Universidad de Buenos Aires

Facultades de Ciencias Económicas,

Cs. Exactas e Ingeniería

Carrera de Especialización en Seguridad Informática

Trabajo Final

Tema

Seguridad en Sistemas Operativos

Título

Aspectos de seguridad en los Sistemas Operativos actuales

Autor: Ing. Agustín Roque Cao

Tutor del Trabajo Final: Ing. Juan Alejandro Devincenzi

Año de presentación: 2019

Cohorte: 2016

Declaración Jurada de origen de los contenidos

Por medio de la presente, el autor manifiesta conocer y aceptar el Reglamento de Trabajos Finales vigente y se hace responsable que la totalidad de los contenidos del presente documento son originales y de su creación exclusiva, o bien pertenecen a terceros u otras fuentes, que han sido adecuadamente referenciados y cuya inclusión no infringe la legislación Nacional e Internacional de Propiedad Intelectual.

Agustín Roque Cao

DNI 32.737.125

Resumen

Este trabajo profesional aborda la temática de la seguridad en los sistemas operativos.

Se toman como sujetos de análisis las plataformas Windows y Linux aunque la meta no es dilucidar cuál es considerado “más seguro” sino introducir y analizar los mecanismos de seguridad adoptados por ambas, sobre una serie de tópicos relevantes en la materia.

Se presenta un marco teórico sobre los conceptos a analizar y luego se detallan las decisiones e implementaciones de cada sistema, se establece una breve comparativa destacando las similitudes y diferencias entre ambos esquemas para luego disponer una serie de recomendaciones, en general, de alto nivel, que permitan instituir un marco de seguridad adecuado.

El objetivo de este trabajo es comprender cómo trabajan los sistemas operativos desde el punto de vista de la seguridad y establecer una base sobre la cual construir o mejorar estándares que implementen y controlen medidas de seguridad basadas en las mejores prácticas.

Palabras clave: usuarios de sistema operativo, autenticación, autorización, control de accesos, servicios y demonios, auditoría y eventos de seguridad.

Indice de contenidos

Introducción

Como definición general, puede adoptarse que la disciplina de la seguridad de la información tiene como principal objetivo: garantizar la confidencialidad, integridad y disponibilidad de los activos de información. En este sentido, es necesario proveer de planes, proyectos, herramientas, implementaciones y controles en materia de seguridad, que colaboren en el cumplimiento de las misiones y funciones.

Un esquema de seguridad está conformado por diversos componentes que abarcan desde la seguridad física y lógica hasta la de los recursos humanos, pasando por todos los activos de una organización.

Durante el desarrollo de este trabajo de especialización, se buscará focalizar en uno de los principales desafíos a la hora de implementar un esquema de seguridad lógica: los sistemas operativos.

Para lograr una mayor comprensión del desafío al que nos estamos enfrentando, es vital entender que los sistemas operativos cumplen un rol esencial en la infraestructura tecnológica de una organización, dado que son el soporte sobre el cual se instala y funciona la mayoría del software utilizado y, además, proveen de una interfaz informática utilizable para que los usuarios puedan desarrollar sus tareas diarias en pos del cumplimiento de los objetivos empresariales.

Para abordar la compleja tarea que implica la definición de un esquema o estándar de seguridad en sistemas operativos, se utilizarán como sujetos de estudio, las dos plataformas comercialmente más conocidas: Windows y Linux.

De esta forma se propondrá abordar los conceptos más relevantes en materia de seguridad estructurados en seis capítulos. Cada uno de estos tópicos tendrá una conformación similar: una introducción teórica sobre las ideas necesarias para comprender los elementos de estudio, los detalles y

decisiones adoptados por cada sistema operativo en estos aspectos, una posible visión comparativa de las implementaciones de ambas plataformas y por último, una serie de recomendaciones de seguridad de alto nivel, basadas en las mejores prácticas en la materia, que colaboren para orientar al lector sobre la necesidad de establecer medidas en este sentido.

En el primer capítulo se abordarán los conceptos de usuarios y grupos, que comprenden una base necesaria para entender el funcionamiento de todos los mecanismos de seguridad que se explicarán durante el resto del desarrollo. Los conceptos relacionados a la autenticación, sus sustentos teóricos y prácticos se explicarán en el segundo capítulo.

Durante el tercer y cuarto capítulo se expondrán dos tópicos muy interrelacionados: el control de accesos y la autorización, necesarios para comprender cómo securizar los recursos informáticos de la organización.

En el quinto capítulo se desarrollará la seguridad en los servicios del sistema operativo, indicando, entre otras cosas, cómo administrarlos en forma segura y, finalmente, en el sexto capítulo, se abordará el tema de auditoría y eventos de seguridad cuya importancia es invaluable al momento de requerir trazabilidad sobre los sucesos ocurridos en las plataformas.

Se espera que el desarrollo de este trabajo profesional pueda servir como fuente de consulta para los especialistas de seguridad que se encuentren interesados, ya sea por motivaciones profesionales como laborales, en comprender el funcionamiento de los mecanismos de seguridad en sistemas operativos y, más aún, que sustente una base sobre la cual implementar o mejorar el desarrollo de esquemas que permitan perfeccionar las medidas adoptadas por una organización y ayudar en sus objetivos en materia de seguridad de la información.

1. Usuarios y grupos

1.1.1. Definición

Un usuario o cuenta de usuario es una entidad utilizada para identificar de forma unívoca a una persona que utiliza un sistema informático. Tiene como funciones:

- Representar, identificar y autenticar la identidad de un sujeto.
- Autorizar (conceder o denegar) el acceso a determinados recursos informáticos.
- Permitir auditar las acciones que éste lleva a cabo.

Un grupo es un conjunto de usuarios, equipos y otros grupos que se pueden administrar como una única entidad desde una perspectiva de seguridad. La utilización de grupos tiene como principales ventajas:

- Simplificar la administración. Asignar o denegar derechos, permisos y privilegios comunes a un conjunto de entidades al mismo tiempo.
- Implementar y optimizar modelos de control de accesos basados en roles.

Los usuarios y grupos poseen ciertas propiedades de interés para los sistemas operativos, las mismas, según su naturaleza, se dividen en tres tipos:

1.1.2. Atributos de identificación

Son los datos que refieren a la identificación de un usuario, a semejanza de los datos filiatorios de una persona. Entre los más importantes se pueden encontrar:

- Nombre de usuario: Sirve para identificarse ante el sistema operativo, para efectuar el proceso de autenticación.
- Identificador: Puede ser numérico o alfanumérico, permite al sistema operativo identificar unívocamente a cada usuario o grupo.

Son utilizados para favorecer la ejecución de todas las actividades de un sujeto bajo un contexto de seguridad específico.

1.1.3. Atributos de seguridad

Son los datos que sirven para que cada plataforma pueda articular los procesos de autenticación, autorización y control de accesos de un usuario. Se destacan:

- Contraseña: es el dato privado que sólo conoce la persona propietaria del usuario y que permite al sistema operativo validar la identidad del mismo.
- Atributos de cuenta: fecha de expiración, estado (habilitación y bloqueo), fechas de último login exitoso y último login fallido, entre otros.
- Atributos de contraseña: contador de contraseñas erróneas, historial, fecha de expiración, último cambio, entre otros.
- Grupos: información sobre los grupos a los que pertenece el usuario, incluyendo la posibilidad de un grupo primario para ciertas funcionalidades específicas
- Otros atributos de seguridad como el *shell*, el directorio local (*home*), etc.

1.1.4. Atributos de contacto

Son datos que aportan mayor información sobre la identidad de un usuario, por ejemplo: la descripción, oficina donde trabaja, dirección personal o laboral, e-mail, teléfono, entre otras.

Los grupos tienen prácticamente los mismos atributos que los usuarios, poseen nombre, identificador, lista de sus miembros y atributos de contacto, aunque, por ejemplo, no tienen contraseña.

1.2. Usuarios y grupos en Windows

1.2.1. Identificadores de seguridad

En la plataforma Windows [1,3] tanto a los usuarios como a los grupos y otros objetos que ejecutan bajo un contexto de seguridad determinado se los denomina principales de seguridad (*security principals*). Cada uno de estos sujetos tiene asignado un identificador denominado identificador de seguridad (*Security Identifier* o *SID*) que se conserva durante toda su vida útil.

Cada cuenta o grupo tiene un SID único emitido por una autoridad local o de dominio según su contexto, que se almacena en forma segura en la registry del equipo o como un atributo en Active Directory respectivamente.

La plataforma de Microsoft genera el identificador y lo asigna al momento de la creación de cada entidad. Los SID siempre son únicos y es por ello que las autoridades de seguridad nunca reutilizan los identificadores para otras cuentas, inclusive si las originales ya no existieran.

1.2.2. Arquitectura de un SID

Un SID es una estructura de datos que se divide en tres subpartes que pueden visualizarse con mayor claridad con la notación estándar: ***S-R-X-Y1-Y2-...-Yn-1-Yn*** [1,2,3]

- Los primeros campos contienen información sobre la estructura del SID. “S” indica que se trata de un SID, “R” indica el número de revisión y “X” indica la autoridad identificatoria (*Identifier Authority*).
- Los valores de Y1, Y2, … , Yn-1 corresponden a la jerarquía de dominios a los que pertenece el usuario o grupo.
- Los últimos dígitos (Yn) se utilizan para el identificador relativo (RID) de la cuenta en un dominio específico.

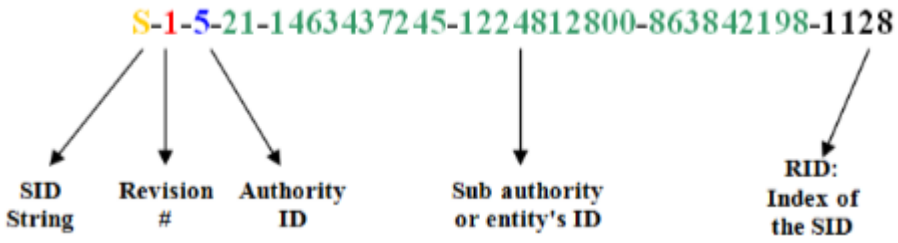

Imagen 1 - Identificadores en Windows - Fuente: [2].

Existen SID predefinidos y públicamente conocidos que permanecen constantes en todas las instalaciones de Windows y que sirven para identificar a los usuarios y grupos que existen en forma predeterminada en la plataforma, por ejemplo el RID del usuario “Administrador” es 500 y el de “Invitado” es 501.

1.2.3. SAM

El almacenamiento y la administración de usuarios locales de un equipo es responsabilidad de un componente denominado administrador de cuentas de seguridad (*Security Account Management* o *SAM*) [2,3].

El SAM se implementa a través de la autoridad de seguridad local que se verá con mayor detalle en el capítulo correspondiente al tópico de autenticación. La base de datos SAM almacena los datos de usuarios y grupos locales junto con sus contraseñas y otros atributos en la registry de Windows bajo las claves HKLM\SAM.

1.2.4. Active Directory

Es el servicio de directorio nativo de Windows que contiene una base de datos para almacenar información sobre los objetos de un dominio.

Un dominio es conjunto de usuarios, grupos, equipos de escritorio, servidores, etc que se gestionan unificadamente. Active Directory guarda las contraseñas, propiedades, privilegios y otros atributos de los sujetos

antes descritos y se vale de la utilización de la autoridad de seguridad local de la misma forma que la SAM [2,3].

1.2.5. Usuarios y grupos predefinidos

Windows define una serie de usuarios y grupos predefinidos que poseen ciertas funcionalidades necesarias por la plataforma [1,4]. Si bien la lista es extensa, algunos de los más conocidos son:

- Usuario Administrador (*Administrator*): que posee control total sobre todos los recursos de su alcance (local o dominio), archivos, directorios, servicios, etc. Puede crear y administrar otros usuarios y grupos.
- Usuario Invitado (*Guest*): que tiene privilegios muy limitados y se utiliza por usuarios puntuales que no tienen cuenta y requieren autenticarse temporalmente.
- Usuarios de servicio (Servicio local, servicio de red, etc): que se utilizan para ejecutar diversos servicios o procesos necesarios por el sistema operativo, no poseen privilegios de administradores totales pero sí los necesarios según su funcionalidad.
- Grupo Administradores (*Administrators*): cuyos miembros poseen los privilegios administrativos sobre el sistema.
- Grupos con privilegios limitados (por ejemplo: usuarios de escritorio remoto): que se utilizan para proveer determinadas funcionalidades sobre un contexto de seguridad específico.

1.3. Usuarios y grupos en Linux

1.3.1. Identificadores de usuario y grupo

En las distintas distribuciones de Linux, el sistema operativo identifica a los usuarios a través de un identificador de usuario (*User Identifier* o *UID*) y a los grupos con un identificador de grupo (*Group Identifier* o *GID*) [2,5,6].

Los mismos son asignados por la plataforma al momento de la creación de una identidad y por defecto existe una directiva de sistema que resuelve dinámicamente los valores de los identificadores, sin embargo, éstos pueden ser modificados o asignados arbitrariamente por otro usuario con privilegios suficientes.

Es importante destacar que, contrariamente a lo que implica el concepto de unicidad, los identificadores en Linux pueden no ser únicos, esto se traduce en que más de un usuario o grupo puede poseer el mismo UID o GID.

1.3.2. Arquitectura de un UID/GID

La arquitectura de los identificadores Linux es bastante más sencilla que los SID de Windows en cuanto a su composición, ya que implica únicamente la restricción de ser un número de tipo entero de 16/32 bits de acuerdo a la versión y distribución.

Las especificaciones de Linux indican que los UID/GID en el rango de 0 a 99 se encuentran reservados para su asignación estática a determinadas identidades, de la misma forma, los del rango de 100 a 999 (499 en algunos casos) son reservados para la asignación dinámica por un administrador. [2,5,6]

Por otra parte, desde el 1000 en adelante, se utilizan para cuentas vinculadas a un LDAP remoto, de todas formas, estas configuraciones se pueden modificar en el archivo /etc/login.defs.

Existen UID y GID predefinidos y públicamente conocidos que permanecen constantes en todas las instalaciones de Linux y que sirven para identificar a los usuarios y grupos que existen en forma predeterminada en la plataforma, por ejemplo el UID/GID 0 identifica al usuario y grupo “root”.

1.3.3. Archivo passwd

La información y relación entre identificadores y nombres de usuario locales en un equipo linux, es almacenada en el archivo /etc/passwd [2,5,6].

Por otra parte, las distribuciones Unix poseen módulos para la administración de usuarios, contraseñas y otros atributos que alteran el contenido no sólo del archivo passwd sino también del /etc/shadow.

1.3.4. LDAP

Si bien las plataformas Linux no poseen un servicio de directorio nativo, pueden integrarse con casi cualquier LDAP, de esta forma, la información de los usuarios, grupos y contraseñas no se almacenan localmente en el equipo, sino en los servicios de directorio correspondientes.

1.3.5. Usuarios y grupos predefinidos

Las distribuciones Linux determinan una serie de usuarios y grupos predefinidos que poseen ciertas funcionalidades necesarias por la plataforma [5,6]. Si bien la lista es extensa, algunos de los más conocidos son:

- Usuario “raiz” (*root*): que posee control total sobre todos los recursos locales, archivos, directorios, servicios, etc. Puede crear y administrar otros usuarios y grupos.
- Usuario “nadie” (*nobody*): que tiene privilegios muy limitados y se utiliza por usuarios puntuales que no tienen cuenta y requieren autenticarse temporalmente. No posee contraseña.
- Usuarios de servicio (ftp, wheel, etc): que se utilizan para ejecutar diversos servicios o procesos necesarios por el sistema operativo, no poseen privilegios de administradores totales pero sí los necesarios según su funcionalidad.
- Grupo “raiz” (*root*): cuyos miembros poseen los privilegios administrativos sobre el sistema.

