GRADO EN INGENIERÍA INFORMÁTICA

Mención Tecnologías de Información

ADMINISTRACIÓN DE SISTEMAS

Wilfredo J. Torres Moya

viu Universidad Internacional de Valencia

Edita
Universidad Internacional de Valencia

Grado en

Ingeniería Informática

Admnistración de Sistemas

Mención Tecnologías de Información

6 ECTS

Wilfredo J. Torres Moya

Índice

Leyenda

Glosario
Términos cuya definición correspondiente está en el apartado "Glosario".

Enlace de interés
Dirección de página web.

Tema1. Introducción a la administración de sistemas informáticos

La administración de sistemas se refiere la gestión de todos los procesos y actividades que apoyen el correcto despliegue, operación y mantenimiento de los sistemas informáticos de una organización. Independientemente del tamaño de la organización, por lo general una buena práctica de la administración de sus sistemas incluirá procesos para el **diseño** e **instalación** (incluyendo *hardware* y *software*); **configuración** para el correcto acceso y utilización por parte de los usuarios; y la **operación**, **monitorización** y **mantenimiento**, pensando en la continuidad y buen desempeño del sistema durante su vida útil.

Muchos de los conocimientos y procedimientos a aplicar se centran en el manejo de los **sistemas operativos** a ser instalados en los diferentes elementos de la infraestructura tecnológica, desde los terminales de los usuarios, hasta los servidores, pasando por dispositivos de apoyo en las operaciones como dispositivos de redes y comunicaciones, unidades de almacenamiento, seguridad, impresoras, escáneres, entre otros. En este sentido, muchos de los temas a tratar en este material se enfocarán en funcionalidades soportadas por los sistemas operativos, como son los sistemas de archivos, gestión de paquetes de *software*, configuración, monitorización, automatización de tareas, redes, servicios y seguridad.

Así pues, la experiencia deseada para este curso se basa principalmente en el conocimiento del funcionamiento de sistemas operativos en general, y específicamente en ambientes como Unix/Linux

y Windows; además de conocimientos sólidos de redes TCP/IP. De esta manera se espera que este material represente un paquete de información coherente para desarrollar las competencias de un profesional de administración de sistemas, rama muy importante de la Ingeniería en Informática.

1.1. ¿Qué es la administración de sistemas?

La administración de sistemas se refiere a todas las actividades necesarias para **diseñar**, **instalar**, **configurar**, **operar**, **mantener** y **monitorizar** el correcto funcionamiento de los sistemas informáticos en una organización. Para ello se deben conocer los procedimientos de instalación y configuración de los sistemas específicos, de manera que estos puedan ser accedidos y operados adecuadamente por los usuarios asignados. Una vez estos sistemas son puestos en operación, se deben establecer mecanismos para la correcta monitorización y mantenimiento de sus recursos de *hardware* y *software*, de manera que se aseguren las necesidades de los usuarios con las buenas condiciones y continuidad de operación.

Tomando en cuenta esta definición, el administrador de sistemas es **un profesional que debe tener un conocimiento integral sobre muchas de las áreas de la informática y la tecnología**. En las organizaciones que despliegan varios sistemas informáticos, la administración puede llegar a ser compleja. De esta forma, las funciones del administrador de sistemas se dividen en roles, cada uno con sus tareas específicas:

- **Administrador de servidores**
- **Administrador de bases de datos**
- **Administrador de redes**
- **Administrador de correo electrónico**
- **Administrador de servidores web (*web master*)**
- **Administrador de seguridad**
- **Administrador de almacenamiento y respaldo (*storage*)**

1.2. El diseño de sistemas

Más que sólo mantener computadoras y componentes de infraestructura, los administradores de sistemas controlan sistemas completos. Con más experiencia y antigüedad, en última instancia estarán a cargo de construir estos sistemas o de diseñar la infraestructura y sus componentes para que su rol se transforme en el de un **Arquitecto de Sistemas**. En este sentido se podría comparar la situación a la que tienen los ingenieros y arquitectos en diferentes áreas. Los paralelos entre el diseño y la construcción de una casa, una computadora o una pieza compleja de *software* son abundantes.

La función del administrador de sistemas también es frecuentemente comparable a la del ingeniero estructural en el mundo de la mecánica, pero el administrador de sistemas avanzado (que finalmente entra en el papel de planificación, diseño y supervisión de la construcción de sistemas complejos)

podría ser mejor conocido como "Arquitecto de sistemas". Luego se necesitan ingenieros de sistemas estructurales para ayudar en el diseño final y la implementación del sistema en cuestión.

Los administradores de sistemas excepcionales son **generalistas**, debido a que han adquirido experiencia en una amplia gama de temas, o han profundizado su comprensión en varios de estos, y tal vez se han convertido en expertos en algunos. Finalmente vuelven a un rol que les permite aplicar su capacidad para "conectar los puntos", para ver el panorama general, las conexiones, y el impacto de múltiples sistemas entre sí. Tienen la experiencia para modelar nuevos componentes de una infraestructura o rediseñar un sistema desde cero para cumplir con nuevos requisitos. Para que un sistema sea adecuado tanto para los requerimientos inmediatos como para los futuros, cuando las necesidades actuales hayan cambiado, debe incorporar dos principios básicos: **escalabilidad** y **seguridad**. **Ninguno de estos principios puede agregarse a un sistema después de haber sido construido.** Por ejemplo, intentar aplicar "seguridad" después de que las interfaces del sistema han sido definidas puede producir limitaciones en el desempeño; tratar de hacer que un sistema con limitaciones inherentes funcione en situaciones para las que no fue diseñado, puede generar debilidades en el mismo. El resultado final con frecuencia será un sistema muchas deficiencias en cuanto a su fiabilidad (Schaumann, 2019).

Un tercer principio a tomar en cuenta por los expertos diseñadores de sistemas se refiere a la **simplicidad**. En un sistema bien diseñado, se podría decir que la simplicidad está implícita en la escalabilidad y la seguridad, puesto que la reducción de la complejidad implica interfaces mejor definidas, minimizando la ambigüedad en las comunicaciones o el procesamiento de datos, y aumentando la flexibilidad. Al igual que los principios de escalabilidad y seguridad, la simplicidad no puede ser añadida con posterioridad, ésta debe estar incluida durante el diseño o la arquitectura del sistema.

En los siguientes apartados se comentan estos tres principios del diseño de los sistemas.

1.2.1. Escalabilidad

En los últimos años, la palabra "**escalabilidad**" se ha convertido en uno de los requisitos de definición para prácticamente todas las soluciones técnicas. Los proveedores de servicios y productos de infraestructura lo lanzan en un intento de impresionar a sus clientes por la cantidad de carga que pueden manejar sus sistemas. Por lo general, parece casi sinónimo de "alto rendimiento", la capacidad de manejar grandes cantidades de datos, tráfico, conexiones y similares. Esto puede ser engañoso, ya que las personas pueden sentirse inclinadas a creer que tales capacidades no tienen un costo adicional (incremental o proporcional). **De hecho, la escalabilidad no se relaciona con los costos del sistema en ejecución, sino con su arquitectura.** Un sistema escalable puede requerir recursos adicionales y, en muchos casos, asignar dinero a un problema es una solución perfectamente viable. Sin embargo, una **arquitectura escalable** es aquella que no requiere una refactorización de todo el sistema, una reestructuración de cómo se manejan los datos en función de las circunstancias cambiantes. En cambio, **extrae datos del flujo del proceso y simplemente se adapta**.

1.2.2. Seguridad

Con demasiada frecuencia, la industria del *software* o la tecnología de la información trata la seguridad del sistema como una ocurrencia tardía, como algo que se puede agregar al producto

final una vez que cumple con todos los demás requisitos funcionales, después de que se haya determinado la interfaz de usuario y después de que todo el código haya sido escrito. Entonces no es sorprendente que el viejo adagio de que la **seguridad** y la **usabilidad** estén inversamente relacionadas parezca ser cierto. Sin embargo, se podría decir que la misma declaración del problema "cuanto más seguro se hace algo, menos utilizable se vuelve" refleja un malentendido fundamental, ya que implica que la usabilidad está presente primero y la seguridad "se agrega" después.

Los componentes – y sus restricciones- bien definidos, aún pueden proporcionar la flexibilidad para construir sistemas escalables, pero **cada componente debe diseñarse desde el principio teniendo en cuenta la seguridad**. Así como una cadena es tan fuerte como su eslabón más débil, una infraestructura es tan segura como su componente más abierto o expuesto. Desde este punto de vista, la seguridad se reduce a tomar en cuenta factores en el diseño que **reduzcan los riesgos** que puedan aparecer en el funcionamiento del sistema. En general, al hablar de seguridad en sistemas, se deben tomar en cuenta, al menos los siguientes factores:

- **Criptografía:** aplicada específicamente a proveer los siguientes servicios:
  - **Confidencialidad:** se trata de proteger, mediante técnicas de cifrado, la información sensible, o que no pueda ser expuesta ante agentes externos.
  - **Integridad:** que la información que se almacene o se transmita en el sistema no pueda ser modificada sin autorización, y se mantenga fiable.
  - **Autenticidad:** que se pueda verificar la identidad de los usuarios o componentes que procesan información en el sistema.
- **Autenticación o acceso:** control del registro de usuarios y componentes de los sistemas, para luego verificar sus identidades con cada intento de acceso o acción sobre el sistema.
- **Seguridad física:** protección de los accesos y condiciones a las instalaciones físicas de la infraestructura.
- **Disponibilidad:** mecanismos para mantener disponibles todos los servicios ofrecidos por el sistema, incluidas situaciones de recuperación antes fallos y desastres.
- **Ingeniería Social:** mecanismos para mitigar los riesgos asociados a la intervención humana sobre los sistemas.

1.2.3. Simplicidad

"**La complejidad es el enemigo**". Esta cita se ha repetido infinitamente en tantos contextos que casi se puede suponer un conocimiento común entre ingenieros de *software*, criptógrafos y otros expertos en seguridad. Pero, aunque muchas personas en la industria de la tecnología de la información pueden estar de acuerdo en principio, se ha descubierto que, como tantos aforismos, la traducción al mundo real (la aplicación real de sus consecuencias) va muy por detrás de las buenas intenciones.

En el mundo de la administración de sistemas, reducir la complejidad es particularmente difícil. Administrar grandes redes de computadoras tiene una complejidad inherente: los múltiples componentes deben interactuar entre sí de muchas maneras para ayudar a las personas (simultáneamente el origen de la entropía verdadera en lo que respecta a las computadoras y peligrosamente predecibles) para lograr sus objetivos individuales y colectivos. Pero es importante diferenciar entre la **complejidad requerida** y la **complejidad accidental**. Se trata de reducir la complejidad general mediante la construcción de sistemas complejos a partir de componentes más pequeños y simples.

La capacidad de resistir la tentación de "mejorar" la arquitectura o herramienta de *software* para implementar funciones adicionales que, de ser necesarias, podrían ser proporcionadas por un componente separado, generalmente solo viene con una fuerte fuerza de voluntad y años de experiencia. Al mismo tiempo, se encuentra que la simplicidad subyace en todas las tecnologías ejemplares. A veces se necesita una segunda mirada para reconocer y apreciar el genio de la simplicidad.

1.3. Generalidades sobre sistemas operativos

Las funciones de un administrador del sistema varían ampliamente de una organización a otra. Una tarea imprescindible se refiere a la instalación, el soporte y el mantenimiento de servidores u otros sistemas o dispositivos informáticos, lo cual podría traducirse a la tarea de gestión de los **sistemas operativos** sobre dichos elementos.

Un sistema operativo es el *software* más importante que se ejecuta en una computadora. Gestiona la memoria y los procesos de la computadora, así como todo su *software* y *hardware*. Permite la comunicación y control de la computadora a través de una interfaz usuario-máquina que puede ser gráfica o basada en comandos de texto.

El sistema operativo administra los recursos de *hardware* y de *software* de una computadora, que incluyen, entre otros:

- Dispositivos de entrada como un teclado y ratón.
- Dispositivos de salida como pantalla, impresoras y escáneres.
- Dispositivos de red como módems, enrutadores y conexiones de red.
- Dispositivos de almacenamiento como unidades de disco internas y externas.

El sistema operativo también proporciona servicios para facilitar la ejecución, administración eficiente y asignación de memoria a los programas de aplicación de *software* instalados (a parte del propio sistema operativo). Si se ejecutan varios programas al mismo tiempo (como un navegador de Internet, antivirus, editor de documentos, etc.), el sistema operativo asignará los recursos de la computadora (**memoria RAM**, **CPU** y almacenamiento o espacio en disco) para asegurarse de que cada uno reciba lo necesario para su función.

Los conceptos generales de administración de sistemas se aplican en todos los sistemas operativos e incluso en dominios administrativos. Existen sistemas operativos diseñados para ser instalados desde elementos tan sencillo como los computadores de escritorio y dispositivos personales, y hasta en implementaciones de servidores en centros de datos. Por consecuencia, y debido a las filosofías significativamente diferentes y los medios prácticos por los cuales se mantienen estos sistemas, existen en la industria diferentes tipos de sistema operativos, de los cuales se pueden mencionar los siguientes:

- Unix, y varios sistemas basados en él como Solaris o BSD.
- **GNU/Linux.**
- Microsoft Windows.
- MacOS X.

Por otro lado, con la popularización de los dispositivos móviles como los teléfonos inteligentes, tabletas y otros, se tienen sistemas operativos especiales para este tipo de dispositivos:

- Android
- iOS (iPhone de Apple)
- Windows Phone

Figura 1. Logos de los sistemas operativos más populares hoy día en la industria. Elaboración propia.

Los procedimientos específicos de administración de sistemas operativos varían mucho dependiendo de la filosofía de su arquitectura, aunque en general se siguen los mismos principios. Se podría decir que existen **dos filosofías de arquitecturas dominantes**:

- **Sistemas basados en Unix:** Unix AT&T, GNU/Linux, MacOS, BSD, FreeBSD, NetBSD, OpenBSD, y Solaris, por mencionar los más populares.
- **Sistemas basados en Microsoft Windows:** toda la gama de versiones del sistema operativo Microsoft Windows desarrollada para computadores de escritorio, servidores, y dispositivos móviles.

Cada arquitectura tiene lo que se podría llamar el árbol genealógico de su desarrollo, en el que se han liberado a la industria diferentes familias de sistemas operativos. La Figura 2 y la Figura 3 muestran los árboles genealógicos de los sistemas basados en UNIX y en Microsoft Windows, respectivamente.

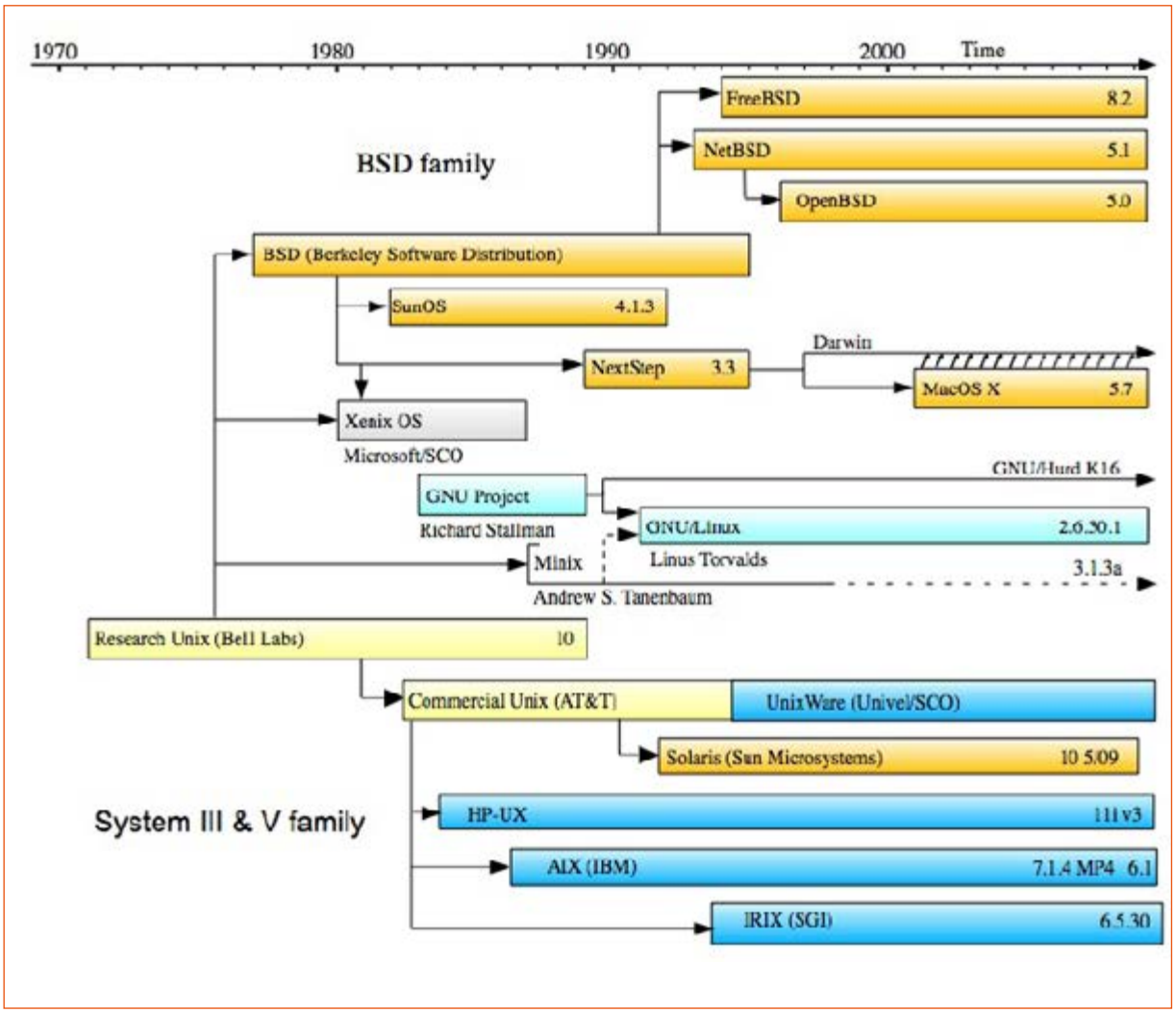

Figura 2. Árbol genealógico de los sistemas operativos basados en UNIX. Adaptado de Schaumann (2019).

