:

256: conexiones operaciones y resultados. Muestra 4 a 10 líneas por conexión. Utilizarlo siempre como base unido a algún otro nivel.

512: entradas enviadas

4: heavy trace debugging (muy útil para depurar errores en alguna conexión) no es excesivamente verboso para analizarlo al vuelo pero sí puede serlo para registrarlo.

64: analiza el archivo de configuración.

128: muestra chequeos de ACLs, es muy verboso pero es la única opción para comprobar prohibiciones.

Se utiliza el sistema syslog, es mejor ponerlo en modo no síncrono (-/ruta/ldap.log) para evitar que afecte al rendimiento del servidor ldap.

10. Backups

Valoración: IMPORTANTE

Con el comando slapcat se puede realizar un dump completo de la base de datos a un archivo ldif. En las versiones actuales de OpenLDAP se puede realizar sin parar el servidor. Por ello, para evitar inconsistencias, realizarlo en momentos en los que se prevea que el número de actualizaciones o modificaciones en el directorio sea mínimo.

Se debe tener especificada en un documento la política de copias de seguridad. Es conveniente que estas copias se guarden cifradas.

11. Aplicaciones web que se conectan con el directorio

Valoración: IMPORTANTE

La mayor parte de los ataques a un directorio se producen a través de las aplicaciones web que lo utilizan. Como se ha visto en la sección de “Ataques y amenazas a un directorio” las posibilidades son numerosas.

El principal agujero de seguridad es recoger los datos enviados desde el cliente y utilizarlos sin haberlos filtrado previamente, permitiendo posibles inyecciones de código en las consultas del programador.

Medidas a tomar:

1.- Filtrar siempre los datos recibidos del lado del cliente.

2.- Revisar los datos que se envían al cliente y comprobar la cantidad de información enviada, la cual debe estar limitada.

3.- Realizar autenticaciones: se debe utilizar el modo correcto de autenticación de usuarios explicado en el punto(4.3)

4.- HTTPs: los datos transmitidos entre el cliente y el servidor web deben estar cifrados.

5.- Configurar el módulo de apache modSecurity.

6.- Webservices: Ofrecer a los nuevos aplicativos una pasarela controlada para conectarse al LDAP. De este modo se evitan las posibles inyecciones de código, búsquedas por atributos no indexados o cualquier otro error en la programación de las aplicaciones web.

Un inconveniente es que es el aplicativo del cliente el que recoge el nombre de usuario y su contraseña (que pueden ser registradas en sus logs). Esto está bien solo para

aplicaciones internas.

7.- Instalar un sistema de autenticación centralizada de usuarios, como por ejemplo PAPI, para ofrecer un punto de validación de usuarios a cualquier aplicativo que lo solicite de un modo completamente controlado.

8.- Configurar ACLs sobre los atributos y clases. Dar siempre los mínimos privilegios posibles, sobre todo a las identidades que ejecutan servicios a través de la web.

12. Gestión de IDENTIDAD

Valoración: IMPORTANTE

La institución o la empresa debe tener descrita una política de gestión de identidades, es decir el ciclo de vida de cada usuario en el directorio.

Es un importante agujero de seguridad mantener usuarios con ciertos privilegios después de que su relación con la misma haya terminado.

En nuestro caso hay servicios que se ofrecen indefinidamente como el correo electrónico; Aunque no se elimine dicha entrada, se deben actualizar los grupos o roles a los pertenece un usuario cuando cambia su relación con la institución.

Caso práctico: Actuaciones para la securización del directorio de la USAL

El OpenLDAP de la USAL siempre ha estado en una red interna, no accesible desde internet. En principio solo lo usaban los servidores de correo y el portal de los Servicios Informáticos.

Pero en el último año se ha convertido en el punto de entrada de la mayor parte de las

aplicaciones y servicios, tales como accesos a la red inalámbrica, vpn, docencia virtual, acceso a portales institucionales , incluso para aplicaciones tan sensibles como modificar las actas de notas de alumnos o consultar las nóminas.

Primera Fase:

El primer objetivo fue proteger las Password, para ello se tomaron las siguientes medidas:

1.- Crear ACLs sobre los atributos de password para evitar que se puedan leer con usuarios no autorizados.

2.- Crear cuentas administrativas para cada servicio

3- Evitar que las contraseñas viajen en claro. No solo desde el LDAP y el servidor que ofrece el servicio, eso hubiera sido sencillo implementando TLS y con la directiva security. El problema es securizar las conexiones entre los usuarios y los servicios, por ejemplo: los servidores de correo electrónico deben implementar POPs y SMTPs, el webmail se ofrece a través de HTTPs, portales y otros servicios que precisan autenticar al usuario: se les ha solicitado que implementen SSL.

4.- Evitar autenticaciones incorrectas: Es decir que una identidad privilegiada realice la validación de un usuario, haciendo una búsqueda con un filtro con el usuario y la contraseña.

Segunda Fase:

1.- Añadir directivas que limiten tiempos de consulta, de conexiones idle y el número de registros devueltos.

2.- ACLs que controlen por IP desde donde se conectan las cuentas administrativas.

3.- Evitar que se monten los usuarios del ldap como cuentas del sistema con NSS desde los servidores; esto, además de cargar al LDAP, no es seguro. Es preferible utilizar usuarios virtuales. Esto se controla con ACLs en los atributos uidNumber y gidNumber.

4.- Webservice con nusoap: Ofrecer a los nuevos aplicativos esta pasarela, para conectarse al LDAP. De este modo se evitan las posibles inyecciones de código, búsquedas por atributos no indexados etc. Inconveniente: es el aplicativo del cliente el que recoge el nombre de usuario y contraseña, (que pueden ser registradas en sus logs). Esto

está bien solo para aplicaciones internas.

5.- Instalar PAPI, para ofrecer un punto de validación de usuarios a cualquier aplicativo que lo solicite de un modo completamente controlado.

6.- Comenzar la política de contraseñas: El cambio de password se obliga que tenga mínimo 7 caracteres y que no contenga su alias. Todos los alumnos o trabajadores de la USAL tienen una entrada en el directorio. A los nuevos usuarios, cuando activan su cuenta, se les obliga a cambiar su password .

Tercera fase:

1.- TLS. A pesar de estar dentro de una protegida red interna, es una capa de protección que nunca esta de más.

2.- Reconducir hacia el Webservice o a PAPI los aplicativos que se conectan a LDAP directamente.

3.- Instalar el overlay ppolicy para bloquear cuentas tras varios intentos fallidos y registrar la antigüedad de las contraseñas.

4.- Iniciar una concienciación de la importancia de cambiar las password: hacerlo obligatorio no parece fácil.

Cuarta Fase:

1.- Gestión de identidad.