- Grupos con privilegios limitados (ftp, wheel, etc): que se utilizan para proveer determinadas funcionalidades sobre un contexto de seguridad específico.

1.4. Comparativa Windows/Linux

De acuerdo a los conceptos vistos en el desarrollo del capítulo, se puede evidenciar que ambas plataformas de estudio consideran los conceptos de usuarios, grupos y otras entidades para articular sus mecanismos de autenticación y control de accesos.

Si bien existen diferencias respecto de los nombres y arquitecturas elegidas, ambos sistemas aplican el concepto de identificadores para referir a las identidades mencionadas anteriormente y los utilizan para proveer un contexto de seguridad adecuado para cada proceso.

La principal diferencia en los esquemas presentados puede encontrarse en particularidades de cada implementación, por ejemplo, la forma en la que se almacenan los atributos de los sujetos, en Windows en la registry HKLM\SAM y a través de Active Directory; y en Linux en archivos de acceso restringido como el /etc/passwd o en un LDAP.

Por último, ambas plataformas presentan módulos exclusivos de administración de usuarios, grupos y otros objetos y sus contraseñas, si bien esto no se ve en detalle en el desarrollo anterior, se evidencian similitudes conceptuales (no de implementación) en los tópicos estudiados.

1.5. Buenas prácticas de seguridad

1.5.1. Usuarios privilegiados

- Minimizar la cantidad de usuarios con privilegios elevados o de administrador. El tener más cuentas de este tipo aumenta el vector

de ataque dado que significan mucho mayor esfuerzo la protección de sus credenciales.

- Utilizar los usuarios administradores únicamente para tareas que requieren este tipo de cuentas. Controlar dónde, cómo y para qué se utilizan.
- Quitar funcionalidades innecesarias como por ejemplo la navegación en Internet.
- En Linux, controlar la pertenencia al “uid = 0” y las membresías al grupo “root”.
- En Windows, renombrar los usuarios “Administrator” y controlar la pertenencia al grupo “Administrators”.

1.5.2. Credenciales

- Configurar las contraseñas con un nivel de complejidad alto, acorde a los estándares de la organización.
- Utilizar credenciales únicas para administradores locales y de dominio, no repetirlas en distintos equipos, servidores y ambientes.

1.5.3. Atributos

- Utilizar nombres claros para identificar usuarios y grupos en el contexto organizacional.
- Agregar toda la información necesaria para facilitar la identificación, por ejemplo: la descripción del usuario o grupo.

1.5.4. Otras medidas de seguridad

- Deshabilitar usuarios innecesarios (por ejemplo “Guest” en Windows).
- Segregar los privilegios de los usuarios de acuerdo a su funcionalidad, crear usuarios independientes para una persona que tenga una distintas responsabilidades, por ejemplo: si esa persona administra los servidores pero además requiere navegar en internet

otorgar dos usuarios, separando los permisos necesarios para sus funciones.

2. Autenticación

2.1.1. Definición

La autenticación es el acto que consiste en confirmar que algo o alguien es quien dice ser. En un contexto informático, es habitual que este proceso ocurra cuando un usuario pretende acceder a ciertos recursos de un equipo o servidor en particular y es el sistema operativo, quien intenta verificar su identidad. Dicho esto, para que se produzca una autenticación es necesario un probador (por ejemplo un usuario), un verificador (en este caso un sistema operativo) y una identidad común entre ambos.

2.1.2. Tipos

Existen distintos tipos de autenticación, entre los más utilizados se encuentran:

- Sistemas basados en algo conocido: La identidad se verifica a través de por ejemplo, una contraseña. Es el sistema más conocido y utilizado en la mayoría de los casos.
- Sistemas basados en algo poseído: La identidad se verifica cuando se prueba que el usuario posee algún elemento probatorio. El ejemplo más común es el de un token o una tarjeta de coordenadas al operar con un cajero automático.
- Sistemas basados en biometría: La identidad se verifica a través de alguna característica biométrica del usuario, por ejemplo, una huella dactilar, patrones oculares o patrones de voz.

De acuerdo al nivel de seguridad requerido por la naturaleza o criticidad de los recursos sobre los que se intente probar el acceso, pueden combinarse los tipos de autenticación antes descritos resultando en lo que se conoce como doble o triple factor de autenticación, por ejemplo, en operaciones bancarias, un usuario y contraseña (conocido) más una tarjeta de coordenadas (poseído).

2.2. Autenticación en Windows

Para efectuar las tareas de autenticación en Windows, se requieren una serie de componentes, procesos y servicios que posibilitan el desarrollo de dicho flujo.

Como se muestra en la imagen de acuerdo a la tipo de autenticación, distintos serán los componentes que entran en juego en el proceso.

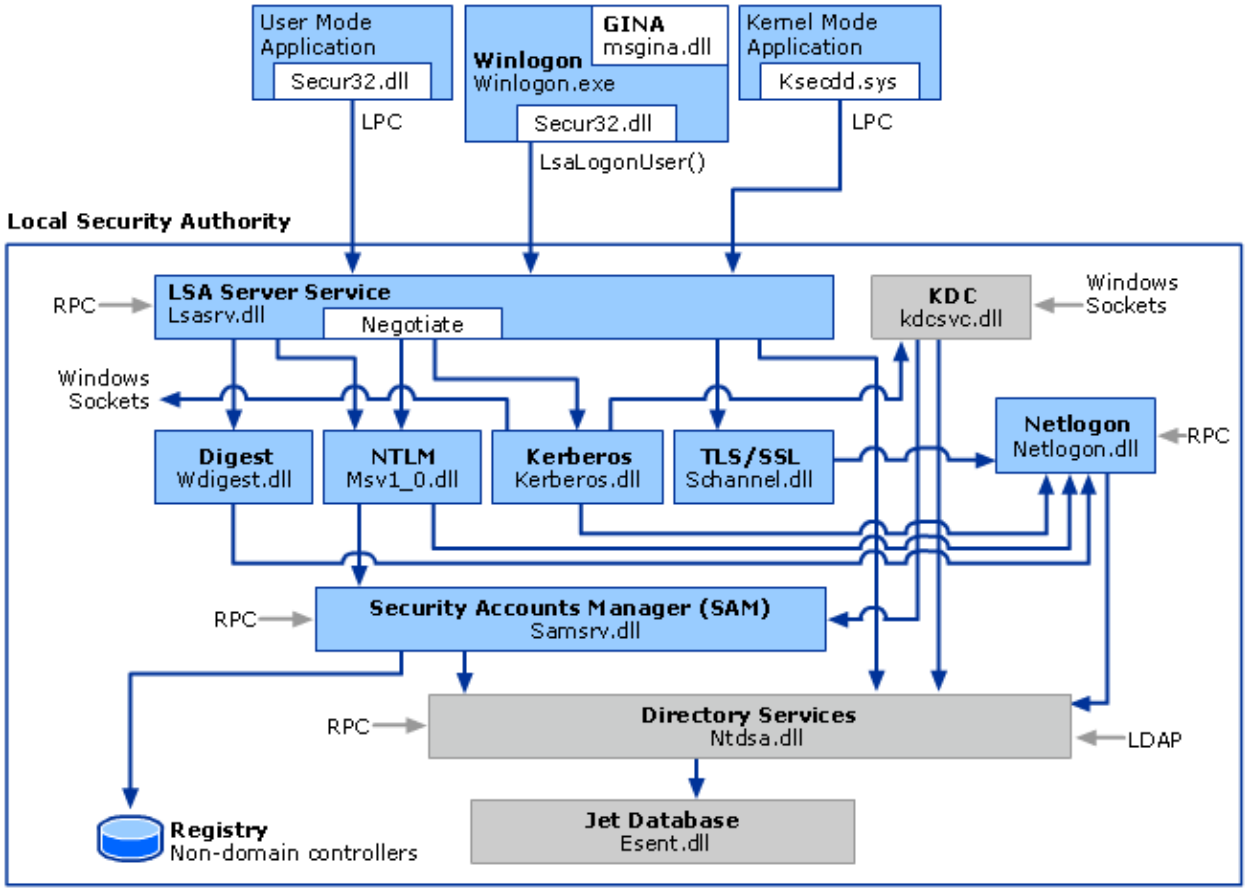

Imagen 2 - Proceso de autenticación en Windows - Fuente: [1].

Al proceso de autenticación en Windows se lo denomina inicio de sesión interactivo (*Interactive Logon*) y se efectúa a través del proceso "Winlogon" con su interfaz gráfica GINA, la utilización de la LSA y sus servicios, los paquetes de autenticación, las bases SAM y Active Directory y los protocolos NTLM (nativo de Microsoft) y Kerberos [1,2,7].

En este sentido, en las siguientes secciones se describen los principales instrumentos intervinientes.

2.2.1. LSA

Para efectuar el proceso de autenticación local/remota en un equipo con Windows, es necesaria la intervención de un proceso de usuario llamado: autoridad de seguridad local [7].

LSA tiene como función principal mantener la información respectiva a todos los aspectos de seguridad local en un equipo, conocidos también como directiva de seguridad local y además se ocupa de varios servicios de traducción entre nombres de usuario e identificadores de seguridad (SID), o envío de eventos al log de seguridad.

La autoridad de seguridad local implementa la mayoría de sus paquetes de autenticación basada en la utilización de librerías dinámicas de Windows (dll).

Estos paquetes son responsables de chequear si la información de usuarios y contraseñas provistos son válidos o no retornando la respuesta al proceso LSA con la identidad del sujeto en cuestión.

La plataforma de Microsoft utiliza los paquetes y protocolos Kerberos: para usuarios de un dominio de Active Directory; y MSV con NTLM (nativo de Windows) para usuarios locales.

Por último, cuando se trata de autenticaciones de dominio, se requiere la intervención de un servicio Windows llamado “Netlogon” que es quien verifica los requerimientos de inicio de sesión, registra y autentica o no contra los controladores de dominio.

2.2.2. Winlogon y GINA

Winlogon es un proceso que ejecuta en modo usuario y que es responsable de interactuar con la LSA en la administración de las sesiones interactivas de los distintos sujetos [7].

Este proceso utiliza librerías dinámicas para proveer de una interfaz gráfica a los usuarios que pretendan efectuar el proceso de autenticación, esta interfaz es denominada GINA (Graphical Identification and Authentication) y se encuentra almacenada en el directorio local \System32\msgina.dll. La interfaz provista por esta dll es el cuadro de diálogo estándar de Windows que permite al sujeto ingresar los datos requeridos para autenticarse, por ejemplo, nombre de usuario, contraseña y dominio (local o remoto) donde reside dicha información.

2.2.3. Protocolo NTLM

Es un protocolo nativo de Microsoft que se basa en un mecanismo de desafío-respuesta (*challenge-response*) entre un cliente y un servidor.

Por defecto, NTLMv2 (versión vigente) es utilizado por el paquete de autenticación MSV y consiste en una serie de pasos que se detallan a continuación:

1) El usuario accede a un equipo (cliente) y consigna los datos de nombre de usuario, contraseña y dominio.
2) El cliente obtiene el digesto de la contraseña (hash HMAC-MD5) y descarta las credenciales en texto plano.
3) El cliente envía un mensaje de negociación para indicar cuáles son sus versiones soportadas del protocolo NTLM.
4) El servidor envía un desafío (*challenge*) al cliente basado en la compatibilidad obtenida del punto anterior con el objetivo de probar la identidad del mismo. En la mayoría de los casos se trata de un mensaje de 16 bytes que contiene un nonce.

5) El cliente cifra el desafío utilizando el digesto del punto 2 como clave y el algoritmo de encripción DES. Luego lo envía al servidor como respuesta (*response*).
6) El servidor envía a la SAM la información de usuario, desafío y respuesta.
7) SAM compara la información enviada por el servidor con la almacenada en la base y si la misma coincide, la autenticación es exitosa.

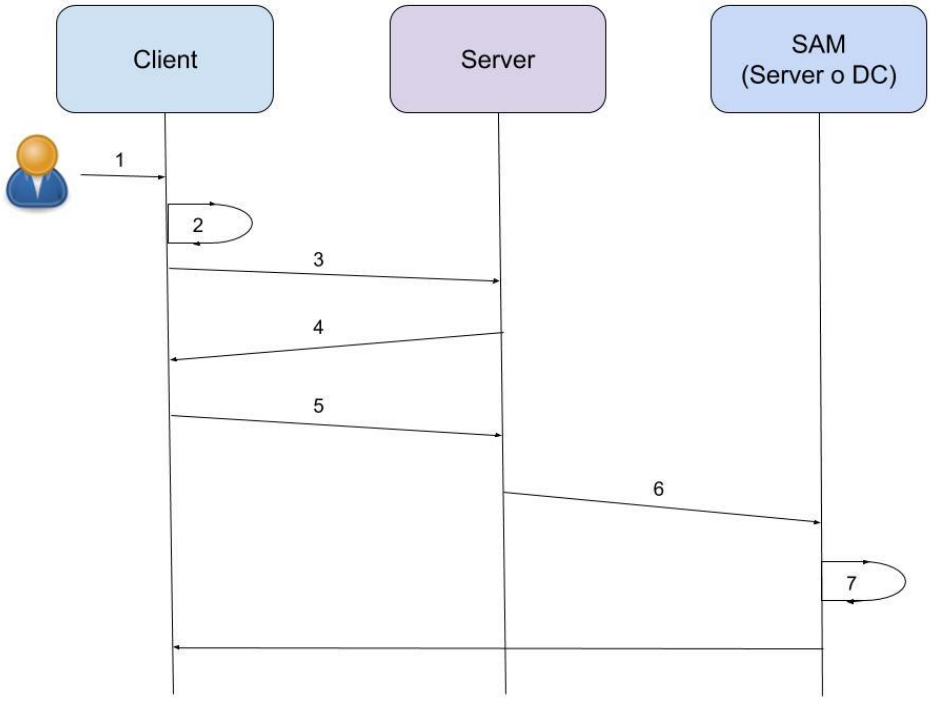

Imagen 3 - Proceso de autenticación vía NTLM - Fuente: Elaboración propia.

2.2.4. Protocolo Kerberos

Kerberos es un protocolo de autenticación en la red cuyos atributos son sólidos en materia de seguridad del tipo cliente / servidor [8]. Existen implementaciones tanto gratuitas como comerciales.

Windows utiliza el protocolo Kerberos para la autenticación de usuarios de dominio mediante Active Directory. En Linux, eventualmente, puede establecerse un LDAP como reino Kerberos para utilizar el protocolo.

Para entender el funcionamiento de Kerberos, es importante definir algunos objetos que intervienen en el proceso:

- Cliente: Típicamente un usuario que desea utilizar un servicio.
- *Host* (anfitrión) o Servidor: Un equipo o servidor accesible a través de la red.
- Kerberos *Realm* (reino): Cada base de datos donde se almacenan los usuarios, grupos, etc. Por ejemplo: un dominio de Active Directory o LDAP.
- *Key Distribution Center* (KDC): Un equipo que se encarga de la emisión de tickets.
- *Ticket Granting Service* (TGS): Servicio que se encarga de la emisión de tickets.
- *Ticket-Granting Ticke*t (TGT): Un tipo de ticket que permite al cliente obtener otros tickets en el mismo reino.
- *Ticket*: Un permiso temporal para un usuario para la utilización de un servicio.
- Servicio: Cualquier programa o equipo que sea accesible a través de la red. Por ejemplo: ssh, ftp, etc.