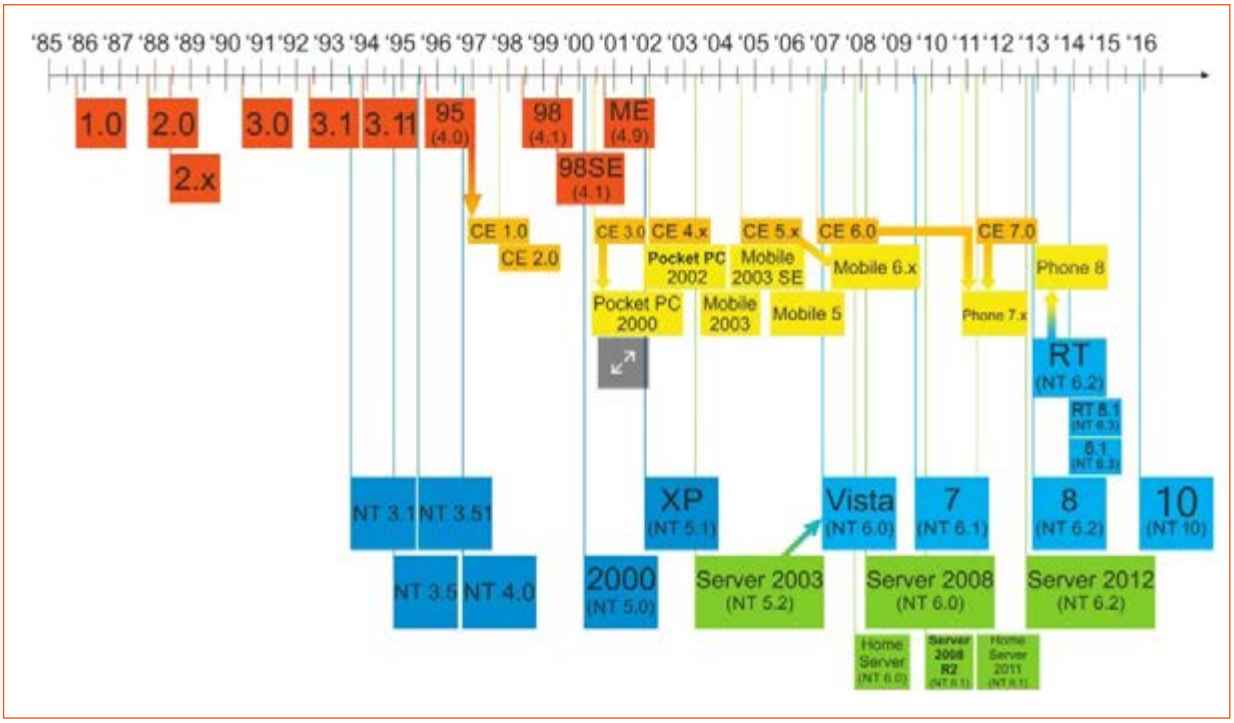

Figura 3. Árbol genealógico de los sistemas operativos basados en Microsoft Windows. Recuperado de http://www.wikiwand.com/

En la Tabla 1 se resumen las diferencias de algunas características importantes entre los sistemas operativos basados en Unix/Linux versus los sistemas basados en Microsoft Windows.

Tabla 1
Comparación de características entre los sistemas operativos basados en UNIX/Linux y Microsoft Windows

Característica	Unix/linux	Microsoft windows
sistema de archivos	Se usan diferentes sistemas, entre los más comunes se encuentran los extX, (ext2, ext3, ext4) y XFS (inspirado en el Unix File System, UFS). Dos tipos de particiones: "swap" y "datos". Gracias a las particiones *swap*, se toma parte de la memoria permanente de almacenamiento para completar memoria RAM que no esté disponible de forma temporal, cubriendo de esta manera la posible insuficiencia de memoria para procesos al alcanzar el máximo de la memoria RAM física. **Si no es un archivo es un proceso**, incluyendo programas, dispositivos de almacenamiento, puertos seriales, dispositivos de entrada/salida, y otros dispositivos. Los ***sockets*** son archivos especiales que proveen información entre procesos, bien sea de forma local (*pipes*) o a través de la red entre máquinas remotas. **Nombres de archivos y directorios:** *case-sensitive*.	Principalmente se usan FAT (*File Allocation Table*) y NTFS (*New Technology File System*). FAT32 es la última version de este esquema con particiones máximas de 32GB. NTFS soporta mayores tamaños de particiones, y características avanzadas como indexado, cuotas de usuarios, cifrado, compresión y puntos de restauración/reparación. No existe el concepto de partición swap. La cantidad de memoria a destinar a los procesos está limitada a la capacidad de memoria RAM física. **Nombres de archivos y directorios:** *non-case-sensitive*.

Característica	Unix/linux	Microsoft windows
Estructura de directorios	Estructura de árbol. El directorio raíz es "/". No se usan letras para representar las unidades de disco/almacenamiento. Los volúmenes de almacenamiento se controlan en el directorio "/dev", y el sistema de archivos con los datos se monta en el directorio "/mnt". Lo directorios son tratados como archivos, en este caso como archivos que contienen nombres y archivos anidados en su interior. Las rutas se separan igualmente con "/"; por ejemplo, el directorio de los usuarios del sistema está por defecto en la ruta **/home/[nombre usuario]**.	Estructura de árbol. Se usan letras para representar las unidades de disco/almacenamiento. La raíz de directorios por defecto es la unidad de disco duro principal (o volumen principal), normalmente "C", y las rutas de directorios y archivos se separan con "\". Por ejemplo, el directorio de los usuarios del sistema está por defecto en la ruta C:\Users\ [nombre usuario].
Manejo de usuarios	Tres tipos de usuarios: **Regular:** cuentas en /home. No tiene permiso a los directorios de otros usuarios, y no puede ejecutar comandos de administración del sistema, a menos que se hagan excepciones puntuales modificando los niveles de ejecución de dichos comandos y asociándolos a cuentas de usuarios regulares específicas. **Superusuario (*root*):** cuenta en /root. Tiene acceso a toda la estructura de ditrectorios y puede ejecutar todos los comandos del sistema. **Servicios:** los servicios del sistema tienen asociados usuarios especiales, pudiéndose permitir o denegar el acceso a varios recursos dependiendo del servicio. Esto mejora los niveles de seguridad del sistema.	Cuatro tipos de usuarios: **Administrador:** es el superusuario del sistema, teniendo acceso a todos los directorios y procesos de administración como la instalación de programas, gestión de usuarios y cambios de configuraciones. **Estándar:** usuarios regulares con permisos sólo a su rama en el árbol de directorios, y restricciones en funciones administrativas. **Child:** cuentas especiales para niños y jóvenes, con restricciones de acceso y posibilidades de monitoreo desde cuentas de adultos. **Invitado:** usuarios estándares con validez temporal para acceso al sistema.

Nota. Elaboración propia

Si bien este material ofrece información de fundamentos generales de administración de los sistemas operativos, los ejemplos concretos configuraciones y comandos se desarrollan principalmente para sistemas basados en Unix/Linux, debido a su popularidad en muchos ambientes de administración de sistemas (especialmente a nivel de servidores), y a la facilidad del montaje de laboratorios y prácticas con distribuciones libres y de código abierto, como es el caso de las distribuciones de GNU/Linux.

1.3.1. Componentes principales de un sistema operativo

Son varios los componentes necesarios para el funcionamiento del sistema operativo en un sistema de computación, que bien podría ser un *hardware* o una máquina virtual. A continuación, se describen los más importantes, de cara a la instalación y gestión del sistema operativo.

El núcleo o *kernel*

Proporciona **control de bajo nivel sobre todos los dispositivos de *hardware* de la computadora, y procesos críticos del sistema** como el manejo de archivos, leer y escribir datos en la memoria RAM, procesar órdenes de ejecución contra el CPU, determinar cómo reciben y envían los datos los dispositivos de entrada/salida como el monitor, el teclado y el ratón; y determinar cómo interpretar los datos recibidos de las redes.

Los núcleos monolíticos tienen un diseño más simple y consisten en un único código que se comunica con todo el *hardware* y *software*.

Los ***microkernels*** implementan servicios de usuario y núcleos en diferentes espacios de direcciones, reduciendo su tamaño, pero forzando el uso de mensajes que pasan para ejecutar servicios.

La interfaz de usuario (*User Interface*, UI)

Este componente **permite la interacción con el usuario**, lo que puede ocurrir a través de iconos gráficos y un escritorio o mediante una línea de comandos de texto.

La interfaz de usuario se divide además en la interfaz de línea de comandos (CLI), que consiste en una interfaz basada en texto donde los usuarios avanzados pueden solicitar comandos específicos al escribirlos, y una interfaz gráfica de usuario (**GUI**). Esta última es una interfaz visual que permite al usuario emitir comandos al interactuar con símbolos, iconos y menús utilizando un dispositivo de entrada como un mouse o panel táctil.

La interfaz de programación para aplicaciones (*Application Programming Interface*, API)

Este componente **permite a los desarrolladores de aplicaciones escribir código modular que funcione sobre la computadora (física o virtual) controlada por el sistema operativo**. Una **API** define cómo otros sistemas o componentes pueden usar una determinada aplicación.

Gracias a la de la API, las aplicaciones se pueden desarrollar para acceder a los recursos del sistema de forma personalizada, sin necesidad de escribir programas de que interactúen con el *hardware* o los elementos del sistema a bajo nivel: puertos físicos o lógicos, dispositivos de entrada/salida, dispositivos de red, espacios de memoria, etc. Las aplicaciones a desarrollarse alcanzan la API a través de la interfaz del sistema operativo (comandos de operación crítica), o directamente, para que la API ejecute rutinas que alcanzan los recursos a través del *kernel* de sistema operativo. En este sentido, la API representa una interfaz entre las aplicaciones y el *kernel*.

En la Figura 4 se muestran los componentes mencionados del sistema operativo, y las interfaces que tienen con las aplicaciones de los usuarios, y el *hardware* y dispositivos que se controlan a través del núcleo. Se observa la estructura jerárquica que predomina en el funcionamiento del sistema, donde los niveles superiores de funcionamiento (aplicaciones de usuario) logran acceder a los recursos a través de las diferentes capas de abstracción que se implementan en el sistema operativo.

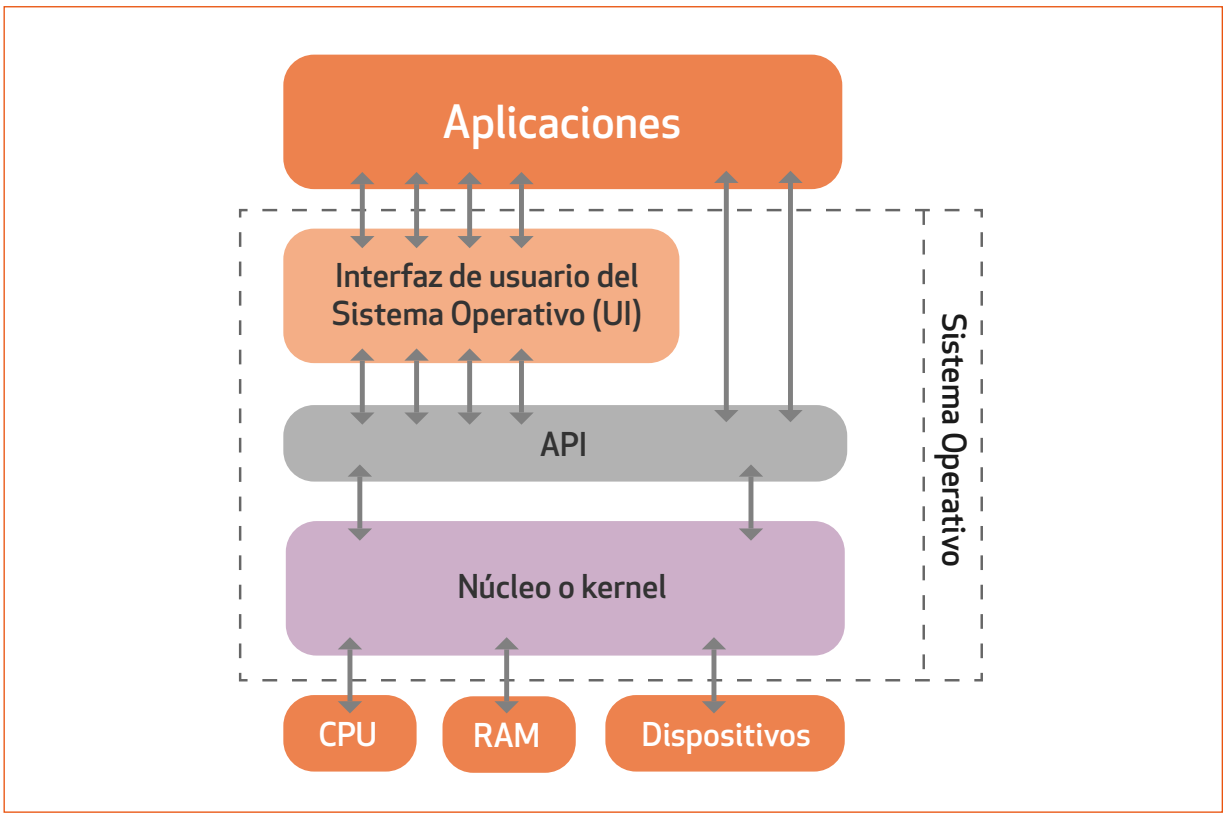

Figura 4. Componentes del sistema operativo. Elaboración propia.

1.3.2. Instalación y mantenimiento de los sistemas operativos

El sistema operativo representa "el alma" del equipo informático sobre el cual se instala, sea éste una PC, laptop, teléfono móvil, servidor, dispositivo de red, u otro que en general provea algún servicio informático. Como se mencionó anteriormente, existen muchos tipos de sistemas operativos, cada uno con sus propios procesos para distribución de imágenes u procesos de instalación. En el próximo capítulo se ofrece el detalle de este proceso para los sistemas basados en Unix/Linux, aunque este proceso será similar para el resto de los sistemas operativos, ya que la **arquitectura de los computadores o equipos** sobre los que se instalan los sistemas son los mismos:

- Memoria principal (disco duro local)
- Memoria RAM
- Dispositivos de entrada/salida
- CPU

Los administradores de sistemas deben conocer este proceso de instalación, tomando en cuenta no solo los aspectos de *software* del sistema operativo como tal, sino también el entorno sobre el cual dicho sistema se instalará, en su conjunto. Ambos aspectos serán claves luego para el mantenimiento. En general se deberán tomar en cuenta los siguientes aspectos para el mantenimiento, los cuales se desarrollan con mayor detalle en los próximos capítulos:

- Seguridad y estabilidad del ambiente físico de instalación y operación, incluyendo seguridad en acceso, energía, y condiciones ambientales (humedad y temperatura).
- Monitorización del *hardware* y *software* del sistema.
- Actualizaciones periódicas del *software* y parches de seguridad.
- Actualizaciones periódicas del *hardware* para mantener la capacidad del sistema.
- Atención de incidentes y soporte técnico.
- Respaldo y recuperaciones.
- Desincorporación del sistema al final de su vida útil.

Tema 2. Instalación, despliegue y mantenimiento de sistemas operativos

En este capítulo se describirán procesos relacionados con la instalación, el despliegue y el mantenimiento de sistemas operativos, como una actividad fundamental en la administración de sistemas. Todo comienza con la **instalación del *hardware* o la plataforma** sobre la cual se instalará el *software* con interfaz de usuario del sistema operativo. Dicha plataforma puede ser **física** (máquinas físicas con todos sus componentes), o **virtualizada**, en la cual el *hardware* es emulado en una capa de abstracción sobre la cual pueda funcionar el sistema operativo y todo el ***firmware*** que controlará el *hardware*. Aún en las plataformas virtualizadas, debe existir una plataforma física que soporte dicha virtualización.

Acto seguido se instala el sistema operativo con todos sus componentes (ver Figura 4), teniendo que hacer en muchos casos algunos ajustes en la instalación para, por ejemplo, actualizar el *firmware* provisto por los fabricantes para el correcto funcionamiento de los dispositivos, como los de entrada/salida, tarjetas de video, red, entre otros. Una vez se tenga correctamente configurada la plataforma y el sistema operativo, esta se prepara para el acceso de los usuarios, configurando correctamente los parámetros de red y seguridad (grupos, usuarios, niveles de acceso, etc.). El detalle de estos tópicos de administración de sistema operativos se desarrolla en los siguientes apartados.

2.1. Instalación del *hardware*

Para ser administrador del sistema, es importante tener una apreciación básica de las debilidades y procedimientos que rodean al *hardware*. A parte del sentido común y la adecuada lectura de las instrucciones y consideraciones de seguridad, es importante tomar en cuenta las buenas prácticas en el manejo de sistemas electrónicos, frágiles y vulnerables ante aspectos como las descargas electromagnéticas y la estática.

Luego, será importante el entendimiento de la estructura básica de un sistema de computación. Todas las computadoras contemporáneas de uso común se basan en la arquitectura Eckert – Mauchly – von Neumann, esbozada en la Figura 5. Cada computadora tiene un reloj que impulsa una unidad de procesador central (CPU), una memoria de acceso aleatorio (RAM) y una variedad de otros dispositivos, como unidades de disco y almacenamiento. Para que estas partes funcionen juntas, la CPU está diseñada para ejecutar programas que pueden leer y escribir en dispositivos de *hardware*. El programa más importante es el **núcleo del sistema operativo**. Además de esto, hay capas de *software* que proporcionan abstracciones de trabajo para programadores y usuarios, que son las mencionadas APIs del apartado anterior. Estos consisten en archivos, procesos y servicios. Parte del "sistema" se refiere a los dispositivos de red que transportan mensajes de una computadora a otra, incluidos los propios cables. Finalmente, el sistema se refiere a todas estas partes y niveles que trabajan juntos (Burgess, 2004).

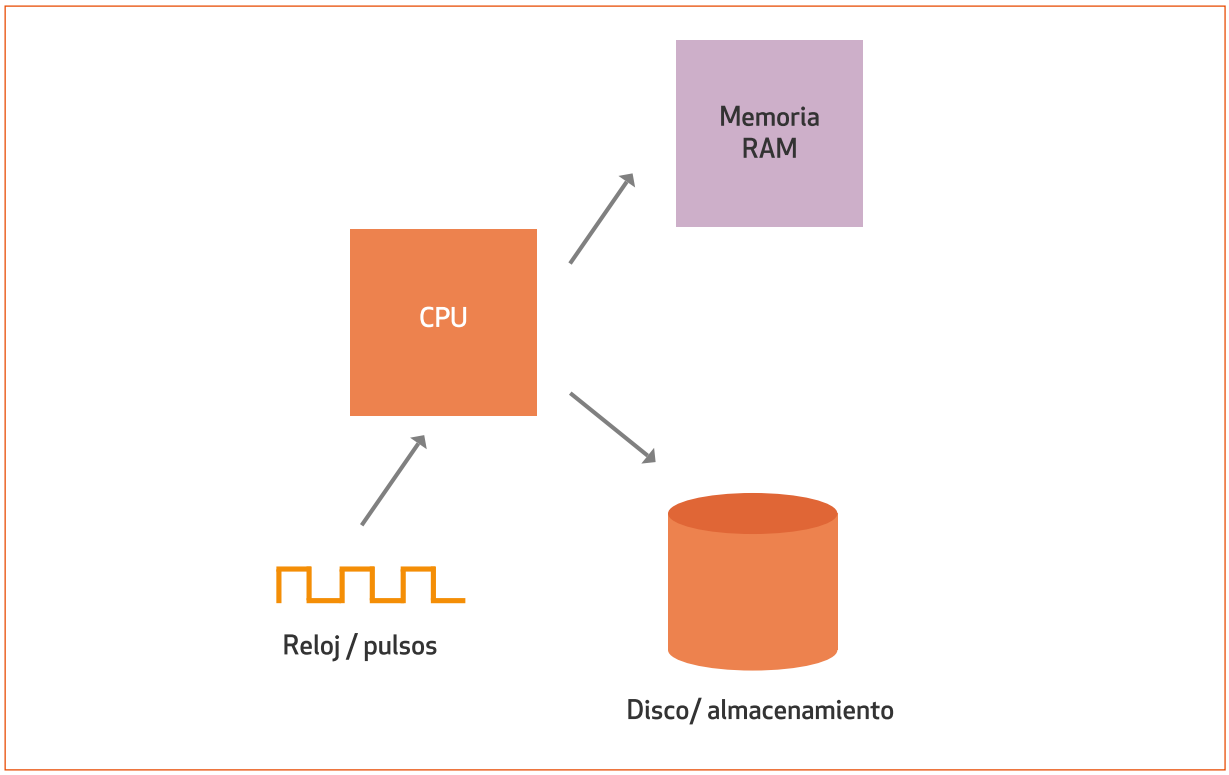

Figura 5. Arquitectura Eckert – Mauchly – von Neumann de una computadora. Adaptado de Burgess (2004).