El protocolo consiste en una serie de pasos que se detallan a continuación:

1. El cliente envía un mensaje de solicitud al servicio de autenticación del KDC.
2. El servicio de autenticación responde al cliente con un ticket para el TGS.
3. El cliente envía un mensaje de solicitud para un servidor en particular.
4. El TGS responde enviando el ticket requerido.
5. El cliente utiliza el ticket del punto anterior para solicitar la utilización del servicio específico.
6. El servicio responde y concede el acceso al cliente.

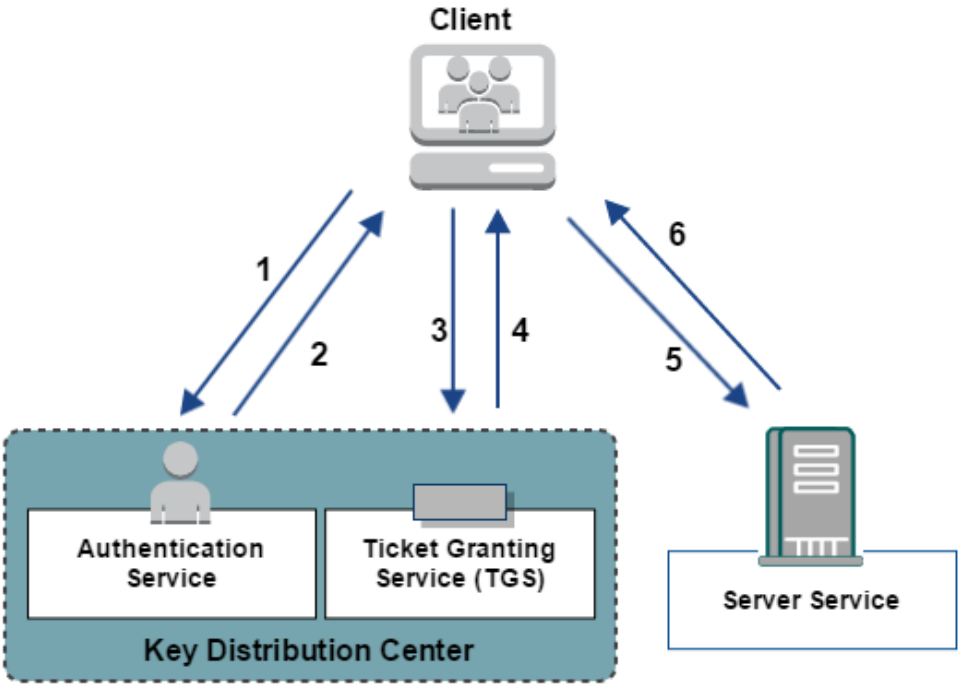

Imagen 4 - Proceso de autenticación vía Kerberos - Fuente: [8].

2.3. Autenticación en Linux

2.3.1. PAM

Los módulos de autenticación conectables (PAM) son librerías de Linux que proveen de distintos métodos de autenticación, además de las interfaces y funciones necesarias para completar el proceso [2,9].

PAM posee un archivo de configuración ubicado en /etc/pam donde el administrador puede definir qué tipo de mecanismo probatorio se utilizará para comprobar la identidad de un sujeto en cada caso. Una vez que la librería PAM es iniciada, el sistema operativo carga el módulo de autenticación correspondiente.

Linux es una plataforma que se encuentra modularizada y esto no varía desde el punto de vista de seguridad, en este sentido, el módulo de autenticación posee varios procedimientos de este tipo que incluyen la creación de esquemas y asignación de privilegios a los usuarios autenticados.

2.3.2. etc/passwd

Para efectuar una autenticación local/remota en un equipo cuyo sistema operativo sea alguna distribución de Linux, es necesaria la intervención de ciertos archivos de sistema que almacenan información sensible de usuarios y grupos.

Los ficheros /etc/passwd y /etc/group guardan los atributos más relevantes de usuarios y grupos respectivamente, en el caso de los primeros, almacenan inclusive sus contraseñas [5,6,9].

Cada registro del archivo passwd guarda el nombre de usuario, su contraseña cifrada, el identificador del usuario y del grupo principal del mismo, su descripción, el directorio home y el shell; todos ellos separados por el caracter ":".

Debido a que se utiliza durante el proceso de autenticación, todos los usuarios deben tener lectura de este archivo, esto, junto con su cifrado actualmente debilitado, hacen de que la seguridad de este método diste en demasía de la esperada. De esta forma, todos los usuarios conocen las credenciales cifradas de todos los usuarios y con las herramientas de *crackeo* de contraseñas actuales, resulta increíblemente sencillo obtener los secretos planos.

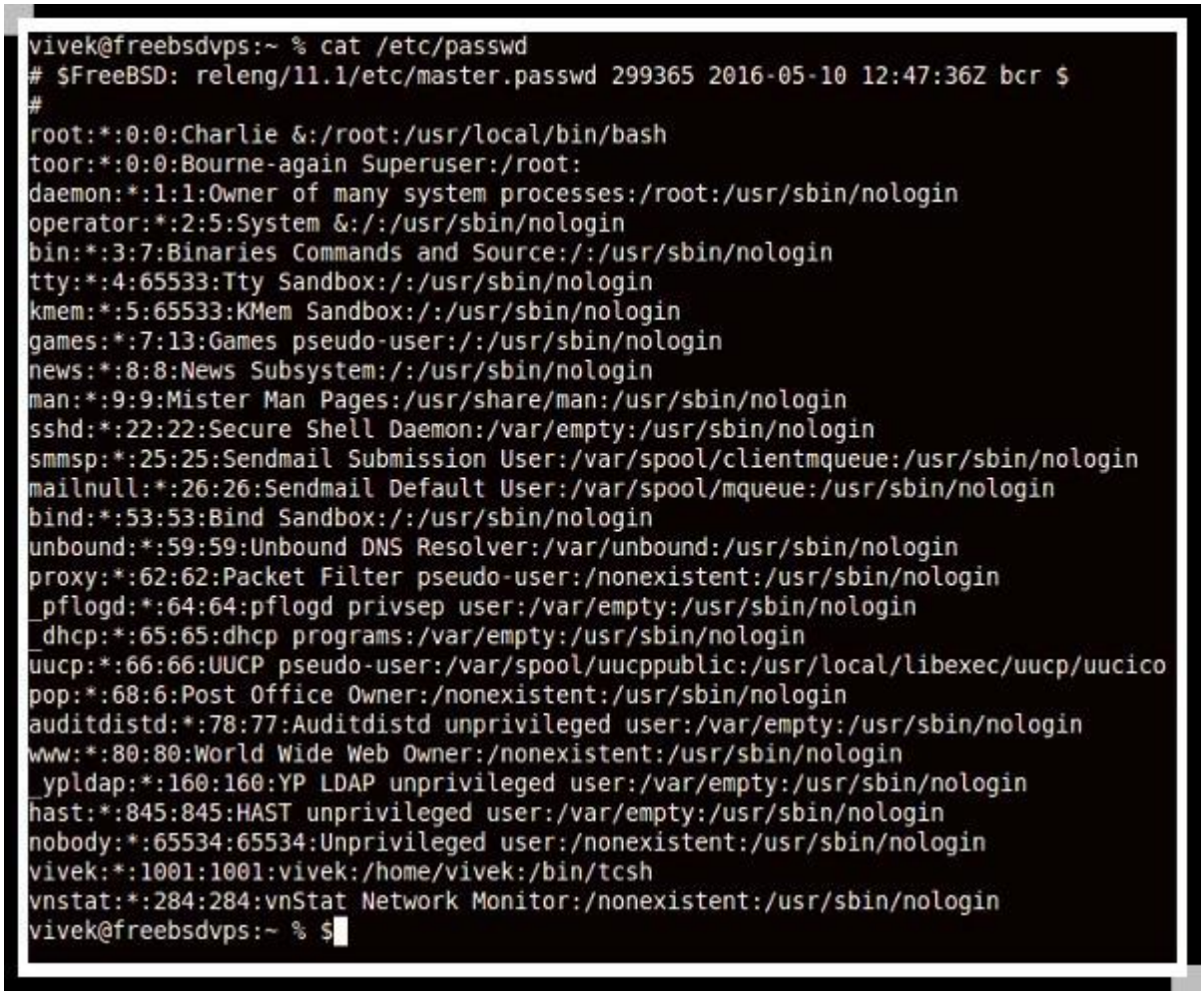

Imagen 5 - Ejemplo de archivo /etc/passwd que utiliza shadow password - Fuente: Elaboración propia.

2.3.3. etc/shadow

Para corregir las debilidades mencionadas anteriormente, surge el método “shadow password file” que implica la existencia del fichero /etc/shadow que almacena información en registros que contienen: nombre de usuario, contraseña cifrada con métodos de encripción más robustos (una variante del algoritmo MD5), y otra información relativa a las credenciales como por ejemplo: fecha de último cambio de contraseña, días para solicitar el cambio o fecha de expiración de la cuenta [5,6,9].

La forma de verificar si se está utilizando shadow en un equipo Linux, es visualizar que en el archivo /etc/passwd, el segundo campo de cada

registro, es decir la contraseña del usuario, se encuentre reemplazada por el caracter “X” o “*”.

La utilización del método shadow también soluciona el inconveniente del permiso de lectura obligatorio de todos los usuarios al archivo passwd, ya que, utilizando este sistema, solo los usuarios privilegiados, por ejemplo el root, deben tener acceso al mismo.

Imagen 6 - Ejemplo de archivo /etc/shadow - Fuente:[https://linuxforgeek.com/etc-shadow-file-linux/].

2.3.4. Protocolo LDAP

LDAP es un servicio de directorio que permite almacenar las entradas de los usuarios y grupos, junto con sus atributos, entre ellos sus contraseñas, en una estructura jerárquica en forma de árbol.

Cuando se establece una sesión entre un usuario de dominio y un equipo Linux que utiliza autenticación LDAP, el proceso que interviene se denomina Bind y el mismo es el responsable de autenticar la conexión y establecer un contexto de seguridad para los sujetos. Por otra parte, la configuración de la SASL (Capa de Seguridad para Autenticación Simple) en el sistema operativo, define, según la RFC 4513, dos mecanismos de autenticación posibles: Simple Bind o SASL Plain y SASL Bind.

El primer método hace que el cliente envíe el nombre de usuario, en concreto el atributo “nombre distinguido” (DN) y la contraseña (sin cifrado) al servidor LDAP quien busca el objeto en su estructura, compara la credenciales y autentica la conexión si las mismas coinciden. Se debe tener en cuenta que, debido a que los secretos se proporcionan en texto plano, se recomienda mitigar dicho riesgo protegiendo los enlaces a través de la utilización de TLS.

SASL Bind está más involucrado desde el punto de vista de la seguridad y permite que el cliente y el servidor negocien un mecanismo de autenticación particular, por ejemplo Kerberos, cuyo funcionamiento se explicó anteriormente.

2.4. Comparativa Windows/Linux

De los conceptos abordados en el presente capítulo, se puede observar que ambos sistemas operativos implementan varios mecanismos de autenticación que intentan adaptarse a las necesidades de los procesos.

Si se comparan los mecanismos básicos y predefinidos para probar la identidad, Windows pareciera tener una ventaja con su paquete MSV que utiliza el protocolo NTLMv2 con un cifrado HMAC-MD5, superior al de Linux con su esquema de contraseñas shadow.

En cuanto a la integración de la autenticación contra un dominio o servicio de directorio, Microsoft utiliza el producto nativo Active Directory e integra en su mecanismo el protocolo Kerberos que es quizás, el más seguro actualmente. Linux, por su parte, puede integrarse a un LDAP e inclusive utilizar al mismo como un reino Kerberos.

En conclusión, la mayor ventaja de Windows es la utilización de esquemas robustos con tecnologías nativas, en cambio Linux, tiene la virtud de poder integrarse a otros productos y endurecer la seguridad tanto como se desee siempre que se opte por otros métodos que no sean los definidos por defecto.

2.5. Buenas prácticas de seguridad

2.5.1. Credenciales

- Configurar las contraseñas con un nivel de complejidad alto, acorde a los estándares de la organización.
- Implementar políticas que reflejen el nivel de seguridad de autenticación definido.

2.5.2. Protocolos

- Utilizar protocolos conocidos que implementen métodos de encripción seguros con implementaciones estándar, robustos y no quebrados aún.

2.5.3. Mecanismos

- Establecer los mecanismos y tipos de autenticación adecuados de acuerdo a la sensibilidad y criticidad de las operaciones a realizar, por ejemplo, doble factor en una operación bancaria, usuario + contraseña para equipo de trabajo diario.

3. Control de accesos

3.1.1. Definición

En un contexto informático, el control de accesos refiere a las funciones, permisos, privilegios y su administración, que tienen los operadores de un sistema sobre los recursos existentes en el mismo. Para comprender mejor el concepto, es necesario definir los términos sujeto y objeto de un sistema operativo.

Los sujetos, de acuerdo a lo explicado en el primer capítulo, son los usuarios, grupos, procesos, hilos, etc que operan e interactúan sobre los recursos disponibles. Los objetos son dichos recursos, entre los que podemos encontrar archivos, directorios, servicios, puertos, dispositivos de entrada/salida, entre otros.

El control de accesos es un pilar fundamental en la seguridad informática, en este sentido, es importante contar con los mecanismos necesarios para proporcionar una apropiada segregación de permisos y privilegios de los sujetos, así como su facilitar su administración y auditoría.

3.1.2. Permisos y herencia

Un permiso es un tipo de acceso, una forma de utilización, denegación o manipulación que un sujeto tiene sobre un recurso u objeto.

Los más característicos son compartidos por todos los sistemas operativos y son los de ejecución, lectura, escritura o control total. Luego existen permisos especiales que varían según el sistema operativo pero que son necesarios para proveer mayor cantidad de opciones en la administración del control de accesos. El significado o funcionalidad que otorga cada permiso varía en según el recurso sobre el que se aplique.

El concepto de herencia refiere a la transitividad de permisos de un objeto sobre otros contenidos en él. El ejemplo más sencillo es la aplicación

de un privilegio sobre un directorio en particular y su transición automática por herencia, de forma que los archivos y subdirectorios contenidos en él, también posean el privilegio asignado.

La herencia es administrable y puede ser desactivada en el recurso que desee el administrador o bien puede elegir a qué elementos es propagada y a cuáles no.

3.2. Modelos de control de accesos

Existen tres tipos o modelos en lo que refiere al control de accesos: discrecional, basado en perfiles o roles y mandatorio [10].

En la imagen que se muestra a continuación, se incluyen los elementos intervinientes en cada modelo y su relación, para, a continuación, describir los fundamentos en los que se basa cada uno.

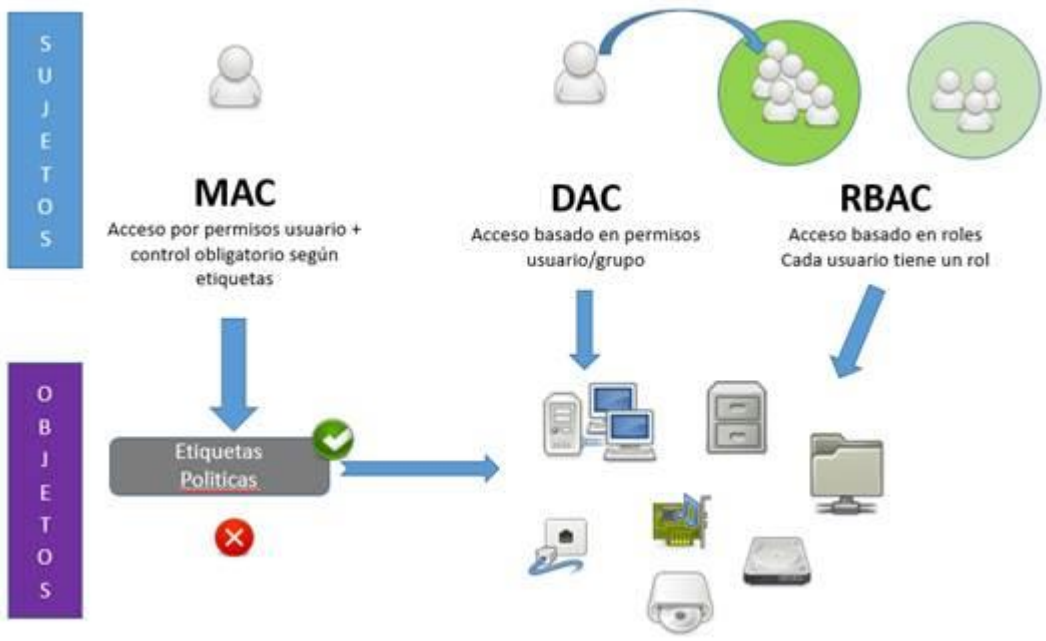

Imagen 7 - Modelos de control de accesos - Fuente: [10].

3.2.1. Modelo discrecional (DAC)

El control de accesos discrecional o DAC por sus siglas en inglés (Discretionary Access Control), es un método que basa el acceso o no a determinado objeto, en la identidad de los sujetos [10]. Tanto Windows como Linux utilizan en su mayoría DAC, el mecanismo más conocido es el de permisos establecidos por el dueño, donde es el propio dueño del objeto quien determina qué usuarios y con qué privilegios acceden a sus objetos.

Para este tipo de control de accesos es habitual utilizar los permisos o atributos de lectura, escritura, ejecución, etc que varían su significado según el tipo de objeto sobre el que se aplica el privilegio, por ejemplo, no es lo mismo un permiso de escritura sobre un directorio (permite agregar o eliminar archivos) que sobre un archivo (permite editar el contenido del mismo).

3.2.2. Modelo basado en roles (RBAC)

Como una evolución desde el punto de vista de la administración y granularidad respecto del DAC surge el control de accesos basado en roles o RBAC (Role-Based Access Control) [5,10], que proporciona mayor versatilidad para organizaciones que en sus sistemas tienen una gran cantidad y variedad de usuarios y funciones.

El RBAC consiste en la definición de roles o perfiles a los que se les asignan una serie de privilegios y acciones que pueden llevar a cabo dentro del sistema de acuerdo a las funciones de las áreas representadas por esos roles. Por ejemplo, podrían existir roles como sistemas, finanzas, recursos humanos y cada uno tendrá los permisos necesarios para desarrollar su trabajo en lo que refiere a la utilización de los recursos como archivos, directorios, etc.

Con este mecanismo se segmenta y se organiza de forma más eficiente el acceso a los objetos y las tareas.

El modelo basado en roles es muy implementado en servicios de directorio como LDAP o Active Directory.

3.2.3. Modelo mandatorio (MAC)

El control de accesos mandatorio o MAC (Mandatory Access Control) aporta una capa más desde el punto de vista de seguridad en permisos y privilegios y eventualmente se complementa con los modelos vistos anteriormente [2,5,10].

MAC es el más estricto de todos los niveles de control y su utilización más conocida es la del gobierno de Estados Unidos o su milicia.

El modelo mandatorio está basado en un enfoque jerárquico donde el acceso a todos los objetos está definido por el administrador del sistema.

El elemento esencial para comprender su funcionamiento son las etiquetas de seguridad, que, a su vez, tienen dos componentes: una clasificación (por ejemplo: *top secret*, *confidential*, etc) y una categoría que es una tipificación del proyecto, departamento o nivel de gestión al que pertenece el recurso.

De esta forma, todos los sujetos y objetos son catalogados con estas etiquetas de seguridad, entonces, al momento de la comprobación de un acceso, se comparan la clasificación y la categoría de ambos y se determina su autorización o no.

Si bien el MAC es el control de accesos más seguro, su planificación, administración y mantenimiento suelen ser mucho más costosos para los administradores del sistema.

Implementaciones representativas del modelo discrecional son [15] SELinux para RedHat y el nivel de integridad en el entorno Windows.

3.3. Control de accesos en Windows

Windows adopta el modelo discrecional como el principal en su esquema de control de accesos, y, si bien cuenta con la posibilidad de

implementar RBAC o MAC, sólo se abordan a continuación, los detalles de implementación del DAC.

Todo objeto o *securable* en Windows posee una estructura de datos asociada denominada *Security Descriptor (SD)* o descriptor de seguridad, que contiene información sobre el control de accesos a dicho objeto [1,3]. Un SD contiene información sobre:

- El propietario o dueño del objeto, almacenando el SID de un sujeto.
- El grupo primario del dueño del objeto, almacenando el SID de dicho grupo.
- La lista de control de accesos discrecional (DACL) que determina qué accesos y de qué tipo son permitidos a los sujetos.
- La lista de control de accesos de auditoría (SACL) que determina qué accesos y de qué tipo son auditados a los sujetos.

A su vez, cada lista de control de accesos está compuesta por entradas denominadas ACE (*Access Control Entry*), que contienen la información individual sobre el tipo de acceso de un sujeto particular sobre el objeto en cuestión, por ejemplo: el sujeto “A” posee permisos de lectura y ejecución sobre el archivo “B.txt”.

Cada ACE de la lista de accesos discrecional puede tener cuatro valores posibles: acceso permitido, acceso denegado, objeto permitido u objeto denegado, mientras que en el caso de la lista de accesos de auditoría, puede ser: de auditoría de sistema o de objetos.

En la siguiente imagen pueden visualizarse los componentes de un SD:

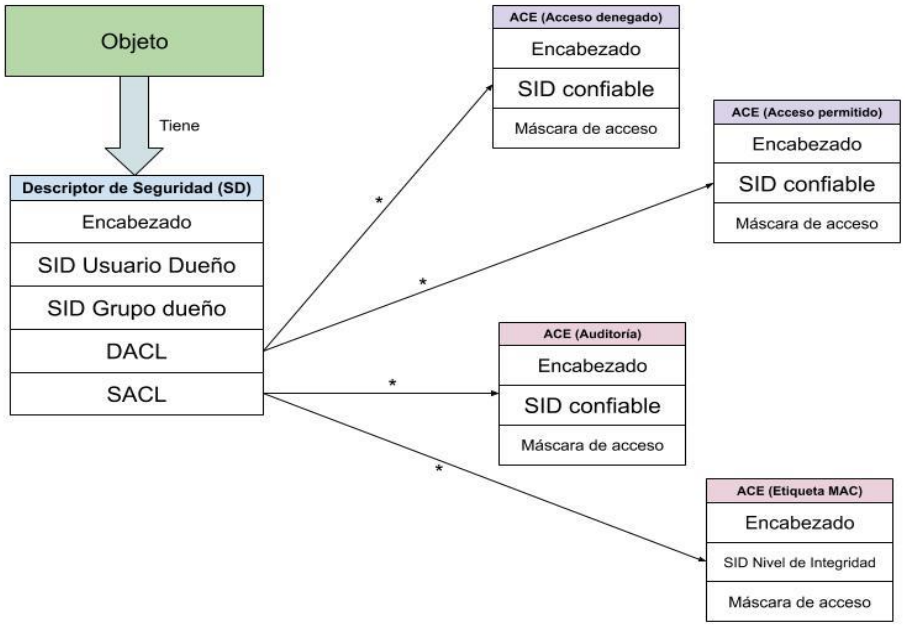

Imagen 8 - Contenido de un descriptor de seguridad (SD) - Fuente: Elaboración propia.

Cuando un usuario intenta acceder a un objeto, Windows compara el token de acceso del usuario contra el SD del objeto y autoriza o deniega el requerimiento en función a la misma. La conformación del token se verá con mayor detalle en el capítulo siguiente, sin embargo, en la imagen a continuación pueden verse los elementos que entran en juego al momento de la comprobación de los accesos de un sujeto.

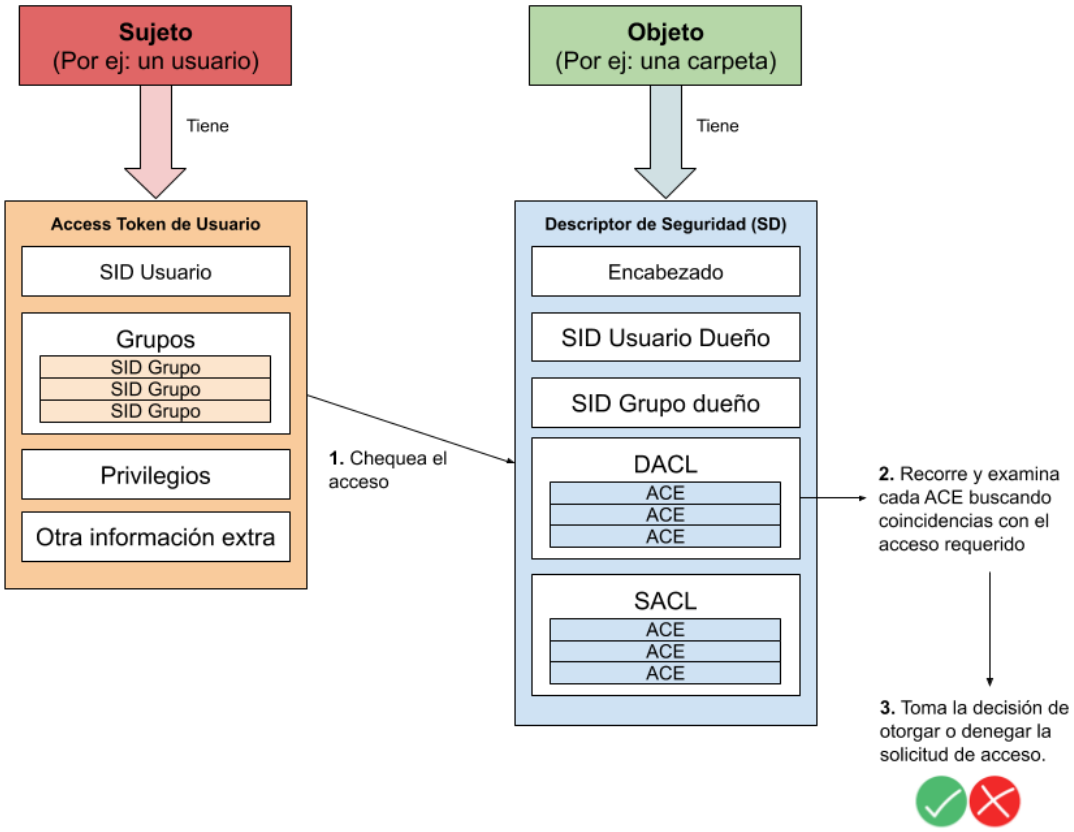

Imagen 9 - Comprobación de accesos a un objeto - Fuente: Elaboración Propia.

3.4. Control de accesos en Linux

Linux, al igual que Windows, también adopta el modelo discrecional como el predeterminado en su esquema de control de accesos, y, si bien a continuación se explican los detalles de implementación del mismo, es importante tener en cuenta que el modelo mandatorio es utilizado para forzar privilegios generales a los sujetos que se ve con mayor detalle en el apartado correspondiente al tópico de autorización.

Todos o casi todos los objetos dentro de sistemas Linux son archivos o directorios, los mismos tienen una estructura asociada que puede denominarse: lista de control de accesos (ACL) [5,6]. Cada ACL se encuentra formada por tres ternas:

- Una terna para los permisos del usuario dueño del objeto.
- Una terna para los permisos del grupo del usuario dueño del objeto.
- Una terna para los permisos del resto de los usuarios que no pertenezcan a los dos items anteriores.

A su vez, cada terna contiene tres posibles atributos: lectura (r), escritura (w) y ejecución (x). El modelo de asignación de estos permisos se denomina octal, y responde a los valores 4,2 y 1 respectivamente. Por ejemplo: el valor 5 en la segunda terna indica que el grupo del dueño del archivo tiene permisos de lectura y ejecución (4 + 1).

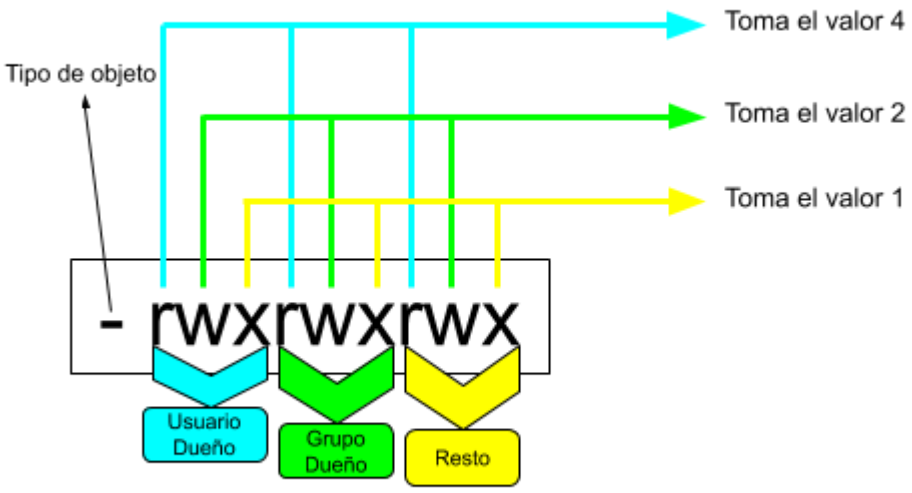

Imagen 10 - Lista de control de accesos en Linux - Fuente: Elaboración Propia.

Existen otros permisos especiales como setuid, setgid y stickybit que añaden más opciones al DAC de Linux y que utilizan una cuarta terna de atributos:

- Setuid: Al ejecutarse un archivo o crearse un archivo en un directorio, los permisos resultantes son los del dueño del archivo o directorio.

- Setgid: Al ejecutarse un archivo o crearse un archivo en un directorio, los permisos resultantes son los del grupo del dueño del archivo o directorio.
- Stickybit: En archivos modifica el vuelco de memoria a disco, en directorios cada usuario borra o modifica sus propios archivos.

3.5. Comparativa Windows/Linux

Se puede visualizar que ambos sistemas operativos utilizan el concepto de lista de control de accesos en lo que al modelo discrecional (predeterminado tanto en Windows como en Linux para el acceso a recursos) refiere. Sin embargo, las diferencias en las implementaciones son sustanciales a la vez que su administración.

El modelo mandatorio puede ser implementado en Linux para restringir el acceso a ciertos objetos del sistema, y mientras que en Windows existe la alternativa de un modelo similar a través de los niveles de integridad, esto se suele realizar a través de la asignación de derechos de usuario y privilegios que se ven con mayor detalle en el capítulo siguiente.