Para que la computadora (o dispositivo de computación en términos generales) trabaje, el primer paso se refiere a la instalación del *hardware*. Son muchos los aspectos a tomar en cuenta para este proceso, desarrollados los más importantes en los siguientes apartados.

2.1.1. El ambiente de la instalación

Las condiciones del ambiente de la instalación son muy importantes para la operación del *hardware*. A continuación, se comentan varios aspectos a tomar en cuenta.

- **Puesta a tierra:** los picos de voltaje debidos a descargas eléctricas inesperadas como lo rayos o la misma estática generada en el ambiente de instalación pueden destruir equipos frágiles. Ningún fusible protegerá el *hardware* de un rayo, por ejemplo. Los transistores y los chips **CMOS** se queman mucho más rápido que cualquier fusible. **Los protectores de picos electrónicos junto a un adecuado sistema de puesta a tierra pueden ayudar en la protección del sistema en este sentido.**

- **Alimentación: el fallo en los sistemas de alimentación puede causar daños en las unidades de disco y pérdida de datos.** Se debe instalar un subsistema **UPS** (*Uninterruptible Power Supply*) adecuadamente dimensionado para el consumo de los equipos, como una fuente de energía de respaldo para contrarrestar las fallas en los sistemas de alimentación.

- **Calor:** el calor abrasador del verano o un calentador mal colocado pueden causar que los sistemas se sobrecalienten y se apaguen repentinamente. No se debe permitir que la temperatura ambiente cerca de una computadora se eleve mucho más de 25 grados centígrados. Claramente, algunos equipos pueden tolerar el calor mejor que otros equipos. Se debe tener en cuenta que los metales se expanden significativamente, por lo que las partes móviles como los discos serán las más afectadas por el calor. El aumento de la temperatura también aumenta los niveles de ruido que pueden reducir la capacidad de la red en una fracción de un porcentaje. Si bien esto puede no parecer mucho, una fracción de un porcentaje de un cable Gigabit Ethernet de cobre, por ejemplo, involucra la afectación de mucha capacidad. El calor puede hacer que la RAM funcione de manera impredecible y que los procesos de lectura/escritura de los discos fallen. **Una buena ventilación es esencial para las computadoras y pantallas para evitar fallos eléctricos.**

- **Frío:** los cambios repentinos de calor a frío son igual de perjudiciales. Pueden causar cambios impredecibles en las propiedades eléctricas de los chips y hacer que los sistemas se bloqueen. A largo plazo, estos cambios podrían provocar **grietas en las placas de circuitos y daños irreparables en los chips**.

- **Humedad:** en épocas de clima muy frío y calor muy seco, la humedad cae a niveles muy bajos. En estos momentos, la cantidad de electricidad estática se acumula a niveles bastante altos sin disiparse. Esto puede ser un riesgo para los circuitos electrónicos. Los humanos recogen la carga simplemente caminando, lo que puede destruir los circuitos frágiles. El papel se adhiere entre sí y causa fallas en las impresoras láser. **Demasiada humedad puede provocar condensación y cortocircuitos.** Por ende, será importante la instalación de equipos que regulen los niveles de humedad en el sitio de instalación.

Lo anterior son solo aspectos básicos a tomar en cuenta en cualquier sitio de instalación de un sistema de computación. Esto puede ser más complejo, si se toma en cuenta sitios como centros de datos,

que deben cumplir estándares de instalación rigurosos, que toman en cuenta aspectos de obra civil y seguridad física de acceso, entre otros.

2.1.2. Los componentes y su ensamblado

Una vez se tenga el sitio de instalación, con las condiciones ambientales adecuadas, normalmente el siguiente paso es la instalación de los componentes del equipo de computación. A continuación, algunos aspectos a tomar en cuenta.

- **Interfaces y conectores:** el *hardware* a menudo se conecta a una interfaz mediante un cable o conector. **Obtener el cable correcto es de vital importancia**. Muchos fabricantes usan cables que se parecen, superficialmente, pero que en realidad son diferentes. Un cable incorrecto puede dañar la interfaz. Los cables de módem en particular pueden dañar una computadora o un módem si están cableados incorrectamente, ya que algunas computadoras suministran energía a través de estos cables, lo que puede dañar el equipo que no espera encontrar niveles de voltaje en el cable.

- **Manipulación de componentes:** los chips CMOS modernos funcionan a bajos voltajes (generalmente 5 voltios o menos). De pie en el suelo con zapatos aislantes, se pueden recoger cargas eléctricas estáticas de varios miles de voltios. Tal carga puede destruir instantáneamente los chips de la computadora. **Antes de tocar cualquier componente de la computadora, se debe poner a tierra tocando el chasis de la computadora, o usando pulseras o guantes anti-estáticos**. Especialmente si se está instalando el componente o dispositivo dentro de una computadora. Evitar usar sandalias de goma o zapatos que aíslen de la tierra cuando se trabaje con equipos de caja abierta, ya que esto hace que el cuerpo acumule carga que puede descargar a través de ese equipo. Por otro lado, es una buena idea usar suelas de goma cuando trabaje cerca de fuentes de alto voltaje o corriente.

2.1.3. Los discos de almacenamiento

La tecnología de discos ha mejorado constantemente durante las últimas décadas. Los tipos de disco más comunes en la industria se dividen en dos familias:

- **Discos ATA** (*Advanced Technology Attachment*, antiguos IDE)

- **Discos SCSI** (*Small Computer System Interface*)

Los IDE (*Integrated Drive Electronics*) y SCSI, originalmente tenían propiedades que desde entonces han evolucionado más rápido que los prejuicios sobre ellos. Los discos ATA ahora son generalmente más baratos que los discos SCSI (debido a las ventas por volumen) y se destacan en el acceso secuencial, pero los discos SCSI han sido tradicionalmente más eficientes en el manejo de accesos múltiples debido a un diseño de bus multitarea y, por lo tanto, son mejores en sistemas multitarea, donde son aleatorios El acceso es importante. Sin embargo, el diseño del sistema de archivos también juega un papel importante en la determinación del rendimiento percibido de cada uno; es decir, cómo los sistemas operativos utilizan los buses durante las actualizaciones es al menos tan importante como

el rendimiento del bus en sí. Las comparaciones interesantes muestran que la tecnología IDE ha alcanzado la ventaja que los discos SCSI alguna vez tuvieron para muchos propósitos, pero no todos.

SCSI viene en varias opciones: SCSI 1, SCSI 2, SCSI *wide*, *fast-wide*, etc. La diferencia tiene que ver con el ancho del bus de datos y la cantidad de discos que se pueden conectar a cada controlador. Actualmente hay tres estándares SCSI: SCSI-1, SCSI-2 y SCSI-3. El estándar SCSI-2 define también SCSI *wide*, *fast* y *fast-wide*. Cada disco SCSI tiene su propia dirección (o número) que debe establecerse cambiando una configuración en el gabinete del disco o cambiando la configuración del *jumper* dentro del gabinete. Los discos más nuevos tienen identidades programables. La cadena de buses del disco debe terminarse con un conector de terminación adecuado. Los discos más nuevos a menudo contienen mecanismos automáticos de terminación integrados en el *hardware*. **Los dispositivos en el bus SCSI se comunican con la computadora a través de un controlador**. En las PC modernas, el controlador SCSI generalmente está conectado al bus PCI, ya sea como una solución integrada en las placas base o como una tarjeta separada en una ranura PCI. Otros buses también se utilizan como transportadores del protocolo SCSI, como **FireWire (IEEE 1394)** y **USB**. El estándar SCSI también es compatible con dispositivos de medios extraíbles (CD-ROM, CD-R, unidades Zip), capturadores de fotogramas de video, escáneres y serpentinas de cinta (DAT, DLT). En la Tabla 2 se resumen algunas diferencias entre las familias de discos SCSI y ATA.

Tabla 2
Comparación entre discos SCSI y ATA (IDE)

Característica	SCSI	ATA (IDE)
Capacidad de almacenamiento	Permite la conexión de 7 a 15 dispositivos (según el ancho del bus). Esto hace posible que todos los dispositivos estén en la misma placa en lugar de comprar una placa diferente para diferentes dispositivos. Eso aumenta los costos y la complejidad dentro de la CPU.	PATA (*Parallel ATA*) permite la conexión de dos dispositivos por canal.
Velocidades (MB/s)	160/320	150/300/600.
Hot-Pluggable	Sí	No
Expandible	Posible, pero muy costoso.	Más fácil de configurar
Relación costo-desempeño	Alto costo / alta velocidad de transferencia	Buena

Nota. Elaboración propia

Si bien los discos SCSI y ATA (IDE) han sido tradicionalmente los más empleados en las instalaciones de computadoras, en los últimos años han aparecido muchos estándares que aumentan el desempeño en velocidades y capacidades de almacenamiento, popularizándose en instalaciones como los servidores y unidades de almacenamiento para centros de datos. Tal es el caso de los discos SATA (*Serial ATA*) y SAS (*Serial Attached SCSI*).

SATA es una evolución de la interfaz de almacenamiento físico *Parallel ATA* (PATA). SATA es una conexión en serie: un solo cable con un mínimo de cuatro hilos crea una conexión punto a punto entre dispositivos. Con SATA, se amplían las capacidades de ATA y ofrece velocidades de transferencia

a partir de 150 MB/s. SATA se lanzó originalmente como una interfaz de 1.5 Gbits/s, y en su versión más moderna puede transferir hasta 6 Gbits/s. Ofrece conectividad serial y ***hot-puggling***. Luego de unos años de desarrollo, se ha convertido en la corriente principal de las interfaces de disco.

Por otro lado, el sucesor de la interfaz SCSI es SAS a velocidades de hasta 3Gb/s. Además, también aborda problemas de interfaz paralela, como la capacidad de direccionamiento de la unidad y las limitaciones en la cantidad de dispositivos por conexión de puerto. Los dispositivos SAS pueden comunicarse con dispositivos SATA y SCSI (los planos posteriores de los dispositivos SAS son idénticos a los dispositivos SATA). Una diferencia clave entre los dispositivos SCSI y SAS es la adición en los dispositivos SAS de dos puertos de datos, cada uno de los cuales reside en un dominio SAS diferente. Esto permite una completa redundancia de conmutación por error. Si una ruta falla, todavía hay comunicación a lo largo de una ruta separada e independiente.

La comparación de las tres interfaces ATA (IDE) lleva a **la conclusión de que la mayoría de los sistemas personales actuales usan SATA**. IDE es costoso y ha sido reemplazado con éxito por SATA. SCSI casi se ha vuelto obsoleto y bien puede pasar a la historia unos años más adelante. SATA es el futuro hasta que aparezca una opción más conveniente y barata. Su bajo precio y atractivo universal solo se suma a su puntaje. Sin embargo, lo que no se puede negar es que SCSI, IDE y SATA han cambiado la forma en que se usan las computadoras y los dispositivos relacionados en la actualidad.

En la Figura 6 se muestra la apariencia de las unidades de disco duro mencionadas, y su popularidad de utilización en PCs o servidores.

Figura 6. Apariencia de los discos IDE (ATA), SCSI, SATA y SAS, y su uso en computadoras o servidores. Recuperado de https://www.pngwave.com/

2.2. Instalación del sistema operativo

Un sistema operativo está compuesto de los siguientes elementos clave:

a. El **núcleo o *kernel* del sistema**, que es responsable de asignar y compartir los recursos del sistema entre varios programas o procesos en ejecución. Es una capa técnica de *software* para

manejar el *hardware* de la computadora, como unidades de disco, el teclado, la pantalla y las interfaces de red. En el núcleo se ejecutan los llamados **controladores** o *drivers* de los dispositivos.

b. Un **sistema de archivos** que proporciona una forma de organizar los archivos de forma lógica.

c. Una **interfaz de usuario** que permite a los usuarios ejecutar sus propios programas y manipular sus archivos de una manera simple.

El *kernel* del sistema se complementa con una serie de servicios de soporte (paginación, **RPC**, **FTP**, **WWW**, etc.) que ayudan al *kernel* o extienden su uso compartido de recursos al dominio de la red. El sistema operativo puede ser responsable de compartir los recursos de una sola computadora, pero cada vez son más comunes los **sistemas operativos distribuidos** en los que la ejecución de programas y el intercambio de recursos se realiza sin tener en cuenta los límites del *hardware*; o **sistemas operativos de red** en los que un servidor central agrega funcionalidad a estaciones de trabajo relativamente "tontas". A veces, los programas que no afectan el trabajo de compartir recursos se denominan programas de usuario.

El sistema operativo ejecuta programas interactivos para usuarios, servicios para usuarios locales y distribuidos y programas de soporte que trabajan juntos para proporcionar la infraestructura que permite que los recursos de la máquina se compartan entre muchos procesos. Algunos sistemas operativos también proporcionan editores de texto, compiladores, depuradores y una variedad de otras herramientas. Dado que el sistema operativo (SO) está a cargo de una computadora, todas las solicitudes para usar sus recursos y dispositivos deben pasar por el núcleo del sistema operativo. Por lo tanto, un sistema operativo proporciona puntos de entrada legales en su código para realizar operaciones básicas como escribir en dispositivos (Burgess, 2004).

Los sistemas operativos pueden clasificarse tanto por la cantidad de tareas que pueden realizar "simultáneamente" como por la cantidad de usuarios que pueden usar el sistema "simultáneamente". Es decir: **monousuario** o **multiusuario**, y **monotarea** o **multitarea**. Un sistema multiusuario debe ser claramente multitarea. La siguiente tabla muestra algunos ejemplos de sistemas operativos, y su clasificación dentro de las categorías mencionadas.

Tabla 3
Ejemplos de sistemas operativos y su clasificación como mono/multi - usuario/tarea

Sistema operativo	Usuarios	Tareas	Procesadores
mS/PC DOS	S	S	1
Windows 3x	S	QM	1
Macintosh System 7	S	QM	1
Windows 95/98/ME	S	M*	1
AmigaDOS	S	M	1
Unix-like	M	M	n
VMS	M	M	n

Sistema operativo	Usuarios	Tareas	Procesadores
Windows NT-like	S/M	M	n
Windows 2000/XP	M	M	n
OS390 (zOS)	M	M	n
Leyenda: **S:** *Single* -> Monousuario (*Single-user*) – Monotarea (*Single-task*) **QM:** *Quasi-multitasking* **M:** Multi -> Multiusuario (*Multi-user*) – Multitarea (*Multi-tasking*) **n**: Más de un procesador.			

Nota. Adaptado de Burgess (2004)

Ha habido una evolución con los años al pasar de sistemas operativos monousuario y monotarea como lo fueron MS/PC DOS, Microsoft Windows 95/98/ME o AmigaDOS. Microsoft Windows 3x y Macintosh System 7 son sistemas considerados monousuario y "cuasi-multitarea" (QM: *Quasi-Multitasking*) debido a que ejecutaban sobre el *kernel* emuladores de sistemas basados en Unix (un sistema multitarea desde sus inicios) para obtener esta característica. Eso significa que es posible ejecutar varias aplicaciones de usuario simultáneamente, pero sólo un usuario a la vez. Antes de Mac OS X, Macintosh no era un verdadero sistema multitarea; si un programa fallaba, todo el sistema colapsaría. Del mismo modo, Windows 9x pretendía ser multitarea preventiva, pero muchos bloqueos de programas también colapsarían todo el sistema.

Los nuevos sistemas operativos de Microsoft (desde Windows NT y 2000 hasta los actuales Microsoft Windows 10 y Server 2016-2019) están basados en la **familia de Microsoft Windows NT**, que son sistemas multitarea y multiprocesadores desde sus inicios. La familia Windows NT se basa, en parte, en el antiguo núcleo VAX / VMS de *Digital Equipment Corporation* (DEC) y la API de Windows 32. Tiene memoria virtual y soporte multihilo para varios procesadores. NT tiene un modelo de objetos incorporado y un marco de seguridad modernizado con respecto a sus predecesores. Como se ha mencionado, las nuevas líneas de sistemas operativos de Microsoft vienen adoptando muchas de las características exitosas de la familia NT, evolucionando en aspectos como la seguridad, rendimiento del *kernel*, y con manejo de servicios avanzados para ambientes corporativos como los servicios de directorio consistentes. Las versiones posteriores de Windows NT y Windows 2000 permiten una verdadera multitarea y múltiples inicios de sesión también a través de un servidor de terminal. Estas nuevas versiones de los sistemas operativos de Microsoft por lo tanto tienen una funcionalidad comparable a Unix a este respecto.

Las mini computadoras IBM S/370, S/390 *mainframe* y AS/400 son ampliamente utilizadas en bancos y organizaciones grandes con requerimientos de procesamiento de alto nivel. Estos son **sistemas totalmente multitarea de alto calibre**, que **admiten arquitecturas de máquinas virtuales**. Estas computadoras *mainframes* ahora se conocen como las computadoras de la **serie z de IBM**, y el sistema operativo es z/OS. z/OS tiene un administrador de alojamiento virtual que puede soportar múltiples sistemas operativos concurrentes. Las computadoras de la serie Z han experimentado un renacimiento con el advenimiento de GNU/Linux. IBM ha reportado que ejecuta miles de núcleos virtuales de Linux concurrentes en sus computadoras *mainframe*.

"Unix" (en la medida en que es correcto llamarlo así ahora) viene en muchas formas, desarrolladas por diferentes fabricantes y entusiastas. Originalmente diseñado en AT&T, Unix se dividió en dos campos desde el principio: BSD (*Berkeley Software Distribution*) y *System V* (o *System 5*), que es una licencia de AT&T. La versión BSD fue desarrollada como un proyecto de investigación en la Universidad de California Berkeley (UCB). Muchas de las funciones de red y características de uso amigable (*user-friendly*) se originan a partir de estas modificaciones. Con el tiempo, estas dos versiones se han fusionado y **la mayoría de los sistemas ahora son una mezcla de ambos mundos**. Históricamente, BSD Unix ha sido más frecuente en las universidades, mientras que *System V* ha sido dominante en los entornos empresariales. En la década de 1990, **Sun Microsystems** y **Hewlett Packard** comenzaron a avanzar hacia el *System V*, manteniendo solo las características más importantes del sistema BSD, pero luego suprimieron los aspectos visibles de *System V* a favor de BSD nuevamente. Hoy, las diferencias son pocas, gracias a una estandarización de facto. Un comité de estandarización para Unix llamado POSIX, formado por los principales proveedores y grupos de usuarios independientes, ha hecho mucho para brindar compatibilidad al mundo de Unix. algunas versiones comunes de sistemas operativos basados en Unix se muestran en la Tabla 4.