Por último, si bien la plataforma de Microsoft implementa, como método de auditoría de accesos a objetos, el concepto de SACL, en Linux, no se encuentra llevada a cabo una idea similar.

3.6. Buenas prácticas de seguridad [11,12]

3.6.1. Diseño e implementación

- Diseñar el acceso a los recursos basándose en roles o grupos que sirvan para segregar las distintas misiones y funciones de las áreas de la organización. No todos los empleados necesitan tener acceso a todos los recursos. Por lo tanto, es mejor crear un esquema en el que los mismos se identifiquen por roles y se les otorguen las

autorizaciones adecuadas en función del tipo de trabajo que realizan.

- Aplicar el principio del mínimo privilegio es una de las mejores prácticas de seguridad en el control de accesos, consiste en otorgar únicamente los permisos que se necesitan explícitamente. Un caso muy común es el de asignar privilegios de administrador a un usuario que con un permiso menor puede desarrollar sus tareas con total normalidad.

3.6.2. Control y auditoría

- Auditar y corregir los accesos proactivamente y prestar especial atención a la actividad inusual para detectar desvíos en las definiciones de seguridad. Se debe lograr que el proceso de auditoría sea una actividad constante y obligatoria.
- Desasignar los accesos que ya no correspondan a empleados que cambiaron sus tareas o bien ya no pertenecen más a la organización.
- Prestar especial atención a los accesos de los usuarios administrador y con altos privilegios ya que su actividad será más sensible a nivel de seguridad.

3.6.3. Herramientas y automatización

- Utilizar las herramientas y utilitarios provistos por los sistemas operativos para automatizar, ejecutar y esquematizar los procesos de administración, control y auditoría, la implementación de los esquemas de control de accesos suelen ser extremadamente complejos debido a la gran cantidad y variedad de componentes que intervienen en los procesos.

4. Autorización

4.1.1. Definición

El concepto de autorización se encuentra directamente relacionado con el de control de accesos tratado en el capítulo anterior, y refiere, a los privilegios de los usuarios en la utilización de los recursos disponibles en un sistema. En este sentido, es necesario comprender los siguientes conceptos en forma general para luego abordar las implementaciones particulares de estos mecanismos relacionados al tópico de autorización en cada plataforma:

4.1.2. Token de acceso

Un *token* de acceso es un objeto o entidad que contiene toda la información de un contexto de seguridad particular para un proceso, subproceso o hilo [1].

4.1.3. Impersonación

La idea de impersonación o suplantación es un concepto de seguridad que se encuentra directamente relacionado con permitir que un equipo, usuario, servidor o aplicación, sea o actúe como si fuera otro [2]. Todo esto en términos de control de accesos o autorización a realizar determinadas tareas en un sistema operativo.

4.1.4. Privilegios y derechos de usuario

Un privilegio es el derecho o permiso de un sujeto (usuario o grupo) a realizar determinadas operaciones en el sistema operativo que afecten a un equipo o servidor [2].

Los privilegios difieren de los derechos de acceso en que estos últimos son aplicados a los objetos o recursos asegurables para establecer los mecanismos de control de accesos.

4.2. Autorización en Windows

4.2.1. Token de acceso de usuario

En Windows se utiliza una entidad denominada token de acceso, que consiste en un objeto protegido que contiene información sobre la identidad y los derechos/privilegios de un usuario [1,3,4,13].

Cuando un sujeto se autentica correctamente ante un equipo cuyo sistema operativo sea el de Microsoft, el proceso devuelve un SID para el usuario y una lista de SID para los grupos de los cuales es miembro. La autoridad de seguridad local utiliza esta información para generar el token antes mencionado al que adiciona todos los privilegios y derechos de usuario del sujeto. En concreto, un token de usuario en WIndows contiene la siguiente información:

- El identificador de seguridad (SID) para la cuenta del usuario.
- Los identificadores de seguridad de los grupos de los que es miembro el usuario.
- Un SID de inicio de sesión que identifica la sesión actual.
- Una lista de los privilegios que posee el usuario o sus grupos.
- El SID para el grupo primario del usuario.
- La lista de accesos discrecional (DACL) predeterminada que usa el sistema cuando el usuario crea un objeto.
- La fuente del token de acceso.
- El tipo de token: primario o de suplantación.
- Otra información y estadísticas.

Una vez generada la entidad mencionada, se adjunta una copia de la misma a todos los procesos y subprocesos que se ejecutan en nombre del usuario dueño, por ejemplo, un acceso a una carpeta o un archivo o la ejecución de un programa, entre otras.

En definitiva, cada vez que un proceso del sistema operativo interactúa con un objeto protegible o intenta realizar una tarea que requiera

privilegios específicos para un sujeto, Windows comprueba el token de acceso asociado al usuario para determinar si la autorización es positiva.

4.2.2. FIltro de tokens y UAC

Cuando un usuario administrador, es decir con privilegios elevados, inicia sesión en un equipo, el sistema operativo genera dos tokens de acceso diferente: uno de usuario estándar y uno de administrador. El primero contiene la misma información que el segundo, a excepción de los privilegios y los SID de administración de Windows.

El token de acceso de usuario estándar es utilizado para ejecutar aplicaciones que no realizan tareas administrativas, por ejemplo, el explorador de Windows. Como regla general, todo subproceso que herede de un proceso, ejecuta con el token de acceso (sin importar cuál sea) de su padre.

La tecnología de control de cuentas de usuario (UAC) implementada por Microsoft permite brindar mayor seguridad al darle un contexto de seguridad acotado a las funcionalidades de un usuario en la ejecución de procesos y aplicaciones, de esta forma, ayuda a prevenir la modificación de parámetros y configuraciones o instalaciones sobre el sistema, que muchas veces pueden ser involuntarias o malintencionadas (*malware*).

Cuando una aplicación necesita ejecutarse con más derechos que los de un usuario estándar, UAC permite restaurar los privilegios y SID adicionales pasando a utilizar el token de administrador, dando la posibilidad al sujeto de tener control explícito de las aplicaciones que están realizando cambios en la configuración de la plataforma. En concreto, al momento de requerir estos permisos adicionales, un usuario puede elevar privilegios aprobando la solicitud del sistema operativo.

El control de cuentas de usuario posee numerosas configuraciones para parametrizar su comportamiento, puede ejecutar bajo un escritorio seguro, solicitar o no las credenciales del usuario nuevamente al intentar elevar sus privilegios, entre otras [1,13].

4.2.3. Impersonación

La suplantación en Windows es la capacidad de un hilo para ejecutar ciertas operaciones utilizando un contexto de seguridad diferente al del proceso que posee el hilo [1,2,3]. Típicamente, un hilo en una aplicación que ejecuta en un servidor se hace pasar por un cliente en particular. La plataforma de Microsoft, entonces, utiliza el concepto de impersonación en su modelo de programación cliente/servidor. Esto permite que el hilo del servidor actúe en nombre de ese cliente para acceder a los objetos en el servidor o validar el acceso a los propios objetos del cliente.

Por ejemplo, una aplicación puede exportar ciertos recursos como archivos o bases de datos, entonces, los clientes que desean acceder al recurso envían una solicitud al servidor y este último es quien debe asegurarse que el cliente posea los privilegios necesarios para la actividad que está solicitando realizar. Es entonces, que el servidor actúa como si fuera el cliente, suplantándolo o impersonándolo y solicitando a quien contenga la información necesaria para validar esta autorización (por ejemplo, otro servidor que contenga el sistema de archivos donde se aloje el recurso en cuestión) que evalúe dicha solicitud a fin de otorgar o no el acceso.

De esta forma, el servidor a través de su token de impersonación puede reconocer si el cliente tiene el acceso necesario o no.

4.2.4. Derechos de usuario

Los derechos de usuario son los privilegios otorgados a una cuenta de usuario para realizar una operación concreta relacionada con el sistema en sí. Algunas de estas capacidades pueden ser apagar el equipo o servidor, cambiar la fecha y hora, acceder al editor de registro, entre otras. Los derechos en WIndows pueden ser tanto para otorgar un privilegio como para denegar el mismo.

Al detectar el inicio de sesión de un sujeto en un equipo, la autoridad de seguridad local (LSA) verifica y obtiene los derechos de dicho usuario y los adjunta al token correspondiente [1,3,4,13].

Por último, estos privilegios son asignados por un usuario administrador a través de la directiva de seguridad local accesible a través del comando “secpol.msc”.

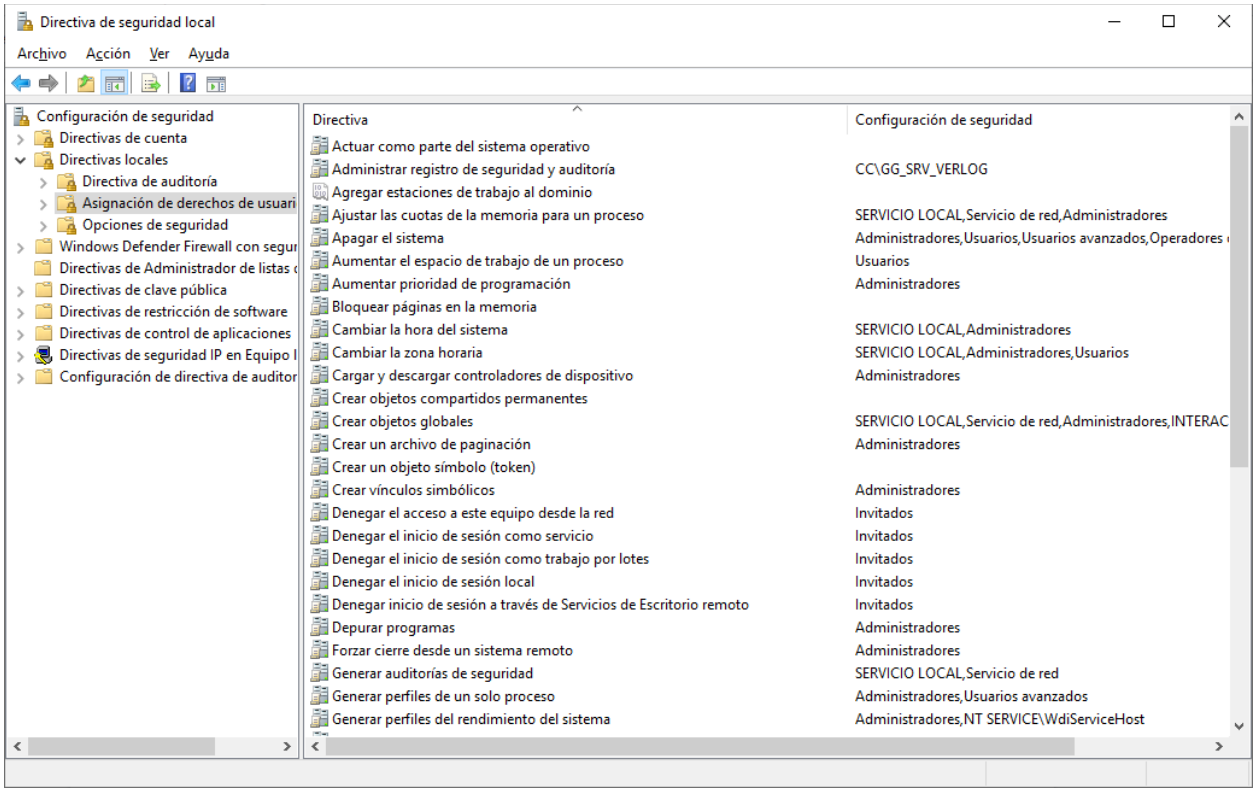

Imagen 11 - Configuración de derechos de usuario - Fuente: Elaboración Propia.

4.2.5. Políticas de grupo

Las políticas o directivas de grupo (GPO) son un concepto introducido por Microsoft para las plataformas Windows y se articulan a través del Active Directory [13].

Su principal objetivo es el de generalizar comportamientos para grupos de usuarios, equipos, y demás sujetos sin necesidad de ser configurados en forma individual. Son infinidad de configuraciones las que se pueden realizar

mediante las GPO y su utilidad en aspectos de seguridad es invaluable. Algunos de los conceptos considerados son:

- Directivas para la autenticación incluyendo factores, características de contraseña, bloqueos, etc.
- Directivas y configuraciones de logs y auditoría.
- Directivas de configuración de servicios y su administración.
- Directivas para el comportamiento de aplicaciones como navegadores o procesadores de texto.
- Directivas de configuraciones de sistema operativo, firewalls, panel de control, utilización de unidades, etc.
- Directivas para autorización, derechos de usuario y control de accesos a los recursos.

Prácticamente cualquier configuración parametrizable puede ser implementada por medio de directivas de grupo (GPO).

4.3. Autorización en Linux

4.3.1. Token de acceso

En Linux el concepto de token de acceso es abordado como una estructura de datos almacenados en memoria y relacionados directamente con un proceso o subproceso cada vez que el sistema operativo lo genera [2,5]. El token de Linux contiene los siguientes elementos:

- El UID para el usuario asociado.
- El GID para el grupo primario del usuario.
- Los GID de los grupos de los cuales el usuario es miembro.
- Una lista de los privilegios de usuario (relacionados con el control de accesos mandatorio).
- Una lista de control de accesos discrecional.

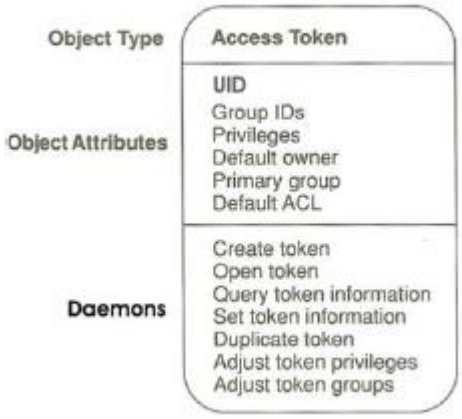

Imagen 12 - Token de acceso Linux - Fuente:[2].

4.3.2. Filtro de identificadores

En los sistemas Linux existen tres tipos de identificadores de usuario, el real, el efectivo y el guardado y los mismos son utilizados por la plataforma para la ejecución de procesos.

El identificador de usuario real es el correspondiente al dueño del proceso y define a qué objetos tienen permisos el mismo.

El identificador de usuario efectivo es comúnmente el mismo que el real, pero es utilizado para permitir que un usuario sin privilegios acceda a recursos que requieren mayores permisos, por ejemplo a los que pueden los usuarios administradores o el root.

El identificador de usuario guardado es utilizado cuando un proceso que ejecuta con privilegios elevados, por ejemplo un administrador o root, necesita realizar una tarea para la cual no se requieren tantos permisos, entonces cambia el uid efectivo por uno de un usuario estándar almacenando el valor del uid del usuario privilegiado en el uid guardado para más tarde restablecer el id efectivo.

Esta solución con los diferentes uid permite ejecutar las tareas con el mínimo privilegio necesario para los usuarios administradores y, a la vez,

elevar privilegios de un usuario estándar para funciones que requieran mayores permisos.

4.3.3. Impersonación

Como se mencionó anteriormente, Linux implementa un método que permite a los usuarios estándar ejecutar ciertas tareas como usuarios privilegiados o root, esto se realiza a través del comando sudo.

El programa sudo utiliza el concepto de setuid visto en el capítulo de control de accesos, /usr/bind/sudo contiene, en su sistema octal, permiso de ejecución para todos los usuarios y a la vez, el flag de setuid, esto permite que el identificador de usuario efectivo pase a ser el del usuario privilegiado (en la gran mayoría de los casos, el root).