Tabla 4
Versiones comunes de Sistemas Operativos basados en Unix

SO basado en UNIX	Fabricante	Tipo
bSD	Univ. California Berkeley	BSD
SunOS (Solaris 1)	Sun Microsystems	BSD/Sys V
Solaris(2)	Sun Microsystems	Sys V/BSD
Tru64	DEC/Compaq/HP	BSD/Sys V
HPUX	Hewlett Packard	Sys V
AIX	IBM	Sys V / BSD
IRIX	Silicon Graphics	Sys V
GNU/Linux	GPL Free Software	Posix (Sys V/BSD)
MacOS X	Apple	BSD/Sys V
Unixware	Novell	Sys V

Nota. Adaptado de Burgess (2004).

Unix es posiblemente el sistema operativo más importante, tanto por su uso generalizado como por su importancia histórica. Unix es uno de los sistemas operativos más portables, de los disponibles en la industria. Funciona en muchas plataformas de *hardware* o virtualizadas, desde computadoras personales hasta supercomputadoras. **Es particularmente bueno para administrar grandes aplicaciones de bases de datos y puede ejecutarse en sistemas con cientos de procesadores.** La mayoría de los sistemas operativos tipo Unix admiten el procesamiento simétrico de subprocesos múltiples y todos admiten inicios de sesión simultáneos de múltiples usuarios. Esta es la razón por la que los procesos específicos de instalación, operación y mantenimiento expuestos en este material se basan en sistemas basados en Unix, especialmente a la versión en código abierto GNU/Linux.

Una vez realizado el proceso de instalación del *hardware* y la preparación del ambiente de instalación, como se ha descrito en el apartado anterior, se procede con la instalación del sistema operativo como

tal, que básicamente se trata de la instalación del *kernel* y la interfaz de usuario del sistema, junto con los drivers que manejen adecuadamente los dispositivos que componen el sistema (*hardware* o plataforma virtualizada).

2.2.1. Instalación del sistema operativo (distribuciones de Linux)

Hay muchas formas de instalar una distribución de Linux. El método más común es arrancar desde un CD-ROM o DVD que contiene el programa de instalación y el *software* instalable. Dicho CD se puede grabar a partir de una imagen ISO descargada, comprado por un módico precio, proporcionado como un disco de portada con una revista, enviado de forma gratuita a pedido, u obtenido como parte de un conjunto de cajas que también puede incluir manuales y *software* comercial adicional. Los nuevos usuarios tienden a comenzar **particionando un disco duro** para mantener su sistema operativo previamente instalado. **La distribución de Linux se puede instalar en su propia partición separada** sin afectar los datos previamente guardados.

Hoy en día, la mayoría de las distribuciones ofrecen conjuntos de CD, DVD o unidades externas de almacenamiento con los paquetes vitales en el primer disco y los paquetes adicionales en los posteriores. Por lo general, también permiten la instalación a través de una red después de iniciar desde un conjunto de discos con solo una pequeña cantidad de datos.

Anaconda (https://fedoraproject.org/wiki/Anaconda), uno de los instaladores más populares, es utilizado por **Red Hat Enterprise**, **Oracle Linux**, **Scientific Linux**, **CentOS**, **Qubes OS**, **Fedora**, **Sabayon Linux** y **BLAG Linux and GNU**, así como otras distribuciones, para simplificar el proceso de instalación. Por su parte, distribuciones como Linux Debian, Ubuntu, y SuSe Linux utilizan sus propias herramientas de instalación y configuración como Synaptic o YaST.

Otro modo de instalación es instalar en una computadora poderosa para usar como servidores y usar máquinas menos potentes (tal vez sin discos duros, con menos memoria y CPU más lentas) **como clientes ligeros en la red**. Los clientes pueden arrancar a través de la red desde el servidor y mostrar resultados y pasar información al servidor donde se ejecutan todas las aplicaciones. Los clientes pueden ser PC normales con la adición de un gestor de arranque de red en una unidad o controlador de interfaz de red; El espacio en el disco duro y la potencia del procesador se pueden descargar en la máquina del cliente si se desea. Los ahorros de costos logrados mediante el uso de clientes ligeros pueden invertirse en una mayor capacidad informática o almacenamiento en el servidor.

En una configuración de Live CD, la computadora arranca todo el sistema operativo desde el CD sin instalarlo primero en el disco duro de la computadora. Algunas distribuciones tienen un instalador de Live CD, donde la computadora inicia el sistema operativo desde el disco, y luego procede a instalarlo en el disco duro de la computadora, proporcionando una transición perfecta desde el sistema operativo que se ejecuta desde el CD al sistema operativo que se ejecuta desde el disco duro.

Independientemente de la distribución del sistema operativo basado en Unix/Linux y del *software* de instalación empleado, el procedimiento general consistirá en los siguientes pasos:

1. **Iniciar la máquina desde el CD/DVD/Unidad de memoria con la imagen (.iso) del sistema operativo.** El primer disco normalmente contendrá el sistema operativo base y los paquetes

de *software* imprescindibles. Se debe emplear una imagen del sistema operativo adecuada a la arquitectura del procesador: x386-32bits, Intet/AMD-64 bits, etc.

2. Durante el proceso de instalación **se preguntará por opciones de configuración** como el idioma, la zona horaria, el conjunto de caracteres a utilizar, y la distribución del teclado.

3. **Crear la cuenta del superusuario (*root*) y la contraseña.**

4. **Crear la cuenta de un usuario estándar del sistema** (sin permisos de administración) **y la contraseña**.

5. **Crear particiones en el disco duro.** Una partición es un espacio en el disco duro que se formatea para la instalación y ejecución del sistema operativo sobre la máquina. Normalmente para Unix/Linux se solicitará la creación de una **partición primaria de datos** y otra **partición *swap***. Opcionalmente se podrán configurar **particiones secundarias**, dependiendo de la forma como se quieran organizar los directorios, cuotas de usuarios, etc. La configuración de este paso dependerá, a parte de las preferencias de configuración de diferentes particiones, de la capacidad que tenga la unidad de almacenamiento principal (normalmente el disco duro principal). En este paso es importante también la selección del **sistema de archivos** a implementar. Normalmente para el caso de Linux se emplea **ext3** o **ext4** como sistemas de archivos estándares en instalaciones locales del sistema operativo.

6. **Instalación del sistema base.** En este paso se instalan sobre la partición primaria los componentes base que conforman el *kernel* del sistema operativo, que contiene los *drivers* y los procesos de bajo nivel, como las interfaces de usuarios, consolas para ejecución de comandos, y servicios críticos del sistema.

7. **Instalación de paquetes de *software*.** Selección e instalación de los paquetes de *software* que contendrá la instalación.

Una vez este proceso se ha completado, se pueden ejecutar las tareas de chequeo de la instalación que se describen en el siguiente apartado.

2.2.2. Tareas de post-instalación del sistema operativo

Luego de realizar la instalación del sistema operativo, normalmente se recomienda seguir las siguientes tareas de post-instalación:

1. **Chequear el proceso de inicio.** Procediendo a reiniciar el sistema.

2. **Iniciar sesión con el usuario *root*.** Con esta cuenta verificar el funcionamiento del *hardware*: tarjeta de video, interfaces de red (cableada e inalámbricas), puertos de memoria (USB, etc). Dependiendo del ambiente de trabajo, se pueden diseñar procedimientos específicos de chequeo para el *hardware*. Por ejemplo, el modo de conexión a la red (cableado o inalámbrico), la disponibilidad de conexión a Internet para verificación de la navegación, la disponibilidad de impresoras u otros periféricos, etc.

3. Iniciar sesión con el usuario estándar creado en el proceso de instalación. Verificar que el usuario tiene acceso a los paquetes de *software* estándar, y tiene restricciones en la visualización de directorios críticos, como el directorio del superusuario (`/root`) o los directorios de configuración del sistema (`/etc`, `/boot`, `/mnt`, `/lib`, etc.)

Los procedimientos de instalación descritos anteriormente se limitan a la instalación en una máquina individual, con un solo usuario local. En los siguientes apartados se describen en detalle más procesos de instalación y preparación del sistema para el trabajo con más usuarios, trabajo en red, y el ajuste de la seguridad del sistema.

2.3. Usuarios y grupos

La administración de usuarios se trata de interconectar humanos con computadoras. Esto implica una serie de factores a tomar en cuenta:

- **Contabilidad**: registrar nuevos usuarios y eliminar los antiguos.
- **Comodidad y conveniencia**.
- **Servicios de soporte al usuario**.
- **Cuestiones éticas**.
- **Gestión de la seguridad**.

Algunos de estos (registro de cuenta) son tecnológicos, mientras que otros (servicios de soporte) son asuntos humanos. El confort y la comodidad se encuentran en un punto intermedio.

Con el pasar de los años se ha experimentado un resurgimiento del uso de servidores centralizados con cuentas de inicio de sesión para cientos de usuarios, además de los servidores distribuidos con pocas cuentas de usuarios. Los administradores necesitan un conocimiento profundo del sistema de cuentas de usuario para administrar los servicios de red y configurar las cuentas adecuadamente para el entorno informático local. A menudo, la administración de cuentas en servidores es solo una pieza del rompecabezas de aprovisionamiento de cuentas para toda una empresa.

Los entornos empresariales actuales no solo necesitan una herramienta para agregar usuarios a máquinas específicas, sino también una herramienta para administrar usuarios y sus innumerables cuentas y contraseñas en todo el entorno informático. A esto se le conoce como un **sistema de administración de identidad**. Los servicios de directorio como ***Active Directory*** (Microsoft), ***OpenLDAP*** y ***Fedora Directory Server*** son muy populares para funciones de administración de identidades, y probablemente sean la solución adecuada para organizaciones de tamaño medio a grande. Incluso si se utilizan los sistemas centralizados y automatizados para agregar y eliminar usuarios de las estaciones de trabajo, es importante comprender los cambios que están haciendo las herramientas a nivel local en dichas estaciones. Por esta razón, se ofrece en los siguientes apartados información sobre la administración de cuentas con los archivos planos que se deben modificar para agregar usuarios y grupos a máquinas individuales, específicamente en máquinas que empleen sistemas operativos basados en Unix como es el caso de GNU/Linux. Así se desarrollan ejemplos con

comandos como **useradd**, **userdel** y **usermod**, y la información del manejo de los archivos asociados a la administración de usuarios.

2.3.1. El archivo `/etc/passwd`

El archivo `/etc/passwd` es una lista de usuarios reconocidos por el sistema. Puede ser extendido o reemplazado por un servicio de directorio, por lo que es completo y autorizado solo en sistemas independientes (*stand-alone*).

El sistema consulta `/etc/passwd` en el momento de inicio de sesión para determinar el UID (*User ID*) y el directorio de inicio de un usuario, entre otras cosas. Cada línea del archivo representa un usuario y contiene siete campos separados por dos puntos (Nemeth, Snyder, Hein, & Whaley, 2011):

- Nombre de usuario (*Login*)
- Marcador de posición de contraseña cifrada
- Número de UID (*User ID*)
- GID predeterminado (*Group ID*)
- Información "**GECOS**"
- Directorio de inicio (directorio *Home*)
- Shell de inicio de sesión (*Login Shell*)

Por ejemplo, las siguientes líneas son todas entradas válidas en `/etc/passwd`:

```
root:x:0:0:The System,,x6096,:/:/bin/sh
jl:!:100:0:Jim Lane,ECOT8-3,,:/staff/jl:/bin/sh
dotty:x:101:20::/home/dotty:/bin/tcsh
```

Las contraseñas cifradas solían estar en el segundo campo, pero eso ya no es seguro. Con *hardware* relativamente modesto se podrían descifrar en minutos. **Todas las versiones de UNIX y Linux ahora ocultan las contraseñas cifradas al colocarlas en un archivo separado que no es fácilmente legible.** El archivo `/etc/passwd` contiene una x en el campo de contraseña cifrada en Linux, Solaris y HP-UX y un `!` o un * en otros sistemas basados en Unix como AIX de IBM. (En los sistemas AIX, * como marcador de posición deshabilita la cuenta). Las contraseñas cifradas se almacenan realmente en `/etc/shadow` en Linux, Solaris y HP-UX, y en `/etc/security/passwd` en AIX. Los formatos varían.

Si las cuentas de usuario se comparten a través de un servicio de directorio como **NIS** o **LDAP**, es posible que se observen entradas especiales en el archivo `/etc/passwd` que comienzan con **+** o **-**. Estas entradas le indican al sistema cómo integrar los datos del servicio de directorio con el contenido de `/etc/passwd`. Esta integración también se puede configurar en el archivo `/etc/nsswitch.conf`. Más adelante se desarrolla la información sobre los servicios de directorio, haciendo énfasis en El LDAP, como el más popular empleado en sistemas basados en Unix.

En la Tabla 5 se resume el significado de los campos que componen una entrada en el archivo `/etc/passwd`.

Tabla 5
Campos de un registro del archivo /etc/passwd

Campo	Significado y reglas
nombre de usuario (*login*)	Nombres de inicio de sesión (*login* o *username*) Deben ser únicos Pueden tener restricciones de longitud y juego de caracteres permitidos. Para el caso de Linux: • Longitud máxima: 32 caracteres • Conjunto de caracteres: a-z0-9_- • Caracteres iniciales permitidos: a-z_ Son *case-sensitive*
Marcador de posición de contraseña cifrada	Una "x" o un "*" indican que la cuenta ha sido creada y que el usuario debe iniciar sesión para crear una contraseña que será guardada cifrada.
UID (*User ID*)	Identifica al usuario para el sistema. Los nombres de usuario se proporcionan para la conveniencia de los usuarios, pero el *software* y el sistema de archivos usan el UID internamente. Los UID suelen ser enteros de 32 bits sin signo. Por definición, el usuario ***root*** tiene UID 0
GID predeterminado (*Group ID*)	Al igual que un UID, un GID o número de ID de grupo es un entero de 32 bits. **/etc/passwd** proporciona un GID predeterminado (o "efectivo") en el momento de inicio de sesión. Sólo es relevante para la creación de nuevos archivos y directorios GID 0 está reservado para el grupo llamado root o system.
Información "GECOS"	Información adicional del usuario: Nombre completo, oficina, extensión, teléfono de casa.
Directorio de inicio (directorio *Home*)	Es el directorio predeterminado del usuario al momento de iniciar sesión. Si falta este directorio al el momento de inicio de sesión, el sistema imprime un mensaje como "sin directorio de inicio" y coloca al usuario en el directorio raíz "/". En Linux, si **/etc/login.defs** establece **DEFAULT_HOME** en **no**, el inicio de sesión no puede continuar.
Shell de inicio de sesión (*Login Shell*)	Es un intérprete de comandos, como el shell Bourne o el shell C (**/bin/sh** o **/bin/csh**), pero puede ser cualquier programa. **sh** es el valor predeterminado tradicional para UNIX y **bash** (el shell GNU "*Bourne again*") es el valor predeterminado para Linux y Solaris. En sistemas Linux, **sh** y **csh** son realmente solo enlaces a **bash** y **tcsh** (una versión mejorada de **csh**), respectivamente.

Nota. Elaboración propia

Algunos sistemas operativos basados en Unix como AIX permiten ajustar la longitud máxima de los nombres de usuario, o incluso activar el uso de caracteres especiales (para lenguajes asiáticos, por

ejemplo), pero no se recomienda activar esto en pro de la compatibilidad de los sistemas a nivel internacional. En el sistema AIX se puede por ejemplo consultar la configuración de la longitud máxima con el comando `lsattr -D -l sys0`, y para cambiarla se usa el comando `chdev`. En el siguiente ejemplo, después de consultar la longitud máxima del nombre de usuario (9 caracteres) se ha cambiado a un máximo de 16 caracteres.

```
aix$ lsattr -El sys0 -a max_logname
max_logname 9 Maximum login name length at boot time True
aix$ sudo su -
aix# chdev -l sys0 -a max_logname=16
```

En relación al cifrado de las contraseñas, los sistemas admiten una variedad de algoritmos: el tradicional ***crypt*** (basado en DES), **MD5**, **Blowfish** y una versión iterativa de MD5 heredado del proyecto del servidor web Apache. Para el caso de los sistemas Linux la configuración de los algoritmos de cifrado soportados, y otros parámetros de la seguridad para el inicio de sesión, se configura en el archivo `/etc/login.defs`.

El UID 0 es asignado por defecto al superusuario del sistema (*root*). No es una buena idea tener varias cuentas con UID 0. Si bien puede parecer conveniente tener múltiples inicios de sesión raíz con diferentes *shells* o contraseñas, esta configuración sólo crea más brechas de seguridad potenciales y produce múltiples inicios de sesión que necesitarán ser protegidos. Si las personas necesitan tener formas alternativas de iniciar sesión con permisos de administrador, es mejor que utilicen un programa como **sudo**. **Los UID deben mantenerse únicos en toda la organización.** Es decir, un UID particular debe referirse al mismo nombre de usuario y a la misma persona en cada máquina que esa persona está autorizada a usar. No mantener distintos UID puede generar problemas de seguridad con sistemas de archivos en red como **NFS** y también puede generar confusión cuando un usuario se mueve de un grupo de trabajo a otro. Puede ser difícil mantener UID únicos cuando grupos de máquinas son administrados por diferentes personas u organizaciones. Los problemas son tanto técnicos como políticos. La mejor solución es tener una base de datos central o un **servidor de directorio** (como LDAP) que contenga un registro para cada usuario y aplique la unicidad. Un esquema más simple es asignar a cada grupo dentro de una organización un rango de UID y dejar que cada grupo administre su propio conjunto. Esta solución mantiene los espacios de UID separados, pero no resuelve el problema paralelo de nombres de inicio de sesión únicos (Nemeth, Snyder, Hein, & Whaley, 2011).

En el pasado, cuando el poder de cómputo era costoso, **los grupos** se usaban con fines contables para que el departamento correcto pudiera ser contabilizado por sus segundos de tiempo de CPU, minutos de tiempo de inicio de sesión y *kilobytes* de disco utilizados. Hoy en día, los grupos se usan principalmente para compartir el acceso a los archivos entre diferentes usuarios que tienen los mismos niveles de acceso.

El archivo `/etc/group` define los grupos, con el campo GID en `/etc/passwd` que proporciona un GID predeterminado (o "efectivo") **en el momento de inicio de sesión**. El GID predeterminado no se trata especialmente cuando se determina el acceso; solo es relevante para la creación de nuevos archivos y directorios. Los archivos nuevos normalmente son propiedad de su grupo efectivo, pero si se desea

compartir archivos con otros en un grupo de proyecto, se debe recordar cambiar manualmente el propietario del grupo de archivos.