La configuración del comando sudo se encuentra en el archivo /etc/sudoers y en el mismo se consigna: qué usuario o grupo, puede ejecutar ciertos comandos, como si fuera cierto usuario. Por ejemplo, el usuario de Juan, puede ejecutar el comando /usr/sbin/lpc como si fuera el usuario root; y, en ese caso, Juan anteponer la palabra sudo.

Una de las ventajas de la solución propuesta por Linux es la posibilidad de segregar la ejecución de ciertos comandos con mayores privilegios a determinados usuarios o grupos, esto lo hace un esquema flexible y seguro.

Por otra parte, se puede parametrizar el comportamiento del comando sudo, se pueden requerir las credenciales del usuario que ejecuta la elevación, la del usuario privilegiado o bien ninguna; se puede cachear la clave por cierto tiempo o solicitarla en cada intento o hasta dar una única posibilidad de escribir las credenciales correctamente.

Por último, toda la actividad ejecutada vía sudo, se almacena en el archivo /var/log/sudolog de forma de poder ser auditada.

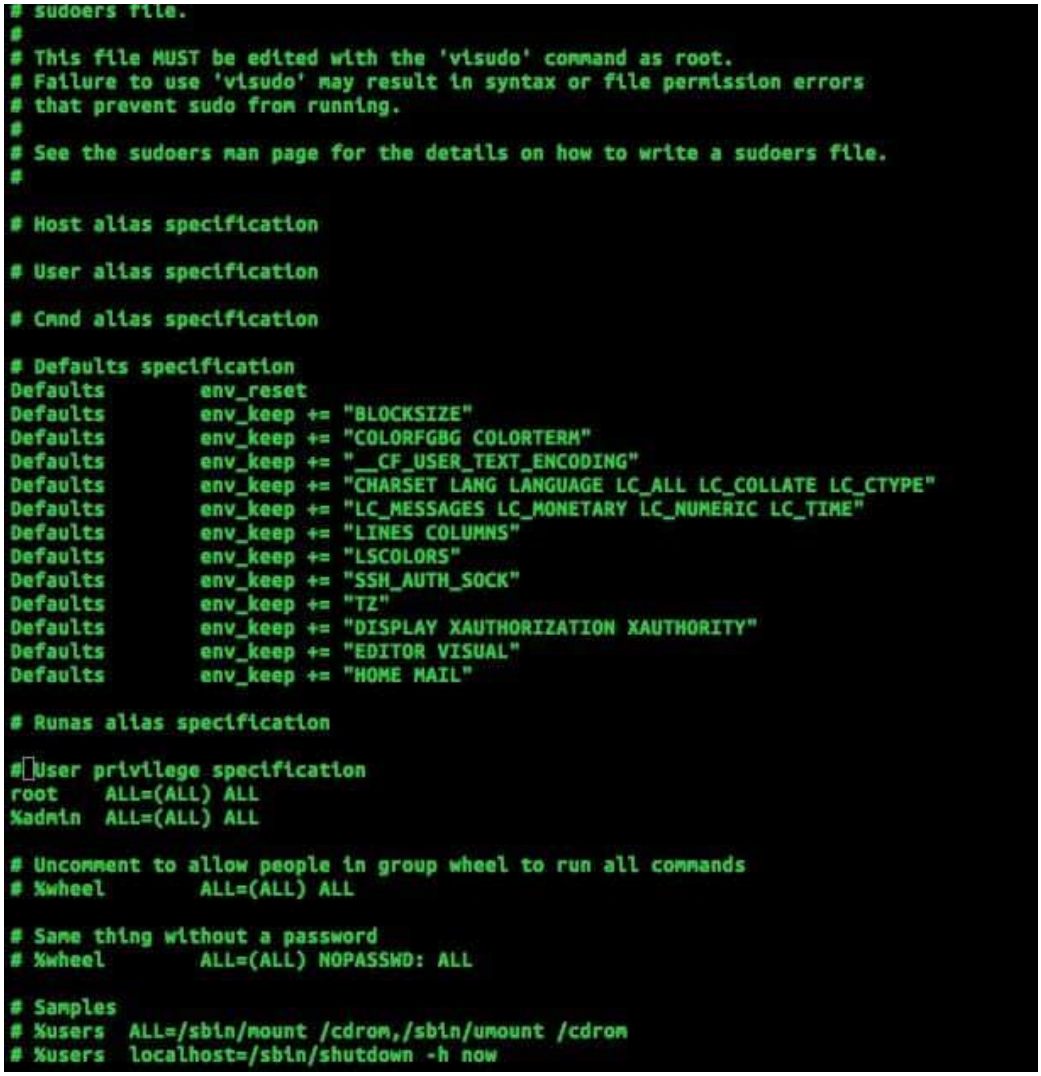

Imagen 13 - Configuración de archivo sudoers - Fuente: [16].

4.3.4. Privilegios

Linux implementa el concepto de privilegios o derechos de usuario a través del funcionamiento de su modelo de control de accesos mandatorio (MAC) [2,14,15], cuya definición se abordara en el capítulo anterior.

En resumen, los objetos del sistema son etiquetados de acuerdo a la sensibilidad de la información que contienen y en base a dicha catalogación es que los usuarios tienen autorización o no para realizar las distintas operaciones.

4.4. Comparativa Windows/Linux

Cuando se busca establecer una comparativa de los contenidos del capítulo, se puede ver que los conceptos implementados en ambas plataformas son los mismos (tokens de acceso, impersonación y privilegios y derechos), pero en este caso, las diferencias son muchas más que las similitudes y no sólo abarcan distintas implementaciones sino que ambos sistemas operativos adoptan enfoques bastante diferentes.

En lo que al término de token de acceso refiere, Windows, almacena las restricciones de acceso en dicha estructura, siendo que este Linux utiliza DAC y MAC para un proceso particular. Además, la plataforma de Microsoft guarda el tipo de token, ya sea primario o de impersonación en dicha entidad, en tanto que Linux lo deduce por el filtro de identificadores.

El diseño e implementación del concepto de impersonación es totalmente diferente en ambas plataformas, en Windows, un servidor puede suplantar su propio token por el de un cliente y decidir si el mismo tiene el acceso requerido o no. En cambio en Linux, un cliente ejecuta en el contexto de seguridad del servidor, lo que lleva a que, por más que la aplicación no tenga acceso, si el servidor lo tiene, entonces el cliente también.

Por último, Windows contiene un complejo pero muy parametrizable mecanismo para administrar los privilegios y derechos de usuario, para los cuales requiere la utilización de directivas de seguridad y la intervención de la LSA. En Linux no existen implementaciones similares sino que los privilegios son adoptados a través de la implementación del modelo de control de accesos mandatorio.

4.5. Buenas prácticas de seguridad

4.5.1. Control de cuentas de usuario [1,13]

- Habilitar el UAC para todos los equipos cliente de la organización.

- Hacer que todos los usuarios inicien sesión con cuentas con privilegios estándar.
- Generar dos cuentas para los administradores de dominio, una estándar y otra con privilegios administrativos, de forma de utilizar esta última únicamente cuando se requiera elevar privilegios.
- Si los administradores utilizan una única cuenta con privilegios administrativos, requerir el “modo de aprobación de administrador” para confirmar cada tarea.
- Capacitar a los usuarios con credenciales de administrador para que aprueben únicamente las solicitudes de elevación de las cuales conozcan el origen, caso contrario, puede tratarse de algún malware.
- Utilizar las configuraciones recomendadas por Microsoft y distribuirlas a través de políticas de grupo.

4.5.2. Configuraciones de sudo

- Prevenir la ejecución de comandos dentro de otros comandos, por ejemplo dentro del editor vim mediante la expresión ! o a través del comando find. Configurar la opción de sudo: no exec para solucionar estos inconvenientes [16].
- Manejar las variables de entorno para controlar su disponibilidad en comandos ejecutados vía sudo. Una buena opción es utilizar env_reset [16].
- Consignar únicamente comandos válidos en el archivo sudoers y además conteniendo la ruta (path) absoluta [16].

4.5.3. Directivas de grupo [1,13]

- Utilizar GPO granulares y con nombres descriptivos.
- Si bien es más relativo a la configuración del Active Directory o LDAP que a la definición de directivas, tener una buena estructura de árbol con unidades organizacionales bien definidas puede hacer

que la tarea de diseño e implementación de políticas sea mucho más sencilla.

- Verificar quiénes pueden editar las GPO y restringirlo a quiénes corresponda.

5. Servicios y demonios

5.1.1. Definición

Los servicios (Windows) y demonios (antiguamente denominados en Linux) son un componente central de los sistemas operativos dado que permiten la creación y administración de procesos de larga duración.

A diferencia las aplicaciones, que son generalmente iniciadas por un usuario final, los servicios pueden iniciarse sin su intervención y continúan ejecutándose aunque ningún usuario tenga una sesión activa. Los servicios y demonios corren en segundo plano y regularmente se configuran para activarse cuando el equipo inicie el sistema operativo aunque también pueden ejecutarse manualmente o ante determinado evento. Otra característica de los servicios y demonios es que no poseen interfaz gráfica.

A través de la utilización de los servicios, pueden gestionarse una gran variedad de funciones del sistema como conexiones de red, manejo de dispositivos de entrada/salida, copias de seguridad y respaldo, etc. Es común que los desarrolladores utilicen estos componentes para ejecutar determinadas tareas sin la intervención del usuario final.

5.1.2. Propiedades de los servicios

Durante las siguientes subsecciones se buscará explicar brevemente las propiedades más relevantes de los servicios, para comprender sobre qué aspectos deben implementarse medidas de seguridad.

Si bien los atributos descritos a continuación pueden variar levemente entre ambas plataformas de estudio, los esquemas utilizados son muy similares.

- Tipo: Los servicios pueden ser de tipo manual: iniciados por un usuario, automáticos: se ejecutan al iniciar el sistema operativo, o programados: ante algún evento o bien en un momento específico.

- Estado: Los servicios pueden encontrarse habilitados o deshabilitados, además su estado puede ser: iniciado o no iniciado. En ambas plataformas se puede interactuar con los servicios de modo de ir modificando el estado de los mismos e incluso reiniciarlos en caso que sea necesario.
- Usuarios: Los servicios corren bajo una cuenta de usuario que debe poseer los privilegios para poder iniciarlo. En este punto, los permisos necesarios para que los usuarios de servicio puedan ejecutarlos, varían según cada plataforma y se verán con mayor detalle más adelante.
- Proceso: El sistema operativo asigna un identificador de proceso a cada servicio activo, a través del mismo, la plataforma administra estos recursos.
- Archivo: Todo servicio tiene un archivo asociado que contiene la lógica del servicio y permite iniciarlo a través de su ejecución. En Windows, los archivos asociados pueden ser .exe o .dll. El usuario asociado al servicio debe poseer privilegios de ejecución al fichero correspondiente.

5.2. Servicios en Windows

5.2.1. Usuarios de servicio

Por defecto, los servicios se ejecutan en un contexto de seguridad diferente al del usuario conectado, en particular, en el de las cuentas predeterminadas Sistema Local, Servicio Local o Servicio de Red; sin embargo, este comportamiento puede ser modificado [17,18].

La propiedad de usuario de servicio o cuenta, permite configurar el servicio utilizando las siguientes alternativas [17]:

- Sistema Local (*LocalSystem*), que se ejecuta en el contexto de una cuenta que proporciona amplios privilegios locales y presenta las

credenciales de la computadora a cualquier servidor remoto. En su token posee los permisos de usuario administrador y gran cantidad de derechos de usuario.

- Servicio Local (*LocalService*), que se ejecuta en el contexto de una cuenta que actúa como un usuario sin privilegios en la computadora local y presenta credenciales anónimas a cualquier servidor remoto. Posee una cantidad mucho más acotada de derechos de usuario en comparación con la cuenta Sistema Local.
- Servicio de Red (*NetworkService*), que se ejecuta en el contexto de una cuenta que actúa como un usuario sin privilegios en la computadora local, y presenta las credenciales de la computadora a cualquier servidor remoto. Posee una cantidad mucho más acotada de derechos de usuario en comparación con la cuenta Sistema Local.
- Usuario personalizado, que hace que el sistema solicite un nombre de usuario y contraseña válidos cuando el servicio se instala y se ejecuta en el contexto de esta cuenta específica. Este usuario debe poseer el derecho de usuario de iniciar sesión como servicio.

5.2.2. Service Control Manager

El *Service Control Manager* (SCM), es el proceso administrador de todos los servicios Windows de un equipo, a través del mismo, se puede detener, pausar, iniciar, retrasar el inicio o reanudar cada servicio según corresponda, también puede modificar el mecanismo de inicio (manual o automático) o especificar un usuario con el cual se ejecute el mismo [3,17,18].

El SCM, en concreto, permite administrar las propiedades de los servicios vistas en la sección anterior: nombre, descripción, estado, tipo y usuario entre otros.

Windows asigna a cada servicio un identificador de seguridad (SID), por lo tanto son considerados sujetos o principals, esto implica que a los

mismos pueden aplicarse los mecanismos de control de accesos vistos en capítulos anteriores.

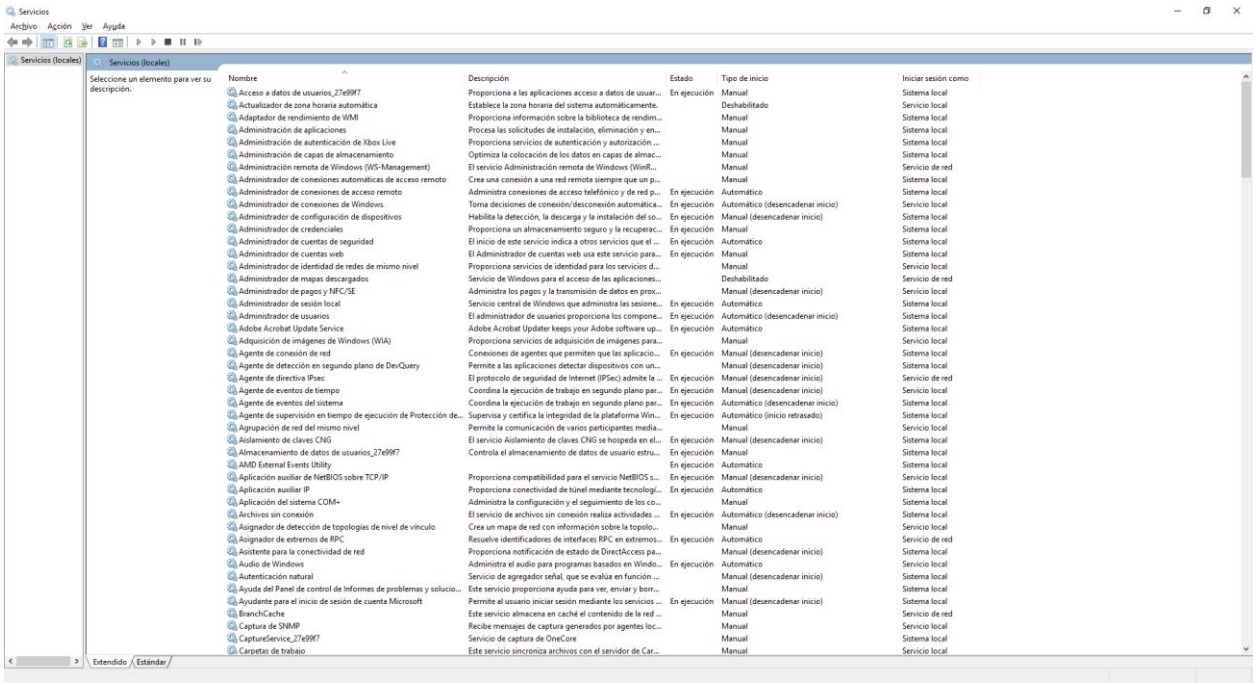

Imagen 14 - Consola de administración de servicios (SCM) - Fuente: Elaboración propia.