2.3.2. El archivo `/etc/shadow`

Un archivo *shadow* (oculto) de contraseñas sólo puede ser leído por el super usuario y sirve para mantener las contraseñas cifradas a salvo de miradas no autorizadas y programas de descifrado de contraseñas. También incluye información adicional de la cuenta que no se proporciona en el archivo `/etc/passwd`. Las contraseñas ocultas son las predeterminadas en casi todos los sistemas. IBM ubica el archivo que almacena las contraseñas cifradas en `/etc/security/passwd`, mientras que el resto de sistemas basados en Unix lo ubican en `/etc/shadow`. Los formatos y contenidos son, por supuesto, diferentes. En este apartado se muestra la estructura y función del archivo `/etc/shadow`. El archivo *shadow* no es un superconjunto del archivo *passwd*, y el archivo *passwd* no se genera a partir de él. Se deben mantener ambos archivos o utilizar herramientas como **useradd** que mantienen ambos archivos en su nombre. Al igual que `/etc/shadow`, `/etc/shadow` contiene una línea para cada usuario. Cada línea contiene nueve campos, separados por dos puntos (Nemeth, Snyder, Hein, & Whaley, 2011):

- Nombre de usuario (*login*).
- Contraseña cifrada.
- Fecha del último cambio de contraseña.
- Número mínimo de días entre cambios de contraseña.
- Número máximo de días entre cambios de contraseña.
- Número de días de anticipación.
- Fecha de vencimiento de la cuenta.
- Un campo reservado que actualmente está siempre vacío, excepto en Solaris.

Solo los valores para el nombre de usuario y la contraseña son mandatorios. Los campos de fecha absoluta en `/etc/shadow` se especifican en términos de días (no segundos) desde el 1 de enero de 1970, que no es una forma estándar de calcular el tiempo en sistemas Unix o Linux. Sin embargo, puede convertir de segundos a días desde la época de Unix aplicando la siguiente instrucción (sabiendo que hay 8600 segundos en un día: 60*60*24):

```
unix$ expr ‘date +%s’ / 86400
```

Una entrada típica en el achivo `/etc/shadow` luce como la siguiente:

```
millert:$md5$em5J8hL$a$iQ3pXe0sakdRaRFyy7Ppj.:14469:0:180:14:::
```

En la Tabla 6 se ofrece una breve explicación de los campos de un registro de `/etc/shadow`.

Tabla 6
Campos de un registro en /etc/shadow

Campo	Significado (/etc/shadow)
nombre de usuario (*login*)	Es lo mismo que en **/etc/passwd**. Este campo conecta las entradas ***passwd*** y ***shadow*** de un usuario.
Contraseña cifrada	La contraseña cifrada es idéntica en concepto y ejecución a la previamente almacenada en **/etc/passwd**. Por ejemplo, un *password* codificado en MD5 sería el mostrado en el texto: $md5$em5J8hLaiQ3pXe0sakdRaRFyy7Ppj
Fecha del último cambio de contraseña	Registra la hora a la que la contraseña del usuario fue cambiada por última vez. Este campo se modifica con el comando ***passwd***.
Número mínimo de días entre cambios de contraseña	Establece el número de días que deben transcurrir entre los cambios de contraseña. La idea es forzar cambios auténticos evitando que los usuarios vuelvan inmediatamente a una contraseña familiar después de un cambio requerido.
Número máximo de días entre cambios de contraseña	Esta característica permite al administrador forzar la expiración de la contraseña, requiriendo su cambio periódico.
Número de días de anticipación	Establece el número de días antes del vencimiento de la contraseña para que el inicio de sesión comience a advertir al usuario del vencimiento inminente.
Fecha de vencimiento de la cuenta	Especifica el día (en días desde el 1 de enero de 1970) en el que caducará la cuenta del usuario. El usuario no puede iniciar sesión después de esta fecha hasta que un administrador haya restablecido el campo. Si el campo se deja en blanco, la cuenta nunca caducará.

Nota. Elaboración propia.

2.3.3. El archivo `/etc/group`

El archivo `/etc/group` contiene los nombres de los grupos de usuarios y una lista de los miembros de cada grupo. Una porción del contenido de este archivo puede lucir como la siguiente:

```
system:!:0:root,pconsole,esaadmin
staff:!:1:ipsec,esaadmin,trent,ben,garth,evi
bin:!:2:root,bin
sys:!:3:root,bin,sys
adm:!:4:bin,adm
nobody:!:4294967294:nobody,lpd
```

Cada línea representa un grupo y contiene cuatro campos:

- Nombre del grupo
- Contraseña cifrada o un marcador de posición
- Número GID
- Lista de miembros, separados por comas (sin espacios)

Como en `/etc/passwd`, los campos están separados por dos puntos. Los nombres de los grupos deben estar limitados a ocho caracteres por compatibilidad, aunque muchos sistemas en realidad no lo requieren. Al igual que con los nombres de usuario y UID, los nombres de grupo y GID deben mantenerse consistentes entre las máquinas que comparten archivos a través de un sistema de archivos de red. La consistencia puede ser difícil de mantener en un entorno heterogéneo ya que diferentes sistemas operativos podrían usar diferentes GID para los mismos nombres de grupo. La mejor manera de abordar este problema es evitar el uso de un grupo del sistema como el grupo de inicio de sesión predeterminado para un usuario. Algunos sistemas usan la propiedad del grupo junto con los bits de permiso para controlar la ejecución de comandos. Las inconsistencias de GID entre los sistemas pueden causar problemas para actualizar e instalar *software* en todo el sistema en su conjunto.

Si un usuario usa de manera predeterminada un grupo en particular en `/etc/passwd` pero no parece estar en ese grupo según `/etc/group`, `/etc/passwd` gana el argumento. Las membresías grupales otorgadas al momento del inicio de sesión son realmente la unión de las que se encuentran en los archivos *passwd* y *group*.

Tradicionalmente Unix agrega nuevos usuarios a un grupo que representa su categoría general, como "estudiantes" o "finanzas". Sin embargo, esta convención aumenta la probabilidad de que los usuarios puedan leer los archivos de los demás debido a la configuración de "**permisos deslizantes**", incluso si esa no es realmente la intención del propietario de los archivos. Para evitar este problema, la recomendación es **crear un grupo único para cada usuario**. Se suele usar el mismo nombre tanto para el usuario como para el grupo. También se puede hacer que el GID sea igual al UID.

El comando **useradd** en todas las distribuciones de Linux (excepto SUSE), por defecto coloca a los usuarios en sus propios grupos personales. Los sistemas Unix predeterminan poner a todos los usuarios nuevos en el mismo grupo, pero sus niveles de uso también pueden configurarse para admitir grupos personales. El grupo personal de un usuario debe contener solo ese usuario. Si se desea permitir que los usuarios compartan archivos a través del mecanismo de grupo, la recomendación es crear grupos separados para ese propósito. La idea detrás de los grupos personales no es desalentar el uso de grupos en sí, es simplemente establecer un grupo predeterminado más restrictivo para cada usuario para que los archivos no se compartan inadvertidamente.

Linux, Solaris y HP-UX proporcionan comandos que crean, modifican y eliminan grupos: **groupadd**, **groupmod**, **groupdel**.

2.3.4. Añadir usuarios en el sistema (Unix/Linux)

Mecánicamente, y basándose en la arquiectura de un sistema Unix/Linux *stand-alone*, el proceso de añadir usuarios al sistema consistiría en los siguientes pasos generales:

- Hacer que el nuevo usuario firme un acuerdo de política de uso de las cuentas de usuario (una formalidad que no está de más).
- Editar los archivos `/etc/passwd` y `/etc/shadow` para definir la cuenta del usuario.

- Agregar el usuario al archivo `/etc/group` (no es realmente necesario, pero no está de más).
- Establecer una contraseña inicial.
- Crear, modificar el directorio de inicio del usuario.
- Configurar roles y permisos.

Para el usuario:

- Copiar los archivos de inicio predeterminados al directorio de inicio del usuario.
- Establecer el correo de inicio del usuario y el alias de correo.

Para el administrador del sistema:

- Verificar que la cuenta esté configurada correctamente.
- Agregar la información de contacto del usuario y el estado de la cuenta a su base de datos.

Afortunadamente existen herramientas como **useadd** o **adduser** que realizan gran parte de las tareas arriba descritas al estilo de un *script*. Estas herramientas son configurables, para cambiar los parámetros por defecto en la creación de las cuentas de usuarios. Por ejemplo, si se desea cambiar la ruta `/home` para directorios de inicio, la localización de las variables de ambiente, el rango de UID y GID, reglas para los nombres de usuarios y grupos (*Regex-based match*, o reconocimiento por expresiones regulares), entre otros. Se tienen las siguientes localizaciones estándar de archivos de configuración para personalizar cada utilidad:

- **useradd:** `/etc/default/useradd` y `/etc/login.defs`
- **adduser:** `/etc/adduser.conf`

Estos comandos (y todos los comandos de admnistración del sistema), deben ser ejecutados con cuentas de usuario con nivel de administración, o con la cuenta del superusuario (***root***). Si los comandos se ejecutan con cuentas de administración diferentes del superusuario (lo recomendable), típicamente se tendrá que preceder el comando en cuestión con el comando **sudo** (*superuser do*). Un ejemplo de ejecución de **useradd** en un sistemas Linux para añadir al usuario "diego", sería como el siguiente:

```
$sudo useradd diego
```

El comando creará una entrada como la siguiente en el archivo `/etc/passwd`:

```
diego:x:1005:20::/home/diego:/bin/sh
```

useradd desactiva la cuenta al poner una "x" en el campo de contraseña. Se debe asignar una contraseña real para que la cuenta sea utilizable.

Un ejemplo más completo se muestra a continuación. Se especifica que el grupo primario de miguel debe ser "facultad" y que también debe ser agregado al grupo "famoso". Se cambia la ubicación predeterminada (`/home/diego`) del directorio de inicio y el *shell*, y se le pide a **useradd** que cree el directorio de inicio si aún no existe:

```
$sudo useradd -c "Diego Torres" -d /home/math/diego -g facultad -G famoso -m
-s /bin/tcsh diego
```

Este comando crea la siguiente entrada en `/etc/passwd`:

```
diego:x:1005:30:Diego Torres:/home/math/diego:/bin/tcsh
```

El UID asignado es uno más que el UID más alto ya ocupado en el sistema, y la entrada **shadow** correspondiente es

```
diego:!:14322:0:99999:7:0::
```

Los caracteres de marcador de posición de contraseña en el archivo **passwd** y **shadow** varían según el sistema operativo. **useradd** también agrega *diego* a los grupos apropiados en `/etc/group`, crea el directorio `/home/math/diego` y lo inicializa desde el directorio `/etc/skel` que contiene los archivos de inicio estándares del sistema (`.profile` y `.bashrc`).

Si bien **useradd** o **adduser** ayudan en la ejecución de la mayoría de los pasos para crear un usuario nuevo, será bueno tomar en cuenta las siguientes utilidades adicionales para completar y verificar los elementos de la nueva cuenta:

- **Comando passswd:** especificar o cambiar el *password* para el nuevo usuario. La sintanxis ejecutando desde una cuenta distinta al superusuario con el comando `sudo` sería algo como

  ```
  $sudo passwd newusername
  ```

- **Verificar la estructura del directorio de inicio del usuario** (en **/home**), mediante el comando **ls -a**. De esta maneta se deberán visualizar los archivos de inicio predeterminados (**.profile** o **.bashrc**) que serán archivos ocultos en dicho directorio (comienzan con un "**.**").

- Una vez que el directorio de inicio esté configurado, **asegurarse de que los permisos son apropiados**. El comando **chown** ayuda con esto, cambiando de forma recursiva la propiedad de todos los archivos del directorio de inicio a *newuser* y *newgroup*:

  ```
  $ sudo chown -R newuser:newgroup ~newuser
  ```

- **Salir de la cuenta y volver a entrar con el nombre de usuario y contraseña configurados.** Con esto se verificará que las credenciales han sido configuradas correctamente. Luego de iniciar sesión ejecutar los siguientes comandos:

  ```
  $ pwd /* Para verificar el directorio de inicio */
  ```

  ```
  $ ls -la /* Para verificar el propietario/grupo de los archivos de inicio
  (ocultos) */
  ```

Se deberá notificar a los nuevos usuarios sus nombres de inicio de sesión y sus contraseñas iniciales. Muchos sitios envían esta información por correo electrónico, pero por razones de seguridad, generalmente no es una buena idea. Es preferible hacerlo en persona o por teléfono, a menos que esté agregando una cantidad importante de usuarios en lotes.

Recordar a los nuevos usuarios que cambien sus contraseñas de inmediato. Se puede aplicar esto configurando la contraseña para que caduque en poco tiempo. Otra opción es hacer que un *script* verifique a los nuevos usuarios y asegurarse de que sus contraseñas encriptadas en el archivo `/etc/shadow` hayan cambiado.

2.3.5. Remover o deshabilitar usuarios del sistema (Unix/Linux)

Cuando un usuario abandona la organización, la cuenta de inicio de sesión y los archivos de ese usuario deben eliminarse del sistema (luego de haber realizado el correspondiente respaldo de los archivos de interés). Este procedimiento implica la eliminación de todas las referencias al nombre de inicio de sesión que se hayan creado mediante los métodos descritos en el apartado anterior (**useradd**, **adduser**, o manualmente). Si se elimina un usuario manualmente, se recomienda una lista de verificación como la siguiente (Nemeth, Snyder, Hein, & Whaley, 2011):

- Eliminar al usuario de las bases de datos de usuarios locales o listas de teléfonos.
- Eliminar al usuario del archivo de alias o agregar una dirección de reenvío.
- Eliminar el archivo **crontab** del usuario y cualquier orden pendiente en *jobs* o *print jobs*.
- Eliminar cualquiera de los procesos del usuario que aún se están ejecutando.
- Eliminar al usuario de los archivos **passwd**, **shadow**, **group** y **gshadow**.
- Eliminar el directorio de inicio del usuario (`/home/usuario` por defecto).
- Eliminar la cola de correo del usuario.
- Limpiar entradas en calendarios compartidos, sistemas de reserva de habitaciones, etc.
- Eliminar o transferir la propiedad de cualquier lista de correo ejecutada por el usuario eliminado.

Una vez que se haya eliminado una cuenta de usuario, es necesario verificar que el UID de dicho usuario ya no posee archivos en el sistema. Para encontrar las rutas de los archivos "huérfanos", se puede usar el comando *find* con el argumento *-nouser*. Debido a que *find* tiene una forma de "escapar" a los servidores de red si no tiene cuidado, generalmente es mejor verificar los sistemas de archivos individualmente con `-xdev`:

```
$ sudo find filesystem -xdev -nouser
```

Si en la organización se asignan estaciones de trabajo individuales a los usuarios, generalmente es más simple y eficiente volver a crear una imagen de todo el sistema desde una plantilla maestra antes

de asignar el sistema a un nuevo usuario. Sin embargo, antes de realizar la reinstalación, es una buena idea hacer un **respaldo de los archivos locales** en el disco duro del sistema en caso de que sean necesarios en el futuro.

En los sistemas basados en Unix/Linux existe el comando **userdel** que automatiza el proceso de eliminación de un usuario. Probablemente no se ejecutará un trabajo tan completo como se esperaría, a menos que se haya agregado religiosamente toda la información y funcionalidades de los lugares donde se haya almacenado información relacionada con el usuario.

En ocasiones, la cuenta de un usuario debe **deshabilitarse temporalmente**. Una forma sencilla de hacer esto es colocar una almohadilla (#) u otro carácter delante de la contraseña cifrada del usuario en el archivo `/etc/shadow`. Esta medida evita la mayoría de los tipos de acceso regulado por contraseña porque la contraseña ya no se descifra a nada sensato. Sin embargo, los comandos de acceso remoto como **ssh** que no verifican necesariamente la contraseña del sistema pueden continuar funcionando.

Los comandos Linux `usermod -L` *`user`* y `usermod -U` *`user`* proporcionan una manera fácil de bloquear (*Lock*) y desbloquear (*Unlock*) contraseñas, respectivamente. Son solo atajos para el bloqueo de contraseñas descrito anteriormente: el `-L` pone un ! delante de la contraseña cifrada en el archivo `/etc/shadow`, y `-U` la elimina.

2.3.6. Centralización de la administración de usuarios: LDAP

Alguna forma de administración centralizada de cuentas de usuario es esencial para las empresas medianas y grandes de todo tipo, ya sean corporativas, académicas o gubernamentales. Los usuarios necesitan la conveniencia y la seguridad de un solo nombre de usuario, UID y contraseña en todo el sitio. Los administradores necesitan un sistema centralizado que permita que los cambios (como las revocaciones de cuentas) se propaguen instantáneamente en todas partes. Dicha centralización se puede lograr de varias formas, la mayoría de las cuales involucran de alguna manera la utilización del protocolo **LDAP** (*Lightweight Directory Access Protocol*) (Nemeth, Snyder, Hein, & Whaley, 2011). Este protocolo es utilizado por la mayoría de los servicios de directorio para la administración centralizada de cuentas de usuario y recursos en la infraestructura tecnológica, como el famoso sistema *Microsoft Active Directory*.

LDAP es un repositorio generalizado similar a una base de datos que puede almacenar datos de administración de usuarios, así como otros tipos de datos. Utiliza un modelo cliente/servidor jerárquico que admite múltiples servidores, así como múltiples clientes simultáneos. Una de las grandes ventajas de LDAP como repositorio de datos de inicio de sesión para todo el sitio es que puede aplicar UID y GID únicos en todos los sistemas. También es compatible con los sistemas basados en Windows, aunque lo contrario es solo marginalmente cierto.

Microsoft Active Directory usa **LDAP** y **Kerberos** y puede administrar muchos tipos de datos, incluida la información del usuario. Es un poco egoísta y quiere ser el dominante si está interactuando con repositorios LDAP de Unix o Linux. Si se necesita un único sistema de autenticación para un sitio que incluye escritorios de Windows, así como sistemas Unix y Linux, probablemente sea más fácil dejar que *Microsoft Active Directory* tenga el control y usar las bases de datos LDAP Unix como servidores secundarios.

LDAP es una base de datos, por lo que la información almacenada allí debe ajustarse a un **esquema** bien definido. Los esquemas se expresan como archivos XML, con los nombres de campo procedentes de los RFC relevantes, principalmente el RFC 2307 para datos de gestión de usuarios.

Los sistemas centralizados de inicio de sesión (*Single-Sign-On*, SSO) equilibran la conveniencia del usuario con los asuntos de seguridad. La idea es que un usuario pueda iniciar sesión una vez (en una pantalla de inicio de sesión, una página web o un cuadro de diálogo de Windows) y autenticarse en ese momento. Luego, el usuario obtiene las **credenciales de autenticación** (generalmente implícitamente, de modo que no se requiere una administración activa), que luego se pueden utilizar para acceder a otras máquinas, aplicaciones o recursos en la red o el sitio. El usuario sólo tiene que recordar una secuencia de inicio de sesión y contraseña en lugar de muchas.

Este esquema permite que las credenciales sean más complejas (ya que el usuario no necesita recordarlas ni tratarlas), lo que teóricamente aumenta la seguridad. Sin embargo, el impacto de una cuenta comprometida es mayor porque un inicio de sesión le da al atacante acceso a múltiples recursos de la infraestructura tecnológica. Los sistemas SSO hacen que alejarse de una máquina de escritorio mientras todavía está conectado sea una vulnerabilidad significativa. Además, el servidor de autenticación se convierte en un cuello de botella crítico. Si está inactivo, todo el trabajo útil se detiene en toda la organización.

En el siguiente capítulo se desarrolla con más detalle este tema de la administración centralizada y los sistemas heterogéneos, siendo los servicios de directorios basados en LDAP uno de los temas centrales.

2.4. Configuración de la red

La creación de redes plantea problemas para la administración del sistema en muchos niveles, desde su implementación hasta su configuración y uso.

Existen muchos protocolos de redes para interconectar las máquinas en un sitio, de los cuales los protocolos TCP/IP (protocolos de Internet) son los dominantes, y en los que se basarán las explicaciones de este apartado y del resto del material.

La versión de TCP/IP que ha estado en uso generalizado durante las últimas décadas es la versión 4 del protocolo, también conocida como IPv4. Utiliza direcciones IP de cuatro bytes (32 bits). Una versión modernizada, IPv6, amplía el espacio de direcciones IP a 16 bytes (128 bits), e incorpora varias mejoras respecto de IPv4. Elimina varias características de IP que en la experiencia han demostrado tener poco valor, lo que hace que el protocolo sea potencialmente más rápido y fácil de implementar. IPv6 también integra seguridad y autenticación en el protocolo básico.

2.4.1. Configuración básica de la red

Solo se requieren unos pocos pasos para agregar una nueva máquina a una red de área local existente, pero cada sistema lo hace de manera diferente. **Los sistemas suelen proporcionar una interfaz gráfica de usuario (un panel de control) para la configuración básica de la red, pero las configuraciones más elaboradas (o automatizadas) pueden requerir la edición de archivos de configuración.**

Los pasos básicos para agregar una nueva máquina a una red local son los siguientes:

- Asignar una dirección IP y un nombre de *host* únicos.
- Verificar que las interfaces de red estén configuradas correctamente en el momento del arranque.
- Configurar una ruta predeterminada y quizás algunas rutas específicas.
- Configurar el apuntador a un servidor de nombres DNS para permitir el acceso al resto de Internet.

Si se tiene un servidor DHCP para el aprovisionamiento automático de las máquinas que ingresan en la red, la mayoría de las tareas de configuración para una nueva máquina se realizan en el servidor DHCP en lugar de en la nueva máquina en sí. Por lo general, las nuevas instalaciones del sistema operativo obtienen su configuración a través de DHCP, por lo que las máquinas nuevas pueden no requerir configuración de red explícita.

En esta sección se suponen resueltos todos los temas relativos a la **instalación de la red física**, y más bien se ofrece información sobre el proceso general de **configuración y pruebas de la red** sobre una máquina del sitio, indicando varios comandos de sistemas Unix/Linux involucrados en dicho proceso.

En el proceso de configuración básica de la red en una máquina del sistema se destacan las siguientes tareas:

Asignación de dirección IP y *hostname*

Los parámetros de **direcciones IP** y **nombres de *host*** (*hostname*) en las máquinas del sistema podrían ser variables, por lo que, **en un ambiente con muchas máquinas para administrar, convienen los esquemas centralizados automatizados, en lugar de realizar asignaciones manuales uno a uno**. Para acelerar la gestión de posibles cambios en el direccionamiento, se puede usar nombres de *host* en archivos de configuración y limitar las asignaciones de direcciones a algunas ubicaciones centralizadas, como la **base de datos DNS** y sus archivos de configuración DHCP.

En un sitio pequeño, se pueden configurar fácilmente nombres de *host* y direcciones IP de forma manual. Pero cuando intervienen muchas redes y muchos grupos administrativos diferentes, ayuda tener una coordinación central para garantizar la unicidad. Para los parámetros de red asignados dinámicamente, DHCP se ocupa de los problemas de singularidad. Algunos sitios usan bases de datos LDAP para administrar sus nombres de host y asignaciones de direcciones IP de forma centralizada. Este tema será desarrollado en el siguiente capítulo.

A continuación, se muestra un ejemplo de configuración manual de un host en Linux con los comandos **ifconfig** y **route**, para asignar una dirección IP y una ruta predeterminada.

```
unix$ifconfig eth0 192.168.1.13 netmask 255.255.255.0 up
unix$route add -net 0.0.0.0/0 gw 192.168.1.254
```

En el ejemplo anterior se ha configurado una interfaz llamada **eth0**, con la dirección IP y máscara indicadas. También se ha colocado una ruta predeterminada (hacia la red 0.0.0.0/0) a través del *router* o *gateway* **192.168.1.254**. Una vez realizada esta configuración, se puede verificar de la siguiente manera:

```
unix$ ifconfig eth0
eth0 Link encap:Ethernet HWaddr 00:02:B3:19:C8:86
inet addr:192.168.1.13 Bcast:192.168.1.255 Mask:255.255.255.0
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:206983 errors:0 dropped:0 overruns:0 frame:0
TX packets:218292 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:100
Interrupt:7 Base address:0xef00

unix$ route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.1.254   0.0.0.0         UG    0      0        0 eth0
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 eth0
```

Con los comandos **ifconfig** y **route**, aparte de asignar una dirección IP como tal a la interfaz deseada o configurar rutas, respectivamente, también se puede usar para **consultar** la configuración. En el ejemplo mostrado se observa que, a parte de la configuración del direccionamiento, también se obtienen estadísticas de transmisión como las de los paquetes recibidos (*RX packets)* o transmitidos (*TX packets*), colisiones de paquetes, entre otros, para el caso del comando **ifconfig**.

2.4.2. Configuración avanzada de la red

En el apartado anterior se mostraron ejemplos de la configuración de la red a través de comandos. Esta configuración no permanecerá en el sistema una vez este es reiniciado, por lo que habría que ejecutar nuevamente dichos comandos para conectar el sistema a la red.

Para que la configuración se realice de forma automática con cada reinicio del sistema, existen diferentes métodos, de los cuales los más populares son la **configuración estática** y la **configuración dinámica** con protocolos como DHCP (*Dynamic Host Configuration Protocol*).

La **configuración estática** consiste en la escritura sobre un archivo de configuración de los parámetros de red que se desea para el sistema en cada reinicio. La sintaxis y ubicación de dicho archivo varía según la distribución de Unix/Linux sobre la que se trabaje. Para el caso de las distribuciones basadas en Debian/Ubuntu, el archivo es `/etc/network/interfaces`. Un ejemplo de configuración puede lucir como el siguiente, para la configuración estática:

```
auto eth0
    iface eth0 inet static
        address 192.0.2.7/24
        gateway 192.0.2.254
```

Por otro lado, se deben configurar también de forma estática los servidores de nombre (DNS), en el archivo `/etc/resolv.conf`. Un ejemplo de configuración puede tener las siguientes líneas, indicado dos servidores DNS:

```
nameserver 8.8.8.8
nameserver 8.8.4.4
```

En cuanto a la **configuración dinámica vía DHCP**, ésta se realiza en un servidor centralizado que es contactado por los *hosts* cuando se conectan a la red, a través de mensajes de *broadcast* de solicitud de configuración. El servidor debe está en el mismo segmento de red o la misma VLAN (la misma capa 2) para poder recibir dichas solicitudes y entregar las configuraciones. Entre los parámetros de red que se entregan vía DHCP están:

- Direcciones IP y máscaras de red
- Puertas de enlace (rutas predeterminadas)
- Servidores de nombres DNS
- Syslog hosts
- Servidores WINS, servidores *proxy*, servidores NTP
- Servidores TFTP (para cargar una imagen de arranque)

Existen muchos paquetes de *software* que se pueden instalar en in servidor DHCP con el sistema operativo Unix/Linux, que estarían atendiendo hosts con cualquier sistema operativo compatible con este protocolo, como Windows. MAC-OS o Unix/Linux. El paquete del servidor se llama **dhcpd** en Red Hat, **dhcp3-server** en Debian/Ubuntu y **dhcp-server** en SUSE. Tomando como ejemplo la configuración de **dhcpd**, el archivo de configuración correspondiente (`/etc/dhcpd.conf`) puede lucir como el siguiente:

```
# global options
option domain-name "synack.net";
option domain-name-servers gw.synack.net;
option subnet-mask 255.255.255.0;
default-lease-time 600;
max-lease-time 7200;
subnet 192.168.1.0 netmask 255.255.255.0 {
range 192.168.1.51 192.168.1.60;
option broadcast-address 192.168.1.255;
option routers gw.synack.net;
}


subnet 209.180.251.0 netmask 255.255.255.0 {
}
```

```
host gandalf {
hardware ethernet 08:00:07:12:34:56;
fixed-address gandalf.synack.net;
}
```

2.5. Seguridad

Si los sistemas van a alojar datos confidenciales y controlar procesos críticos, entonces se deben proteger. Los aspectos de seguridad en la administración de sistemas son un tópico ineludible, puesto que normalmente se trabaja en ambientes conectados y multiusuario. Dicha protección requiere recursos, tanto en términos de tiempo de administración como en el costo de los equipos relacionados con la seguridad. Desafortunadamente, muchas organizaciones no hacen las inversiones apropiadas en esta área hasta que ya se ha producido un incidente.

Abordar este problema no es tan fácil como podría pensarse. La seguridad no es algo que se pueda comprar en una caja, o como un servicio de un tercero. Los productos y servicios comerciales pueden ser parte de una solución, pero no son una panacea. Lograr un nivel aceptable de seguridad requiere una enorme cantidad de paciencia, vigilancia, conocimiento y persistencia, no solo del administrador del sistema, sino de toda su comunidad de usuarios y administración.

El administrador del sistema debe asegurarse personalmente que sus sistemas sean seguros, que estén monitoreados de manera constante, y que los usuarios estén debidamente educados. Es obligatorio familiarizarse y mantenerse al día con la tecnología de seguridad, monitorear activamente las listas de correo de seguridad y contratar expertos en seguridad para asesorar en la atención de problemas que normalmente exceden el conocimiento del administrador del sistema, dado el alto dinamismo con el que estos cambios ocurren.

El tópico de seguridad de sistemas es un tema muy amplio, que abarca desde la seguridad física de los sitios donde funcionan, hasta complejas consideraciones sobre procesos y políticas de seguridad en la organización, pasando por aspectos meramente técnicos como las configuraciones de la seguridad en los dispositivos de red, servidores, y estaciones de trabajo.

Este apartado se centra es un aspecto muy específico de la seguridad, que es la seguridad en el sistema operativo, y más específicamente para equipos con sistemas basados en Unix/Linux. Para estudiar aspectos de seguridad en otros sistemas operativos, como los basados en Microsoft Windows o MAC OS-X, se recomienda revisar los enlaces de interés y demás lecturas complementarias de la asignatura.

2.5.1. Aspectos generales de seguridad de sistemas

Si bien en este apartado se tratan aspectos específicos de la seguridad de sistemas operativos, existen **aspectos generales sobre la seguridad** que es bueno tomar en cuenta para todos los casos de administración de sistemas. A continuación, se enumeran algunos que se consideran importantes para tomar en cuenta en la instalación y operación de los sistemas, junto con sus medidas estándares de protección (Nemeth, Snyder, Hein, & Whaley, 2011):

- **Ingeniería Social - Los usuarios y administradores de un sistema informático son los eslabones más débiles de la cadena de seguridad.** Incluso en el mundo actual de mayor conciencia de seguridad, los usuarios desprevenidos con buenas intenciones se convencen fácilmente para regalar información confidencial. Ninguna cantidad de tecnología puede proteger contra el elemento del usuario: debe asegurarse de que su comunidad de usuarios tenga un alto conocimiento de las amenazas de seguridad para que puedan ser parte de la defensa. Este problema se manifiesta en muchas formas. Los atacantes pueden realizar llamadas a sus víctimas y hacerse pasar por usuarios legítimamente confundidos en un intento de obtener ayuda para acceder al sistema. Los administradores publican involuntariamente información confidencial en foros públicos cuando resuelven problemas. Se producen compromisos físicos cuando el personal de mantenimiento aparentemente legítimo vuelve a cablear el gabinete de red. El término "*phishing*" describe los intentos de recopilar información de los usuarios a través de correos electrónicos engañosos, mensajes instantáneos o incluso mensajes SMS. El *phishing* puede ser especialmente difícil de defender porque las comunicaciones a menudo incluyen información específica de la víctima que les da la apariencia de autenticidad legal. La ingeniería social sigue siendo una poderosa técnica de piratería y es una de las amenazas más difíciles de neutralizar. La política de seguridad de su sitio debe incluir capacitación para nuevos empleados. Las comunicaciones regulares en toda la organización son una forma efectiva de proporcionar información sobre qué hacer y qué no hacer por teléfono, seguridad física, *phishing* de correo electrónico y selección de contraseña. **Para calibrar la resistencia de una organización a la ingeniería social, se pueden intentar algunos ataques de ingeniería social simulados.** Muchas organizaciones encuentran útil comunicar a los usuarios que los administradores nunca solicitarán sus contraseñas u otros datos confidenciales, ya sea por correo electrónico, mensaje instantáneo o por teléfono. Se debe indicar a los usuarios que informen de cualquier solicitud de contraseña al departamento de IT de inmediato.

- **Vulnerabilidades de *software*** - Se han descubierto innumerables errores de seguridad en el *software* de la computadora (incluido el *software* de terceros, tanto comerciales como gratuitos). **Al explotar errores sutiles de programación o dependencias de contexto, los hackers han podido manipular los sistemas para que hagan lo que quieran.** Los desbordamientos del búfer son un error de programación común y uno con implicaciones complejas, en los cuales los atacantes pueden aprovechar para acceder a los espacios de memoria de los equipos y deshabilitar su uso, o incluso, ejecutar código malicioso. Los desbordamientos de búfer son una subcategoría de una clase más grande de errores de seguridad de *software* conocidos como vulnerabilidades de validación de entrada, en las que los atacantes aprovechan la solicitud de entrada de datos para ingresar código malicioso. Normalmente este tipo de ataques es difícil de prevenir, como muchas vulnerabilidades de *software*, las cuales una vez descubiertas (lamentablemente luego de un ataque, en muchos casos) se corrigen y se reparan con la aplicación de parches en el sistema. Esta será entonces una de las formas estándar de protegerse frente a las vulnerabilidades de *software*, el monitoreo constante de sus actualizaciones y las aplicaciones de los parches recomendados por sus proveedores.

- **Errores de configuración** - Muchas piezas de *software* se pueden configurar de forma "segura" o "no tan segura". Desafortunadamente, debido a que el *software* está desarrollado para ser útil

en lugar de molesto, **el *software* "no tan seguro" será lo más común**. Los piratas informáticos con frecuencia obtienen acceso explotando funciones de *software* que se considerarían útiles y convenientes en circunstancias menos traicioneras: cuentas sin contraseñas, discos compartidos con el mundo y bases de datos desprotegidas, por nombrar algunos. La práctica de configurar sistemas sin contraseñas de acceso, o dejar algunos puertos de red de libre acceso, en algunos casos podría resultar conveniente, si esta es necesaria para la usabilidad de dicho sistema, sin embargo, esto se debe hacer de forma consciente, y habrá que pensar en alguna forma de que no represente una brecha de seguridad mediante algún mecanismo alternativo de protección. Los problemas en esta categoría son los más fáciles de encontrar y solucionar, aunque potencialmente hay muchos de ellos y no siempre es obvio qué buscar. **Las herramientas de escaneo de puertos y vulnerabilidades** pueden ayudar a un administrador motivado a identificar problemas antes de que sean explotados.

2.5.2. Seguridad de la configuración de red

Son varios los aspectos de configuración de la red en el sistema que están relacionados con la seguridad del mismo. En la Tabla 7 se resumen los más comunes por su aparición en incidentes de seguridad, a tomar en cuenta durante la instalación y operación de los sistemas operativos basados en Unix/Linux, principalmente para las estaciones de trabajo.

Tabla 7
Características de seguridad en la configuración de red (Sistemas Unix/Linux)

Característica de red	Implicaciones en la seguridad y ajustes
Reenvío de paquetes (*IP forwarding*)	Un sistema Unix/Linux con la bandera de reenvío de paquetes activada (*IP Forwarding*) puede actuar como *router*. A menos que el sistema tenga varias interfaces de red y se suponga que funcione como un *router*, es recomendable desactivar esta función. Los *hosts* que reenvían paquetes a veces pueden verse obligados a comprometer la seguridad al hacer que los paquetes externos parezcan provenir de su red. Es perfectamente aceptable que un *host* use múltiples interfaces de red para su propio tráfico sin reenviar el tráfico de terceros entre diferentes segmentos.
Redireccionamiento de mensajes ICMP (*ICMP redirects*)	Los redireccionamientos ICMP pueden redirigir maliciosamente el tráfico y alterar las tablas de enrutamiento. La mayoría de los sistemas operativos escuchan los redireccionamientos ICMP y siguen sus instrucciones por defecto. Se podría ser víctima de un ataque en el que el tráfico sea redirigido a la red de un competidor durante unas horas, por ejemplo, mientras se ejecutan las copias de seguridad. Se recomienda configurar los enrutadores (y hosts que actúan como enrutadores) para ignorar y quizás registrar los intentos de redireccionamiento ICMP.
Enrutamiento de origen (*Source routing*)	El enrutamiento de origen omite el algoritmo estándar de enrutamiento con el próximo salto que normalmente se ejecuta en cada *router* para determinar cómo se debe reenviar un paquete. El objetivo original de esta funcionalidad es facilitar pruebas, pero puede crear problemas de seguridad porque los paquetes a menudo se filtran según su origen. Si alguien puede enrutar hábilmente un paquete para que parezca que se originó dentro de su red en lugar de Internet, podría pasar por su *firewall*. Se recomienda no aceptar ni reenviar paquetes enrutados en origen.
Pings a *broadcast* (y otros *broadcast* controlados)	Los paquetes o mensajes de *broadcast* (como los pings de *broadcast*) se utilizan en **ataques de denegación de servicio (DoS)**. Por ejemplo, los llamados ataques pitufos (*Smurf attacks*). Se debe desactivar la respuesta a pings de *broadcast* y otros tipos de *broadcast* dirigidos en las interfaces de red.

Característica de red	Implicaciones en la seguridad y ajustes
IP spoofing	El *IP spoofing* (suplantación de identidad en IP) consiste en cambiar la IP origen en un paquete IP para suplantar la identidad del emisor del paquete, que es la víctima del ataque. Los paquetes de error y devolución pueden interrumpir o inundar las conexiones de red de la víctima. Una heurística conocida como "reenvío de ruta inversa de unidifusión" (*unicast Reverse Path Forwarding*, uRPF) ayuda con esto. Hace que las puertas de enlace IP descarten los paquetes que llegan a una interfaz que es diferente de la que se transmitirían si la dirección de origen fuera el destino. Es un control de seguridad rápido que utiliza la tabla de enrutamiento IP normal como una forma de validar el origen de los paquetes de red. Los enrutadores dedicados implementan uRPF, pero también lo hace el *kernel* de Linux. En Linux, está habilitado de forma predeterminada.
Firewalls* instalados en los *hosts	En general las tareas de un *firewall* suelen ser muy complejas para realmente proteger un dispositivo frente a la inmensa cantidad de ataques que existen. Normalmente no se recomienda dejar esta tarea a un *software* que intente filtrar paquetes en un *host*, pues el tema va más allá de eso. Lo mejor es contar con dispositivos de red especiales para que actúe como *firewalls* y protejan la red y la infraestructura en su totalidad.
VPN	Existe *software* que se puede instalar en el sistema operativo y permite a los usuarios establecer túneles cifrados para VPN (*Virtual Private Network*), con protocolos como TLS/SSL, con lo cual estarían eventualmente saltando las restricciones de seguridad que se configuren en los *firewalls* de la red. Se recomienda bloquear la posibilidad de instalación de este tipo de software en las estaciones de trabajo, y mantener activas alarmas en el monitoreo de la red para detectar tráfico cifrado no permitido.