5.2.3. Service handler process

Los manejadores de procesos de servicios (*service handler process*), mayormente conocidos con el nombre se svchost, son los encargados de gestionar la ejecución de un grupo lógico de servicios [3,17,18].

Cada manejador posee un token de seguridad con la lista de permisos y privilegios que contiene. El SCM realiza una evaluación de los privilegios de ejecución que requiere cada servicio de forma de agruparlos para posteriormente asignar ese grupo a un manejador definiendo su token de seguridad. Por ejemplo, existe un svchost para servicios de red, otro para servicios locales, otro para copias de seguridad, etc.

La ventaja de este esquema es que permite segmentar riesgos, ante una eventual caída de un manejador, sólo se verán afectados los servicios asociados y no todos los existentes en ese equipo.

5.3. Servicios en Linux

5.3.1. Rc.d

El directorio /etc/rc.d contiene una serie de comandos que se ejecutan al inicio del sistema, los mismos, se encuentran organizados en archivos y subdirectorios de acuerdo a las tareas que realicen.

El subdirectorio /etc/rc.d/init.d contiene la información de los scripts para correr los distintos servicios del sistema y de red.

El subdirectorio /etc/rc.d/rc.d se utiliza para los niveles de ejecución (*run levels*) y vínculos a los archivos del directorio /etc/rc.d/init.d [5]. Estos niveles especifican los distintos modos de ejecución de los servicios de la plataforma.

File	Description
/etc/sysconfig	Directory that holds system configuration files and directories.
/etc/rc.d	Directory that holds system startup and shutdown files.
/etc/rc.d/ rc.sysinit	Initialization file for your system.
/etc/rc.d/rc.local	Initialization file for your own commands; you can freely edit this file to add your own startup commands; this is the last startup file executed.
/etc/rc.d/init.d	Directory that holds network scripts to start up network connections.
/etc/rc.d/rc*num*.d	Directories for different runlevels, where *num* is the runlevel. The directories hold links to scripts in the **/etc/rc.d/init.d** directory.
/etc/rc.d/init.d	Directory that holds system service scripts.
/etc/rc.d/ init.d/halt	Operations performed each time you shut down the system, such as unmounting file systems; called rc.halt in other distributions.

Tabla 1 - Archivos de inicialización Linux - Fuente: [5].

Runlevel	rc.d Directory	Description
0	**rc0.d**	Halt (shut down) the system
1	**rc1.d**	Single-user mode (no networking, limited capabilities)
2	**rc2.d**	Multiuser mode with no NFS support (limited capabilities)
3	**rc3.d**	Multiuser mode (full operational mode)
4	**rc4.d**	User-defined, implemented by default on Red Hat and Fedora as the same as runlevel 3, multiuser mode
5	**rc5.d**	Multiuser mode with graphical login (full operation mode with graphical login added)
6	**rc6.d**	Reboot system

Tabla 2 - Niveles de ejecución Linux - Fuente: [5].

5.3.2. Init.d

El directorio /etc/rc.d/init.d contiene los scripts que inician y detienen los diferentes demonios específicos, por ejemplo los servicios web, dispositivos de entrada salida como impresoras o la configuración de fecha y hora. Estos archivos ya se encuentran predefinidos por la plataforma y modificarlos implica un riesgo si no se posee el conocimiento necesario.

Por defecto, los servicios alojados en este directorio, inician cuando el sistema operativo arranca y se detienen cuando el mismo se apaga. Este comportamiento, sin embargo, puede personalizarse de acuerdo a las necesidades que se establezcan.

La herramienta *system-config-services* muestra una lista de servicios disponibles, que permite elegir qué demonios se desean iniciar o evitar que se inicien. El comando chkconfig utiliza las opciones de encendido y apagado para seleccionar o no los servicios para el inicio en el arranque del sistema.

Asimismo, los servicios alojados en init.d pueden ser iniciados, reiniciados o detenidos en forma manual en cualquier momento, a través de system-config-services o del comando service [5].

5.3.3. Xinetd

Los demonios ejecutados con bajo init.d se mantienen activos desde el inicio hasta el apagado del sistema, o bien deben gestionarse manualmente. Esta implementación tiene algunas contras en cuanto al consumo de recursos del servidor debido a que los servicios se encuentran constantemente corriendo cuando su verdadero consumo puede no requerirlo.

Como alternativa, Linux implementa el demonio xinetd (*Extended Internet Services Daemon*) que permite invocar un servicio únicamente cuando el servidor recibe una solicitud que demande su consumo [5]. De esta forma, este demonio verifica continuamente cualquier solicitud de usuarios remotos para un servicio de Internet en particular; cuando recibe una solicitud, inicia el servicio del servidor que corresponda.

Xinetd posee algunas medidas de seguridad adicionales a su antecesor inetd, las mismas se definen en el archivo xinetd.conf donde pueden configurarse,por ejemplo, parámetros para el registro de eventos o usuarios y grupos para el consumo de los servicios.

Attribute	Description
ids	Identifies a service. By default, the service ID is the same as the service name.
type	Type of service: RPC, INTERNAL (provided by xinetd), or UNLISTED (not listed in a standard system file).
flags	Possible flags include REUSE, INTERCEPT, NORETRY, IDONLY, NAMEINARGS (allows use of tcpd), NODELAY, and DISABLE (disable the service). See the xinetd.conf Main page for more details.
disable	Specify yes to disable the service.
socket_type	Specify stream for a stream-based service, dgram for a datagram-based service, raw for a service that requires direct access to IP, and seqpacket for reliable sequential datagram transmission.
protocol	Specifies a protocol for the service. The protocol must exist in /etc/protocols. If this attribute is not defined, the default protocol employed by the service will be used.
wait	Specifies whether the service is single-threaded or multithreaded (yes or no). If yes, the service is single-threaded, which means that xinetd will start the server and then stop handling requests for the service until the server stops. If no, the service is multithreaded and xinetd will continue to handle new requests for it.
user	Specifies the user ID (UID) for the server process. The user name must exist in /etc/passwd.
group	Specifies the GID for the server process. The group name must exist in /etc/group.
instances	Specifies the number of server processes that can be simultaneously active for a service.
nice	Specifies the server priority.
server	Specifies the program to execute for this service.
server_args	Lists the arguments passed to the server. This does not include the server name.
only_from	Controls the remote hosts to which the particular service is available. Its value is a list of IP addresses. With no value, service is denied to all remote hosts.
no_access	Controls the remote hosts to which the particular service is unavailable.
access_times	Specifies the time intervals when the service is available. An interval has the form hour:min-hour:min.
log_type	Specifies where the output of the service log is sent, either the syslog facility (SYSLOG) or a file (FILE).

Attribute	Description
log_on_ success	Specifies the information that is logged when a server starts and stops. Information you can specify includes PID (server process ID), HOST (the remote host address), USERID (the remote user), EXIT (exit status and termination signal), and DURATION (duration of a service session).
log_on_ failure	Specifies the information that is logged when a server cannot be started. Information you can specify includes HOST (the remote host address), USERID (user ID of the remote user), ATTEMPT (logs a failed attempt), and RECORD (records information from the remote host to allow monitoring of attempts to access the server).
rpc_version	Specifies the RPC version for a RPC service.
rpc_number	Specifies the number for an UNLISTED RPC service.
env	Defines environment variables for a service.
passenv	The list of environment variables from xinetd's environment that will be passed to the server.
port	Specifies the service port.
redirect	Allows a TCP service to be redirected to another host.
bind	Allows a service to be bound to a specific interface on the machine.
interface	Synonym for bind.
banner	The name of a file to be displayed for a remote host when a connection to that service is established.
banner_ success	The name of a file to be displayed at the remote host when a connection to that service is granted.
banner_fail	The name of a file to be displayed at the remote host when a connection to that service is denied.
groups	Allows access to groups the service has access to (yes or no).
enabled	Specifies the list of service names to enable.
include	Inserts the contents of a specified file as part of the configuration file.
includedir	Takes a directory name in the form of includedir /etc/xinetd.d. Every file inside that directory will be read sequentially as an xinetd configuration file, combining to form the xinetd configuration.

Tabla 3 - Atributos de configuración de xinetd - Fuente: [5].

5.3.4. TCP Wrappers

Los contenedores TCP (*TCP Wrappers*) agregan otro nivel de seguridad a los servidores administrados por xinetd [5]. En efecto, el servidor está envuelto con un nivel intermedio de seguridad, monitoreo de conexiones y control de acceso. Se supervisa una conexión de servidor realizada a través de xinetd, verificando la identidad del usuario remoto y distintas comprobaciones para asegurarse que las solicitudes son válidas.

La forma de administrar los permisos o prohibiciones de consumo, es a través de dos archivos: *hosts.allow* (permitidos) y *hosts.deny* (prohibidos). El formato de los mismos contempla los datos de servicio al que se desea acceder/denegar contra dirección ip o nombre del dominio.

De esta forma, los contenedores TCP hacen las veces de filtro ante cada conexión entrante que requiera la intervención del demonio xinetd.

5.4. Comparativa Windows/Linux

Los conceptos abordados en el capítulo parten de una premisa, los términos servicio y demonio son prácticamente sinónimos. En esta línea de pensamiento, ambas plataformas abordan este tema de formas similares en lo que un esquema general refiere.

Como se ha visto anteriormente, donde mayores diferencias se observan es en las particularidades de cada implementación, aunque siguiendo la misma base conceptual, esto incide en que no se permiten visualizar ventajas claras de uno sobre el otro.

Por último, es importante destacar que Linux orienta sus medidas de seguridad en demonios a los provistos como servidor web a través de la implementación de xinetd, en tanto que Windows considera igualmente a los servicios locales contemplando usuarios por defecto según la naturaleza del servicio.

5.5. Buenas prácticas de seguridad

5.5.1. Gestión de servicios

- Ejecutar cada servicio en un contexto de seguridad específico, es decir cada uno con su propio usuario, de forma de reducir el impacto sobre el sistema en caso de error o explotación del mismo.
- Aplicar el principio de mínimo privilegio para los usuarios de servicio.
- Deshabilitar todos los servicios que no sean necesarios según las funcionalidades del servidor o del equipo de forma de reducir la superficie de ataque.

6. Logs y auditoría

6.1.1. Definición

Los *logs* son eventos generados por alguna fuente, en este caso los sistemas operativos, que permiten registrar la actividad que se lleva a cabo en los mismos.

Se llama auditoría al proceso de configuración y registración de los eventos para su posterior análisis y acción. Comúnmente, se definen reglas para cada uno de estos componentes de acuerdo al objetivo del proceso.

Los logs constituyen una evidencia de la existencia de cierta actividad, por lo tanto, se debe garantizar la confidencialidad, integridad y disponibilidad de los mismos. En este sentido, es necesario mencionar el término de logs seguros.

Un log seguro implementa mecanismos de seguridad que garanticen el cumplimiento de los tres pilares mencionados, por ejemplo, a través de una función de hash o del control de accesos restringido a la información contenida.

6.1.2. Propiedades de los logs

Si bien las propiedades descritas a continuación pueden variar levemente entre ambas plataformas de estudio, las más importantes a nivel general son:

- Categoría o tipo: Los logs se encuentran tipificados según su origen o fuente, por ejemplo: pueden provenir de una aplicación, del sistema operativo, de eventos de seguridad, etc.
- Información de los eventos: Cada evento registrado debe poseer como mínimo información sobre: instante o tiempo de ocurrencia, un identificador, tipo o categoría, fuente que lo originó, criticidad, usuario o grupo que lo originó, un texto descriptivo de la actividad.

- Rotación de logs: Este concepto contempla las decisiones que se deben tomar acerca de la definición del tamaño máximo del archivo de log, qué hacer cuando se llega a ese tamaño (por ejemplo: comprimir el archivo, almacenarlo y usar uno nuevo o sobreescribirlo como un log circular) y cuánto tiempo almacenar los registros antes de depurarlos.

6.2. Logs en Windows

La plataforma de Microsoft almacena el registro de eventos en forma centralizada en la ubicación %SystemRoot%\system32\config.

Windows clasifica cada evento con un nivel de criticidad, siendo los mismos, en orden de menor a mayor gravedad: información, advertencia, error y crítico.

Los mensajes pueden categorizarse según sus características en: logs de aplicación, de seguridad, de sistema, de configuración, o bien que re-enviados de otros equipos o fuentes [19,20].

Los eventos de seguridad, objeto de estudio en el presente, almacenan la información en esta materia, y los eventos típicos incluyen intentos de inicio de sesión, control de accesos a recursos, etc.

El detalle de qué actividad es registrada en el log de seguridad se configura en las directivas de auditoría definidas en el equipo.

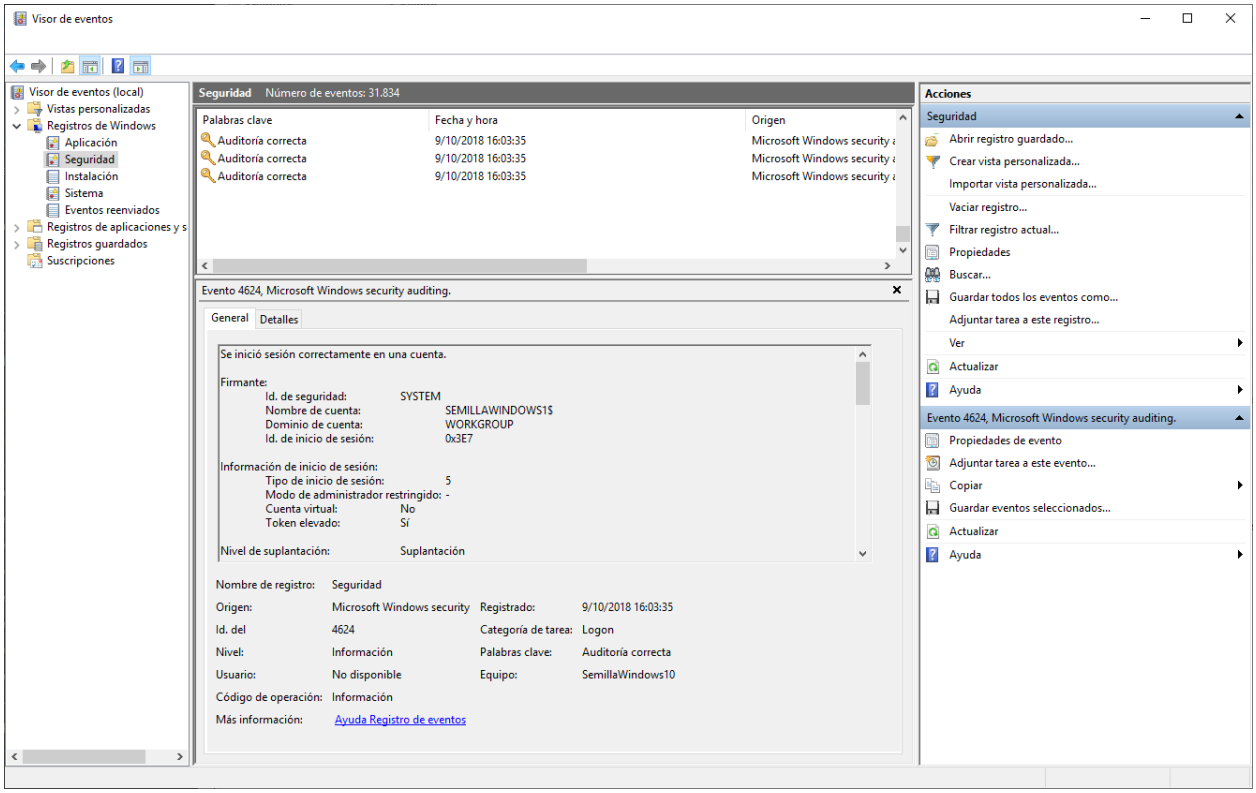

Imagen 15 - Consola de visor de eventos de Windows - Fuente: Elaboración propia.