Nota. Elaboración Propia.

2.5.3. Consejos generales de protección del sistema

A manera de lista de chequeo, a continuación, se presentan en algunos consejos a tomar en cuenta para la protección de los sistemas operativos, después de su instalación y durante su operación. El administrador del sistema podría aplicar dicha lista durante la configuración, y un administrador de seguridad luego confirmar que la lista se aplicó correctamente, y mantener un registro de los sistemas recién asegurados.

Tabla 8
Lista de chequeo: consejos para la adecuada protección del sistema

Consejo de seguridad	Descripción
Monitoreo y aplicación de parches	Mantener el sistema actualizado con los últimos parches es la tarea de seguridad de mayor valor de un administrador. La mayoría de los sistemas están configurados para apuntar al repositorio del proveedor, lo que hace que la aplicación de parches sea tan simple como ejecutar algunos comandos. Los entornos más grandes pueden usar un repositorio local que refleje el del proveedor. Considerar: • Un cronograma de actualizaciones con la suficiente periodicidad para mantenerse actualizado con las vulnerabilidades detectadas, y a ser aplicado en horarios que no impacte las operaciones de los usuarios. • Aplicar un **procedimiento de control de cambios**, en el que se documente el impacto de la aplicación del parche y las medidas a tomar en caso de necesitar revertir la actualización. • Mantener un inventario preciso de aplicaciones y sistemas operativos utilizados en la organización, para garantizar una cobertura completa de la aplicación de parches.

Consejo de seguridad	Descripción
Deshabilitación de servicios innecesarios	La mayoría de los sistemas vienen con varios servicios configurados para ejecutarse de manera predeterminada. Asegurarse de deshabilitar (y posiblemente eliminar) los que sean innecesarios, especialmente si son demonios de red. Por ejemplo, deshabilitar el protocolo IPv6 en la interfaz de red será una buena idea si no se va a utilizar. Los riesgos de seguridad inherentes a protocolos como FTP, Telnet y programas BSD "r" (rcp, rlogin y rsh) utilizan métodos de autenticación y transferencia de datos inseguros. Deben deshabilitarse en todos los sistemas a favor de alternativas más seguras como SSH.
Registro de eventos remotos **(*Remote event logging*)**	La función **syslog** reenvía información de registro a archivos, listas de usuarios u otros hosts en su red. Se puede configurar un host seguro para que actúe como una máquina de registro central que analice los eventos (*logs*) reenviados y se tomen las medidas apropiadas. Un único agregador de registros centralizado puede capturar registros de una variedad de dispositivos y alertar a los administradores siempre que ocurran eventos significativos. El registro remoto también evita que los *hackers* cubran sus pistas reescribiendo o borrando los *logs* locales en sistemas que han sido comprometidos.
Copias de seguridad (*Backup*)	Las copias de seguridad regulares del sistema son una parte esencial de cualquier plan de seguridad del sitio. Tomar en cuenta la seguridad de almacenamiento y los procedimientos para restauración de datos, cuando sean necesarios.
Virus, gusanos y troyanos	Normalmente será una recomendación estándar instalar y mantener actualizado un software antivirus en las estaciones de trabajo, especialmente si usan Microsoft Windows. Para el caso de los sistemas Unix/Linux, especialmente en los servidores de correo electrónico y archivos, para detectar software malicioso que esté siendo enviado a los usuarios finales, y evitar su propagación en la red.
Contraseñas seguras	Aplicar políticas de seguridad adecuadas para configurar contraseñas fuertes, y que se deban actualizar periódicamente. También evitar el envío y almacenamiento de contraseñas sin cifrarse.
Vigilancia	Monitorear regularmente el estado de los sistemas, sus conexiones de red, tabla de procesos y estado general. Realizar autoevaluaciones periódicas, implementando de esta manera el monitoreo proactivo, adelantándose ante posibles problemas.
Recomendaciones generales	• Cifrar los archivos y la información crítica.Implementar políticas de seguridad que informen sobre la criticidad de la información, y que sean aceptadas y formadas por todos los usuarios. • No publicar servicios que puedan servir a los atacantes como puentes: servidores FTP con cuentas Anonymous con acceso de escritura, cuentas compartidas, etc. • Monitorear continuamente los reportes generados por las herramientas de seguridad instaladas. Poner especial atención en las actividades inusuales, que pueden ser detectadas observando gráficos de comportamiento y reportes en las herramientas de seguridad. • Mantener al equipo de administración de sistemas actualizado sobre las vulnerabilidades de seguridad que van apareciendo en la industria, acudiendo a expertos en los casos que sea necesario.

Nota: Adaptado de Nemeth, Snyder, Hein, & Whaley (2011).

2.5.4. Control de acceso a los archivos, directorios y dispositivos

Una cuestión importante con respecto a la seguridad del sistema es el control de acceso del usuario al sistema de archivos: cómo se restringe el acceso a los archivos a grupos de usuarios, y qué comandos son necesarios para administrar esto.

Con el fin de restringir los privilegios a los archivos en el sistema y crear la ilusión de un *host* virtual para cada usuario conectado, **los sistemas basados en Unix registran información sobre quién**

crea los archivos y también quién puede luego acceder a ellos. Unix no establece ninguna política sobre qué nombres deberían tener los archivos: un archivo puede tener cualquier nombre, siempre que no contenga caracteres ilegales como la barra diagonal. El contenido de un archivo se clasifica por los llamados **números mágicos**, que son códigos de 16 o 32 bits que se mantienen al inicio de un archivo y se definen en el archivo de números mágicos para el sistema. Los números mágicos le dicen al sistema a qué aplicación pertenece un tipo de archivo o cómo se debe interpretar. Esto contrasta con sistemas como Windows, donde las extensiones de archivo (por ejemplo, .EXE) se utilizan para indicar que el contenido del archivo es un programa ejecutable. En Unix, las extensiones de archivo (por ejemplo, .c) son solo discrecionales.

Cada usuario tiene un **nombre de usuario** o nombre de usuario único junto con un **ID de usuario** o **UID únicos**. El **ID de usuario es un número**, mientras que **el nombre de usuario es una cadena de texto** (los dos expresan la misma información). Un archivo pertenece al usuario A si es propiedad del usuario A. El usuario A decide si otros usuarios pueden **leer**, **escribir** o **ejecutar** el archivo configurando los bits de protección o el permiso del archivo utilizando el comando `chmod`.

Además de los identificadores de usuarios, hay **grupos de usuarios**. La idea de un grupo es que varios usuarios nombrados puedan querer leer y trabajar en un archivo, sin que otros usuarios puedan acceder a él. Cada usuario es miembro de al menos un grupo, llamado **grupo de inicio de sesión** y cada grupo tiene tanto un nombre textual como un número (ID del grupo, GID). Como se ha mencionado en los apartados anteriores, el UID y el GID de cada usuario se registran en el archivo `/etc/passwd`. La membresía de otros grupos se registra en el archivo `/etc/group`.

La siguiente salida es del comando `ls -la` ejecutado en una máquina Unix/Linux:

```
lrwxrwxrwx 1 root wheel 7 Jun 1 1993 bin -> usr/bin
-r--r--r-- 1 root bin 103512 Jun 1 1993 boot
drwxr-sr-x 2 bin staff 11264 May 11 17:00 dev
drwxr-sr-x 10 bin staff 2560 Jul 8 02:06 etc
drwxr-sr-x 8 root wheel 512 Jun 1 1993 export
drwx------ 2 root daemon 512 Sep 26 1993 home
-rwxr-xr-x 1 root wheel 249079 Jun 1 1993 kadb
lrwxrwxrwx 1 root wheel 7 Jun 1 1993 lib -> usr/lib
drwxr-xr-x 2 root wheel 8192 Jun 1 1993 lost+found
drwxr-sr-x 2 bin staff 512 Jul 23 1992 mnt
dr-xr-xr-x 1 root wheel 512 May 11 17:00 net
drwxr-sr-x 2 root wheel 512 Jun 1 1993 pcfs
drwxr-sr-x 2 bin staff 512 Jun 1 1993 sbin
lrwxrwxrwx 1 root wheel 13 Jun 1 1993 sys->kvm/sys
```

La primera columna es una representación textual de los **bits de protección** para cada archivo. La columna dos es el número de enlaces duros al archivo, para archivos normales, o el número de objetos contenidos en un subdirectorio. Las columnas tercera y cuarta son el **nombre de usuario** y el **nombre del grupo** y el resto muestra el tamaño del archivo en bytes y la fecha de creación. Tomar en cuenta que los directorios `/bin` y `/sys` suelen ser enlaces simbólicos a otros directorios.

Hay **dieciséis bits de protección para un archivo Unix**, pero solo doce de ellos pueden ser cambiados por los usuarios. Estos doce se dividen en cuatro grupos de tres. Cada número de tres bits corresponde a un número octal.

Los cuatro bits invisibles iniciales proporcionan información sobre el tipo de archivo: es el archivo un archivo simple, un directorio o un enlace, etc. En la salida del comando `ls`, esto se representa con un solo carácter: `-`, `d`, o `l`.

Los siguientes tres bits establecen los llamados `s-bits` y `t-bits` que se explican a continuación. Los tres grupos restantes de tres bits indican si un archivo se puede leer (`r`), escribir (`w`) o ejecutar (`x`) por (i) el usuario que los creó, (ii) los otros usuarios que están en el grupo en el que el archivo está marcado, y (iii) cualquier usuario en absoluto.

Por ejemplo, el permiso:

```
Tipo  Propietario  Grupo Cualquiera
 d       rwx        r-x     ---
```

Dice que el archivo es un directorio (`d`), que puede ser leído (`r`), escrito (`w`) y ejecutado (`x`) por el propietario (`rwx`), puede ser leído y ejecutado (pero no escrito) por otros en su grupo (`r-x`), y el "resto del mundo" no puede ejecutar ninguna operación sobre el archivo (`---`).

Nota sobre directorios: Es imposible cambiarse de directorio con el comando `cd` a un directorio a menos que se establezca el bit `x`. Es decir, los directorios deben ser "ejecutables" para que sean accesibles.

Estos son algunos ejemplos de la relación entre binario, octal y la representación textual de los modos de acceso a los archivos:

```
Binary   Octal      Text
001        1        --x
010        2        -w-
100        4        r--
110        6        rw-
101        5        r-x
-         644     rw-r--r—
```

El comando `chmod` se utiliza para cambiar el permiso o el modo de un archivo. Solo el propietario del archivo o el superusuario pueden cambiar el permiso. Aquí algunos ejemplos de su uso.

```
# Hacer el archivo myfile ejecutable por todo el Mundo.
chmod a+w myfile

# Añadir la bandera de "ejecución" al directorio "mydir" y al usuario (propietario)
del directorio
chmod u+x mydir/
```

```
# Abrir todos los archivos para todo el mundo
chmod 755 *


# Activar el s-bit en el grupo de my-dir
chmod g+s mydir/

# Descender recursivamente dentro de un directorio y abriendo todos los archivos
chmod -R a+r dir
```

Finalmente, se menciona lo que se considera la evolución en el estilo de control de acceso a los archivos y directorios tradicional de Unix, como son las **listas de control de acceso** (*Access Control List*, ACL). Anteriormente, la única forma de otorgar acceso a un archivo, para una lista conocida de usuarios, era hacer un grupo de esos usuarios y utilizar el atributo de grupo del archivo. Con las ACL esto ya no es necesario. Una ACL se especifica diciendo qué permisos se quiere otorgar y a qué usuario o grupo de usuarios deberían aplicarse los permisos. Una ACL puede otorgar acceso o denegar el acceso a un usuario específico. Debido a la cantidad de tiempo requerida para verificar todos los permisos en una ACL, las ACL ralentizan las operaciones de búsqueda de archivos.

Los sistemas operativos basados en GNU/Linux y BSD realmente no tienen ACLs. Si se otorga acceso a un archivo que se comparte en la red a una máquina que no admite ACLs, estos se ignorarán. Esto limita su utilidad en la mayoría de los casos.

Tema 3. Sistemas heterogéneos y administración centralizada

La combinación de sistemas operativos radicalmente diferentes en un entorno de red es un desafío tanto para los usuarios como para los administradores. Cada sistema operativo presta un buen servicio a una función específica, y si se permite que los usuarios pasen de un sistema operativo a otro con acceso a sus datos personales, debe haber un equilibrio entre la **conveniencia de la disponibilidad** y la **precaución de la diferenciación**. Los usuarios deberán tener claro dónde están y qué sistema están utilizando, para evitar errores desafortunados. Combinar diferentes sistemas tipo Unix/Linux es un desafío suficiente, pero agregar hosts Windows o Apple a una red basada principalmente en Unix/Linux, o viceversa, requiere una planificación cuidadosa. Se han de tomar en cuenta aspectos como las diferencias de los sistemas de archivos, los estándares de identificación de usuarios y grupos, entre otros, que en la práctica requerirán alguna plataforma en la que se logre centralizar la administración y permita tener visibilidad del conjunto de forma unificada, tanto para las configuraciones como para los accesos a los recursos de forma unificada por parte de los usuarios.

La integración de tecnologías radicalmente diferentes no vale la pena a menos que exista una necesidad particular. Siempre es posible mover datos entre dos *hosts* utilizando el protocolo FTP, por ejemplo, universalmente compatible. Pero habría que hacer un esfuerzo mayor de integración si se necesita compartición de archivos en tiempo real o compatibilidad de *software*. En este capítulo se estudian los diferentes estándares y plataformas que existen al día de hoy en la industria para lograr soportar los desafíos que implica la administración centralizada de sistemas heterogéneos.

3.1. Integración de múltiples sistemas operativos

A continuación, algunos aspectos a tomar en cuenta ente el problema de la integración de múltiples sistemas operativos en una infraestructura tecnológica (Burgess, 2004):

- **Compatibilidad de nombres - Los diferentes sistemas operativos utilizan esquemas de nomenclatura bastante diferentes para los objetos.** Hasta finales de la década de 1990, los nombres de Unix no podían representarse en MSDOS a menos que no fueran más de ocho caracteres. Algunos sistemas operativos no permiten espacios en los nombres de archivo. Algunos asignan y reservan significados especiales para los caracteres. Los nombres pueden desempeñar un papel fundamental en la forma en que se elige integrar recursos dentro de un sistema.

- **Compartición de Sistema de archivos - La compartición de sistemas de archivos entre diferentes sistemas operativos puede ser útil en una variedad de circunstancias.** Los servidores de archivos, que alojan y comparten los archivos de los usuarios, deben ser máquinas de alta capacidad, estables. Las estaciones de trabajo para usuarios finales, por otro lado, se eligen por razones muy diferentes. Los sistemas basados en Mac (Apple) siempre han sido los preferidos para aplicaciones multimedia y de diseño gráfico. Los sistemas operativos Windows son baratos y tienen una base de *software* amplia y exitosa, especialmente en la rama de la ofimática. La mayoría de las soluciones al problema de intercambio de archivos están basadas en *software*. Existe *software* cliente-servidor disponible para implementar protocolos de compartición de archivos en red como NFS (*Network File System*). El *software* **Samba** es un paquete de *software* gratuito que implementa la semántica de archivos Unix en términos de los protocolos Windows **SMB** (*Server Message Block*). Samba proporciona servicios de impresión y archivos seguros, estables y rápidos para todos los clientes que utilizan el protocolo SMB / **CIFS**, como todas las versiones de DOS y Microsoft Windows, OS/2, Linux y otros. Es un componente importante para integrar servidores y escritorios Unix/Linux en entornos de *Microsoft Active Directory*. Puede funcionar como un controlador de dominio o como un miembro de dominio regular.

- **Identificadores de usuarios y contraseñas - Si se desea implementar el uso compartido en sistemas operativos tan diferentes como Unix/Linux y Windows, se deben tener nombres de usuario comunes en ambos sistemas.** La autenticación de usuario en plataformas heterogéneas generalmente se basa en el entendimiento de que el nombre de usuario se puede asignar a través de los sistemas operativos. Los identificadores de usuario (UID) numéricos de Unix/Linux y los IDs de seguridad de Windows no pueden relacionarse fácilmente, sólo se puede establecer dicha relación a través del nombre de usuario. Para lograr la integración, entonces, se deben estandarizar los nombres de usuario para que cumplan las reglas de ambos sistemas. Cada usuario debe tener el mismo y único nombre en cada host. Esto se hace más viable y administrable implementando servicios de autenticación centralizados como los empleados en los servicios de directorio con protocolos como LDAP.

- **Autenticación de usuarios** - Hacer que las contraseñas funcionen en diferentes sistemas operativos es a menudo un problema pernicioso en un esquema para una integración

completa. **Los mecanismos de contraseña para Unix/Linux y Windows son completamente diferentes y básicamente incompatibles**.

3.2. Servicios de directorio

Una forma de vincular a una organización es a través de un modelo de información estructurado: una base de datos de su personal, activos y servicios. El estándar X.500 define un **servicio de directorio** como una colección de sistemas abiertos que cooperan para mantener una base de datos lógica de información sobre un conjunto de objetos en el mundo real. Un servicio de directorio es un servicio de nombres generalizado.

Los servicios de directorio no deben confundirse con los directorios en los sistemas de archivos, aunque tienen muchas similitudes estructurales (Burgess, 2004):

- Los directorios están organizados de manera estructurada, a menudo jerárquicamente (estructura de árbol), empleando un modelo orientado a objetos.
- Los servicios de directorio emplean un esquema común de lo que puede y debe almacenarse sobre un objeto en particular, para promover la interoperabilidad.
- Se proporciona un control de acceso granular a la información, que permite el acceso por registro.
- El acceso está optimizado para la búsqueda, no para la actualización transaccional de información. Un directorio no es una base de datos de lectura y escritura, en el sentido normal, sino una base de datos utilizada para transacciones de solo lectura. Se mantiene y actualiza mediante un proceso administrativo separado en lugar de un uso regular.

3.2.1. El estándar X.500 y LDAP

En 1988, se definió el estándar ISO 9594, creando el estándar para directorios llamado X.500. Las recomendaciones X.500 - X.521 para directorios de la red de comunicaciones de datos surgieron en 1990, aunque todavía se conoce como X.500. X.500 se define en términos de otro estándar, el ASN.1 (*Abstract Syntax Notation*), que se utiliza para definir protocolos formateados en varios sistemas, incluidos SNMP e Internet Explorer. X.500 especifica un Protocolo de Acceso a Directorios (*Directory Access Protocol DAP*) para direccionar un directorio jerárquico, con una potente funcionalidad de búsqueda. Dado que DAP es un protocolo de capa de aplicación, requiere toda la pila de protocolos del modelo OSI para poder funcionar. Esto requería más recursos de los que estaban disponibles en muchos entornos pequeños, por lo que era deseable una alternativa ligera que pudiera ejecutarse solo con la infraestructura TCP/IP normal. Así se definió LDAP, implementado en una serie de proyectos de normas. **La versión actual es LDAPv3, definida en RFC 2251–2256. LDAP es un estándar abierto de Internet y está diseñado para ser interoperable entre varios sistemas operativos y computadoras.** Emplea mejor seguridad que los estándares abiertos anteriores (como NIS). Por lo tanto, reemplaza gradualmente o se integra con sistemas específicos del proveedor, incluido el *Novell Directory Service* (NDS) y el *Microsoft Active Directory* (AD).