6.2.1. Directivas de auditoría de seguridad

A través de las directivas de auditoría se pueden establecer reglas que permitan definir qué eventos y actividades relativas a la seguridad deben ser registradas [3,4], además, permite dar detalle de, según el resultado de la actividad (correcto o fallido), auditar o no la acción. Por ejemplo, en algún caso, puede ser interesante registrar una autenticación correcta pero no una fallida.

Estas configuraciones, corresponden a cada equipo o servidor con Windows y si bien pueden realizarse individualmente a través de la directiva de seguridad local, también es posible hacerlo mediante políticas de grupo.

Si bien la auditoría de nivel básico otorga algunos tipos de eventos a registrar, el nivel avanzado provee mayor granularidad en este sentido, siendo las principales categorías:

- Inicios/cierres de sesión de cuentas: Permite documentar los intentos de autenticación tanto para usuarios locales como de dominio.
- Administración de cuentas: Permite auditar la actividad de alta, baja y modificación de cuentas de usuarios, grupos y equipos.
- Accesos DS: Permite registrar eventos de modificación sobre los servicios de directorio.
- Accesos a objetos: Permite auditar los eventos relativos al control de accesos a los recursos.
- Cambios de directivas: Permite documentar los cambios a las directivas de auditoría.

6.3. Logs en Linux

Linux posee una gestión de logs muy completa que permite registrar casi todos los eventos que se producen en el sistema proporcionando un repositorio centralizado ubicado en /var/log [6].

Los mensajes pueden categorizarse según sus características en: logs de aplicación, de eventos, de servicio o de sistema.

Para el presente desarrollo se hará foco en estos últimos debido a que en los mismos se pueden encontrar los eventos de seguridad de la plataforma.

Los logs de sistema (system logs) se recolectan mediante la utilización de un demonio llamado syslogd. Todos los programas, aplicaciones, etc, envían la información generada de acuerdo a las reglas definidas en el archivo de configuración ubicado en /etc/syslog.conf.

Para comprender el funcionamiento del syslog, es necesario introducir cuatro términos claves: *facility*, *priority*, *message selector* y *action*.

Para la conformación de reglas de logueo, se establecen combinaciones de estos atributos según la necesidad, por ejemplo, los eventos críticos de autenticación se envían al archivo /var/log/auth.log.

```
*.err;kern.debug;auth.notice /dev/console
daemon,auth.notice           /var/log/messages
lpr.info                     /var/log/lpr.log
mail.*                       /var/log/mail.log
ftp.*                        /var/log/ftp.log
auth.*                       @prep.ai.mit.edu
auth.*                       root,amrood
netinfo.err                  /var/log/netinfo.log
install.*                    /var/log/install.log
*.emerg                      *
*.alert                      |program_name
mark.*                       /dev/console
```

Imagen 16 - Ejemplo de configuración de /etc/syslog.conf - Fuente: [6].

6.3.1. Facility

El concepto de facility se encuentra directamente relacionado con el identificador del programa o aplicación que produce el evento que desea registrarse.

Si bien existe una buena variedad de facilities, la más interesante desde el punto de vista de seguridad es “secure” (también conocida como auth en otras distribuciones unix). La misma, considera eventos como inicios de sesión por ssh, utilización de comandos sudo, otros eventos relacionados a procesos de autorización, intentos de inicio de sesión fallidos, etc.

Facility	Description
auth	Activity related to requesting name and password (getty, su, login)
authpriv	Same as auth but logged to a file that can only be read by selected users
console	Used to capture messages that are generally directed to the system console
cron	Messages from the cron system scheduler
daemon	System daemon catch-all
ftp	Messages relating to the ftp daemon

kern	Kernel messages
local0.local7	Local facilities defined per site
lpr	Messages from the line printing system
mail	Messages relating to the mail system
mark	Pseudo-event used to generate timestamps in log files
news	Messages relating to network news protocol (nntp)
ntp	Messages relating to network time protocol
user	Regular user processes
uucp	UUCP subsystem

Tabla 4 - Facilities disponibles en Linux - Fuente: [6].

6.3.2. Priority

El término priority refiere a la importancia o severidad de un mensaje y se divide en niveles para facilitar su categorización. Entre los más severos se pueden encontrar: emerg, alert, crit y error y entre los menos graves: info o debug.

Priority	Description
emerg	Emergency condition, such as an imminent system crash, usually broadcast to all users
alert	Condition that should be corrected immediately, such as a corrupted system database
crit	Critical condition, such as a hardware error
err	Ordinary error
warning	Warning message
notice	Condition that is not an error, but possibly should be handled in a special way
info	Informational message
debug	Messages that are used when debugging programs
none	Pseudo level used to specify not to log messages

Tabla 5 - Priorities disponibles en Linux - Fuente: [6].

6.3.3. Message selector

Un selector de mensaje surge de la combinación de dos conceptos vistos anteriormente: facility y priority. Además, incorpora la posibilidad de utilizar calificadores de forma de otorgar mayor ductilidad a las reglas de registro que se definan.

Por ejemplo, algunos selectores posibles son: todos los mensajes que provengan del log secure (secure.*) o bien los eventos de error del kernel (kernel.error).

6.3.4. Action

Por último, el término action refiere al tratamiento que el sistema operativo debe dar a los eventos que se correspondan con los selectores definidos.

Posibles acciones son: almacenar en el archivo /var/logs/secure.log, enviar al home de un usuario en particular, por correo electrónico, etc.

6.4. Comparativa Windows/Linux

Se evidencia que ambos sistemas de estudio implementan los conceptos de logs y auditoría de eventos y que sus esquemas son similares desde el punto de vista de los indicadores presentes en cada suceso a registrar, por ejemplo: los dos consideran la criticidad, la categoría, la fuente originante y la información relevante como propiedades de valor.

La principal diferencia se observa en la cobertura de eventos de accesos a objetos y recursos como archivos o directorios dado que Windows implementa este concepto a través de la lista de control de accesos de auditoría (SACL) y Linux no lo considera.

6.5. Buenas prácticas de seguridad

6.5.1. Gestión de logs, monitoreo y control

- Otorgar privilegios de lectura a los distintos logs únicamente a los usuarios o grupos que lo requieran.
- Establecer controles que permitan detectar los cambios en configuraciones de auditoría y eliminación de registros.
- Establecer controles y alertas que permitan detectar eventos anómalos y poco comunes de forma de establecer un plan de remediación.
- Implementar mecanismos para proveer la centralización y correlación de eventos a través de un SIEM.

7. Conclusiones

Está comprobada la importancia de los sistemas operativos en la infraestructura tecnológica de una organización. Como se ha mencionado en el cuerpo introductorio de este trabajo, constituyen el soporte sobre el que funcionan aplicaciones, servidores web y bases de datos, entre otros. Es entonces que durante el desarrollo de este documento se buscó responder a la pregunta: ¿Son seguros los sistemas operativos?

La respuesta es un tanto compleja y no puede reducirse a un par de sentencias que lo afirmen o denieguen. ¿La seguridad es una realidad o una sensación? ¿Podemos decir que algo es totalmente seguro?

La motivación de un especialista en la materia, será entonces, acercar esta visión por medio de la utilización de planes que contemplen herramientas, mecanismos e implementaciones comprobables y controlables en la disciplina.

En el ámbito de seguridad, los términos: usuarios, grupos, autenticación, autorización, control de accesos o auditoría, son bien conocidos. La elección de estos principios de estudio fue correcta dado que se identificaron como conceptos en común en ambas plataformas de estudio y esto fue comprobado a través del análisis de las decisiones adoptadas por cada sistema en pos de resolver los desafíos de seguridad que estos tópicos implican.

Detallar las implementaciones y consideraciones de cada sistema operativo y luego buscar establecer una comparativa entre ambas, aunque ésta sin el fin de valorar alguna de ellas por sobre la otra, logró cumplir la meta de proponer al lector ciertos puntos de relación, similitud o diferenciación que faciliten el entendimiento de los procesos involucrados.

La sección de buenas prácticas consignada al final de cada capítulo es vital si el lector tiene que comenzar a delinear una estrategia de seguridad en sistemas operativos. Este apartado cumplió con su finalidad dado

que permite visualizar qué puntos requieren mayor énfasis y cómo se pueden prevenir, mitigar y evitar posibles brechas de seguridad con medidas concretas.

Es importante destacar, que si bien no fue un objetivo trazado al principio del desarrollo de este documento, luego de exponer los contenidos abordados, se puede visualizar que, dada la complejidad de los mismos, resulta inevitablemente necesario el rol de un experto en seguridad de la información con el fin de dar tratamiento a estos aspectos.

Desde el punto de vista de los sistemas estudiados, puede inferirse que los mismos consideran a la seguridad como un factor importante en su funcionamiento, es así que se ha visto a lo largo del desarrollo de este documento la gran variedad de mecanismos y consideraciones que las plataformas realizan sobre estas cuestiones que se eligieron como pilares para abordar la temática.

Por último, se concluye que los objetivos planteados en el inicio fueron resueltos exitosamente. Este trabajo, cumple con la finalidad de ser una introducción a los conceptos de seguridad en sistemas operativos a la vez que representa una base sólida sobre la cual construir o mejorar los esquemas de una organización en esta materia.

8. Bibliografía

[1] Microsoft Windows Security: Identity and access management, https://docs.microsoft.com/en-us/windows/security/identity-protection/ (consultado el 12/05/2019)

[2] Youssef Bassil, “Windows and Linux operating systems from a security perspective”. In: Journal of Global Research in Computer Science, ISSN 2229-371X, Vol. 3, No. 2, 2012.

[3] Hester M. y Henley C., “Microsoft Windows Server 2012 Administration Instant Reference”, Indianapolis, Indiana, 2013.

[4] Orchilles J., “Microsoft Windows 7 Administrator's Reference”, Cap. 7 y 8, 2010.

[5] Richard Petersen, “Fedora 7 & Red Hat Enterprise Linux: The complete reference”, 2007.

[6] Tutoriales Point: Unix/Linux Tutorial, https://www.tutorialspoint.com/unix/ (consultado el 19/05/2019)

[7] Microsoft Windows Authentication, https://docs.microsoft.com/en-us/windows/win32/secauthn/authentication-portal (consultado el 01/06/2019)

[8] Kerberos Authentication Explained, https://www.varonis.com/blog/kerberos-authentication-explained/ (consultado el 01/06/2019)

[9] Pacific Rim Computer Products, “Linux Authentication Systems”, http://www.linuxgeek.net/documentation/authentication.phtml, 2003. (consultado el 22/06/2019)

[10] Antonio Lopez (INCIBE), “Mecanismos básicos de control de acceso”, https://www.incibe-cert.es/blog/control-acceso (consultado el 14/07/2019)

[11] Bernhard Mehl, “Six best practices to follow in access control”, https://www.helpnetsecurity.com/2018/07/31/access-control-best-practices/ (consultado el 20/08/2019)

[12] Mathew Schwartz, “Access Control: 10 Best Practices” https://esj.com/articles/2007/03/27/access-control-10-best-practices.aspx (consultado el 20/08/2019)

[13] Windows Security: Threat Protection - More Windows 10 Security,

https://docs.microsoft.com/en-us/windows/security/threat-protection/ (consultado el 01/06/2019)

[14] Red Hat Enterprise Linux: Security Guide https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/security_guide/index (consultado el 16/05/2019)

[15] Red Hat Enterprise Linux: SELinux User's and Administrator's guide, https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/selinux_users_and_administrators_guide/index (consultado el 16/05/2019)

[16] Leonardo Neves Bernardo, BSD Magazine Article, “Best Practices in UNIX Access Control with SUDO”, https://www.ixsystems.com/blog/best-practices-in-unix-access-control-with-sudo/ (consultado el 20/08/2019)

[17] Stackify: What are Windows Services? How Windows Services Work, Examples, Tutorials and More, https://stackify.com/what-are-windows-services/ (consultado el 15/09/2019)

[18] Derek Melber, “Securing Windows Service Accounts” (Partes 1 y 2), http://techgenix.com/securing-windows-service-accounts-part1/ y http://techgenix.com/securing-windows-service-accounts-part2/ (consultados el 15/09/2019)

[19] Solarwinds Loggly: Ultimate Guide: Windows, https://www.loggly.com/ultimate-guide/windows-logging-basics/ (consultado el 10/10/2019)

[20] Ultimate Windows Security, Windows Security Settings, https://www.ultimatewindowssecurity.com/wiki/page.aspx?spid=SecuritySettings (consultado el 02/08/2019)

[21] Microsoft Security Guidance Blog, https://blogs.technet.microsoft.com/secguide/ (consultado el 12/05/2019)

[22] Tutoriales Point: Learn Windows Server 2012, https://www.tutorialspoint.com/windows_server_2012/index.htm (consultado el 01/06/2019)

[23] NTLM user authentication in Windows, https://support.microsoft.com/en-us/help/102716/ntlm-user-authentication-in-windows (consultado el 01/06/2019)

[24] Diego Geovanny Ordóñez Bazarán, “Diseño, análisis, pruebas e implementación de un esquema de seguridad para la plataforma Windows”, http://dspace.utpl.edu.ec/bitstream/123456789/1476/3/Utpl_Ordo%C3%B1ez_Bazaran_Diego_005x1049.pdf, 2008.

[25] Martin Prpič, Tomáš Čapek, Stephen Wadeley, Yoana Ruseva, Miroslav Svoboda and Robert Krátký, “Red Hat Enterprise Linux 7 Security Guide”, 2013.

[26] Fenzi K. y Wresky D., “Linux Security How To”, https://scrye.com//~kevin/lsh/Security-HOWTO.pdf, 2004.

[27] Real, Effective and saved user id in Linux, https://www.geeksforgeeks.org/real-effective-and-saved-userid-in-linux/ (consultado el 14/08/2019)

[28] Hannifin D., Alpern N., Alpern J., Tiensivu A., “Microsoft Windows Server 2008 R2 Administrator's Reference”, 2010.

[29] Daniel J. Barrett, “Linux Pocket Guide”, 2012.

[30] Constantinescu R. y Zota Razvan D., “Issues of Operating Systems Security” In: ECAI 2007 - International Conference – Second Edition: Electronics, Computers and Artificial Intelligence, 2007.

[31] Sandra Milena Tibocha Roa: “Guía de aseguramiento de servidores Microsoft Versión Windows Server 2008 R2”, https://es.scribd.com/document/290901082/Guia-de-Aseguramiento-Microsoft-Windows-Server-2008-R2-V01, 2014.