Las entradas en un directorio son pares de *nombre-valor* llamados **atributos del directorio**. Puede haber múltiples valores asociados con un nombre, por lo tanto, se dice que los atributos son de **valor único** o de **valores múltiples**. Cada atributo tiene una sintaxis, o formato, que define un conjunto de sub-atributos que describen el tipo de información que se puede almacenar en el esquema de dirección. Una definición de atributo incluye reglas de coincidencia que rigen cómo se deben mapear las mismas. Es posible requerir mapeo total o de subcadenas, así como reglas que especifiquen el orden de coincidencia de atributos en una búsqueda. Algunos atributos son obligatorios, otros son opcionales.

Los objetos del mundo real generalmente se pueden clasificar en categorías que se ajustan a una jerarquía de objetos. Se pueden definir subclases de una clase, que heredan todos los atributos obligatorios y opcionales de su clase principal. La clase *top* es la raíz de la jerarquía de clases de objetos. Todas las demás clases se derivan de ésta, ya sea directamente o mediante herencia. Por lo tanto, cada entrada de datos tiene al menos una clase de objeto (*objectClass*). Existen tres tipos de clase de objeto (Burgess, 2004):

- **Resumen (*Abstract*):** forman los niveles superiores de la jerarquía de clases de objetos; sus entradas solo se pueden completar si las hereda al menos una clase de objeto estructural. Están destinados a ser "heredados de" en lugar de usarse directamente, pero contienen algunos campos de datos. Por ejemplo: "top", "País", "Dispositivo", "Persona-Organizacional", "Objeto de seguridad", etc.

- **Estructural:** representan la "carne" de una clase de objeto, utilizada para hacer entradas reales. Ejemplos de estos son "Persona" y "Organización". La clase de objeto a la que pertenece una entrada se declara en un atributo *objectClass*, por ejemplo: "Computadora" y "Configuración".

- **Auxiliar:** sirve para definir atributos de casos especiales que se pueden agregar a entradas específicas. Los atributos se pueden introducir, como requisito, a solo un subconjunto de entradas para proporcionar sugerencias adicionales. Por ejemplo, tanto una **persona** como una **organización** pueden tener una página web o un número de teléfono, pero no es necesario.

Una clase de objeto especial es el **alias**, que no contiene datos, pero simplemente apunta a otra clase. Las clases de objetos importantes se definen en RFC 2256. Todas las entradas en un directorio X.500 están ordenadas jerárquicamente, formando un árbol de información de directorio (*Directory Information Tree*, DIT). Por lo tanto, un directorio es similar a un sistema de archivos en su estructura. Cada entrada se identifica por su "Nombre Distinguido" (*Distinguished Name*, DN), que es una designación jerárquica basada en la herencia. Esta es una entrada de "coordenadas" dentro del árbol. Se compone de unir un Nombre distinguido relativo (*Relative Distinguished Name*, RDN) con todos los de sus padres, hasta la clase superior o raíz. Un RDN consiste en una asignación de un nombre de atributo a un valor. Por ejemplo:

```
cn=’’Wilfredo Torres’
```

X.500 originalmente siguió un esquema de nombres basado en regiones geográficas, pero desde entonces se ha movido hacia un esquema de nombres basado en la geografía virtual del Servicio de Nombres de Dominio (DNS). Para asignar un nombre DNS a un nombre distinguido (DN), se usa el atributo "dc" (*domain component*). Por ejemplo, para el nombre de dominio del *Oslo University College* (*hio.no*) se usa la siguiente notación:

```
dc=hio,dc=no
```

Los servicios de directorio jerárquico son adecuados para ser distribuidos o delegados a varios *hosts*. Un árbol de información de directorio se divide en regiones más pequeñas, cada una de las cuales es un subárbol conectado, que no se superpone con otras particiones de subárbol (Figura 7). Esto permite que varias autoridades cooperantes dentro de una organización mantengan los datos de manera más racional y permite, al menos en principio, la formación de un directorio global, análogo al DNS. La disponibilidad y la redundancia se pueden aumentar ejecutando servicios de replicación, brindando una funcionalidad de respaldo o conmutación por error. Un servidor maestro dentro de cada partición mantiene registros maestros y estos se replican en sistemas esclavos. Algunas implementaciones comerciales (por ejemplo, NDS) permiten servidores multimaestro.

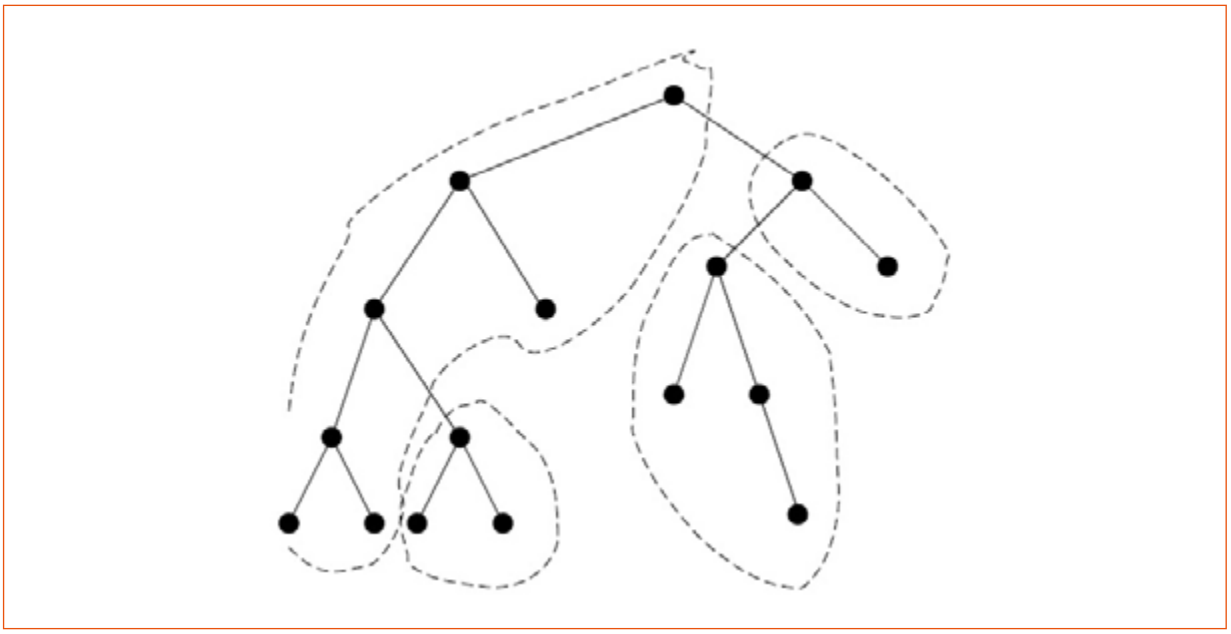

Figura 7. La partición de un directorio distribuido. Cada área punteada es manejada por un servidor separado. Recuperado de Burges (2004).

El proceso o *software* que consulta directorios generalmente está integrado en el *software* de la aplicación. Este se denomina un Agente de Usuario de Directorio (*Directory User Agent*, DUA), y es el encargado de realizar consultas a un servicio de directorio en nombre de los usuarios. Por ejemplo, la biblioteca de resolución de nombres en Unix admite la llamada al sistema `gethostbyname`, que es una función del sistema que delega una consulta en el directorio de nombres de host. El "conmutador de servidor de nombres" se utiliza en Unix para seleccionar una política para consultar una variedad de servicios de directorio de la competencia, al igual que los Módulos de Autenticación Conectables (*Pluggable Authentication Modules*, PAM).

La fortaleza de LDAP es su versatilidad e interoperabilidad con todos los sistemas operativos. Sus desventajas son su estructura sintáctica (algo arbitraria y fea), y su vulnerabilidad a la pérdida de conectividad de red.

Los datos para un directorio simple se ingresan en la forma del formato de archivo común. El LDIF (*LDAP Data Interchange Format*) se utiliza para definir y almacenar datos de origen. Este formato de datos es extremadamente sensible sobre espacios y líneas adicionales, y ofrece poca ayuda para la depuración. Algún día probablemente se reescribirá en XML; hasta entonces, se requiere un cierto cuidado.

El siguiente es un ejemplo de una entrada típica de `/etc/passwd` (simplificado) expresada como un registro LDAP en formato LDIF:

```
uid: ghopper
cn: Grace Hopper
userPassword: {crypt}$1$pZaGA2RL$MPDJoc0afuhHY6yk8HQFp0
loginShell: /bin/bash
uidNumber: 1202
gidNumber: 1202
homeDirectory: /home/ghopper
```

El formato LDIF es utilizado por la mayoría de las herramientas relacionadas con LDAP y las implementaciones de servidores. El hecho de que los datos LDAP se puedan convertir fácilmente desde y hace un texto plano es parte de la razón de su éxito.

Las entradas se organizan en una jerarquía mediante el uso de los *dn* (*Distinguish Name*) que forman una especie de ruta de búsqueda, en una estructura de árbol, como se ha explicado anteriormente. Por ejemplo, el *dn* para el usuario anterior podría ser:

```
dn: uid=ghopper,ou=People,dc=navy,dc=mil
```

Como en el sistema DNS, el "bit más significativo" va a la derecha. Aquí, el nombre DNS `navy.mil` se ha utilizado para estructurar los niveles superiores de la jerarquía LDAP. Se ha desglosado en dos componentes de dominio (*dc*), "navy" y "mil", pero ésta es solo una de varias convenciones comunes. Cada entrada tiene exactamente un *dn*. Por lo tanto, la jerarquía de entrada se parece a un árbol de ramificación simple sin bucles. Sin embargo, existen disposiciones para enlaces simbólicos entre entradas y para referencias a otros servidores.

Los directorios LDAP se definen usando un esquema de clases que pueden heredar otras clases. Cada clase tiene sus propios atributos. Uno de los desafíos de usar LDAP es descubrir qué clases tienen qué atributos y viceversa. Resolver un problema de directorio se trata principalmente de hacer que estas relaciones funcionen. Algunas de las clases de esquema están definidas por X.500, como `cn` (*Common Name*), `description` y `postalAddress`.

El nivel superior del árbol de clase de objeto es la clase denominada *top*, que especifica simplemente que una entrada debe tener un atributo **objectClass**.

En el siguiente ejemplo se muestra una base de datos simple de personas en LDAP.

Suponer el archivo **ejemplo.ldif** con el siguiente contenido:

```
dn:dc=iu,dc=hio,dc=no
objectclass:organization
o:Oslo University College

dn: cn=Mark Burgess,dc=iu,dc=hio,dc=no
objectClass: person
cn: Mark Burgess
cn: Mark Sparc
sn: Burgess

dn: cn=Sigmund Straumsnes,dc=iu,dc=hio,dc=no
objectClass: person
cn: Sigmund Straumsnes
cn: Ziggy
sn: Straumsnes

dn: cn=Frode Sandnes,dc=iu,dc=hio,dc=no
objectClass: person
cn: Frode Sandnes
cn: Frodo
sn: Sandnes
```

Para añadir entradas a este archivo desde la línea de comandos, en un sistema Unix/Linux con OpenLDAP:

```
system$ ldapadd -x -D "cn=Manager,dc=iu,dc=hio,dc=no" -W -f example2.ldif
Enter LDAP Password:
adding new entry "dc=iu,dc=hio,dc=no"
adding new entry "cn=Mark Burgess,dc=iu,dc=hio,dc=no"
adding new entry "cn=Sigmund Straumsnes,dc=iu,dc=hio,dc=no"
adding new entry "cn=Frode Sandnes,dc=iu,dc=hio,dc=no"
```

Para chequear si la entrada ha sido añadida correctamente, se imprimen todos los registros de la siguiente manera:

```
system$ ldapsearch -x -b 'dc=iu,dc=hio,dc=no' '(objectclass=*)'
```

La salida al comando anterior debería lucir como la siguiente:

```
# extended LDIF
#
# LDAPv3
```

```
# filter: (objectclass=*)
# requesting: ALL
#

# iu.hio.no
dn: dc=iu,dc=hio,dc=no
objectClass: organization
o: Oslo University College

# Mark Burgess, iu.hio.no
dn: cn=Mark Burgess,dc=iu,dc=hio,dc=no
objectClass: person
cn: Mark Burgess
cn: Mark Sparc
sn: Burgess

# Sigmund Straumsnes, iu.hio.no
dn: cn=Sigmund Straumsnes,dc=iu,dc=hio,dc=no
objectClass: person
cn: Sigmund Straumsnes
cn: Ziggy
sn: Straumsnes

# Frode Sandnes, iu.hio.no
dn: cn=Frode Sandnes,dc=iu,dc=hio,dc=no
objectClass: person
cn: Frode Sandnes
cn: Frodo
sn: Sandnes

# search result
search: 2
result: 0 Success

# numResponses: 5
# numEntries: 4
```

El ejemplo anterior es una lista plana, formada a partir del esquema de clase principal. ¿Cómo agregar clases y subárboles adicionales? Para heredar atributos de esquema adicionales, se debe incluir el esquema en `slapd.conf`, después de la línea predeterminada del esquema 'core':

```
include /usr/local/etc/openldap/schema/core.schema
include /usr/local/etc/openldap/schema/cosine.schema
include /usr/local/etc/openldap/schema/inetorgperson.schema
include /usr/local/etc/openldap/schema/nis.schema
```

Ahora se puede agregar entradas en "OrganizationalPerson", por ejemplo:

```
dn:dc=iu,dc=hio,dc=no
objectclass:organization
o:Oslo University College

dn: cn=Mark Burgess,dc=iu,dc=hio,dc=no
objectClass: person
objectClass: organizationalPerson
cn: Mark Burgess
cn: Mark Sparc
sn: Burgess
registeredAddress: Cort Adelers Gate 30
telephoneNumber: +47 22453272

dn: cn=Sigmund Straumsnes,dc=iu,dc=hio,dc=no
objectClass: person
cn: Sigmund Straumsnes
cn: Ziggy
sn: Straumsnes

dn: cn=Frode Sandnes,dc=iu,dc=hio,dc=no
objectClass: person
cn: Frode Sandnes
cn: Frodo
sn: Sandnes
```

Tabla 9
Abreviaturas básicas manejadas en LDAP

Abreviatura	Significado	Descripción o función
dN	*Distinguished Name*	Clave primaria
CN	*Common Name*	Típicamente un identificador
RDN	*Relative Distinguished Name*	Clave primaria de un sub-objeto
DIT	*Directory Information Tree*	Jerarquía LDAP
DSA	*Directory System Agent*	Nombre X.500 para un servidor LDAP
DSE	*DSA-Specific Entry*	Nodo raíz de un contexto DIT
DC	*Domain component*	Elemento "punto" sin mayúsculas en el nombre DNS.
O	*Organization*	Nombre de dominio principal de una organización (registro de alto nivel)
OU	*Organizational Unit*	Una subdivisión lógica de la organización, como un departamento. Ejemplo: "*Marketing*".

Nota. Adaptado de Burgess (2004).

Las directivas del archivo de configuración `objectClass` y `attributeTypes` se pueden usar para definir reglas de esquema en las entradas del directorio. Es habitual crear un archivo que contenga definiciones de elementos de esquema personalizados:

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

No se puede mezclar un esquema de clase diferente en los registros. Por ejemplo, no puede registrar información para el esquema 'persona' en la misma estrofa que para 'posixAccount'. Por lo tanto, lo siguiente sería incorrecto:

```
dn: cn=Mark Burgess,dc=iu,dc=hio,dc=no
objectClass: person
objectClass: account
objectClass: posixAccount
cn: Mark Burgess
cn: Mark Sparc
sn: Burgess
uid: mark
userPassword: cryptX5/DBrWPOQQaI
gecos: Mark Burgess (staff)
loginShell: /bin/tcsh
uidNumber: 10
gidNumber: 10
homeDirectory: /site/host/mark
```

Porque cuenta y persona son mutuamente excluyentes. Sin embargo, al extender los DN para dividir el árbol en dos subtipos, se puede terminar con lo siguiente:

```
dn:dc=iu,dc=hio,dc=no
objectclass: top
objectclass: organization
o:Oslo University College
description: Faculty of Engineering
streetAddress: Cort Adelers Gate 30
postalAddress: 0254 Oslo Norway

dn: cn=Mark Burgess,dc=iu,dc=hio,dc=no
objectclass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
cn: Mark Burgess
cn: Mark Sparc
sn: Burgess
uid: mark
```

```
registeredAddress: Cort Adelers Gate 30
telephoneNumber: +47 22453272

dn: cn=Sigmund Straumsnes,dc=iu,dc=hio,dc=no
objectclass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
cn: Sigmund Straumsnes
cn: Ziggy
sn: Straumsnes
uid: sigmunds
registeredAddress: Cort Adelers Gate 30
telephoneNumber: +47 22453273

dn: cn=Frode Sandnes,dc=iu,dc=hio,dc=no
objectclass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
cn: Frode Sandnes
cn: Frodo
sn: Sandnes
uid: frodes
registeredAddress: Cort Adelers Gate 30
telephoneNumber: +47 22453274

dn: uid=mark,dc=iu,dc=hio,dc=no
cn: Mark Burgess
cn: Mark Sparc
objectClass: account
objectClass: posixAccount
uid: mark
userPassword: cryptX5/DBrWPOQQaI
gecos: Mark Burgess (staff)
loginShell: /bin/tcsh
uidNumber: 10
gidNumber: 10
homeDirectory: /site/host/mark

dn: uid=sigmunds,dc=iu,dc=hio,dc=no
cn: Sigmund Straumsnes
cn: Ziggy
objectClass: account
objectClass: posixAccount
```

```
uid: mark
userPassword: cryptX5/sdWPOQQaI
gecos: Sigmund Straumsnes (staff)
loginShell: /bin/zsh
uidNumber: 11
gidNumber: 11
homeDirectory: /site/host/sigmunds

dn: uid=frodes,dc=iu,dc=hio,dc=no
cn: Frode Sandnes
cn: Frodo
objectClass: account
objectClass: posixAccount
uid: frodes
userPassword: cryptX5/DBr111QaI
gecos: Frode Sandnes (staff)
loginShell: /bin/bash
uidNumber: 12
gidNumber: 12
homeDirectory: /site/host/frodes
```

Ahora se pueden buscar clases de objetos individuales y atributos:

```
ldapsearch -x -b 'dc=iu,dc=hio,dc=no' '(objectclass=account)'
ldapsearch -x -b 'dc=iu,dc=hio,dc=no' '(objectclass=person)'
ldapsearch -x -b 'dc=iu,dc=hio,dc=no' '(cn=Mark*)'
ldapsearch -x -b 'dc=iu,dc=hio,dc=no' '(uid=fr*)'
ldapsearch -x -b 'dc=iu,dc=hio,dc=no' '(loginShell=/bin/zsh)'
```

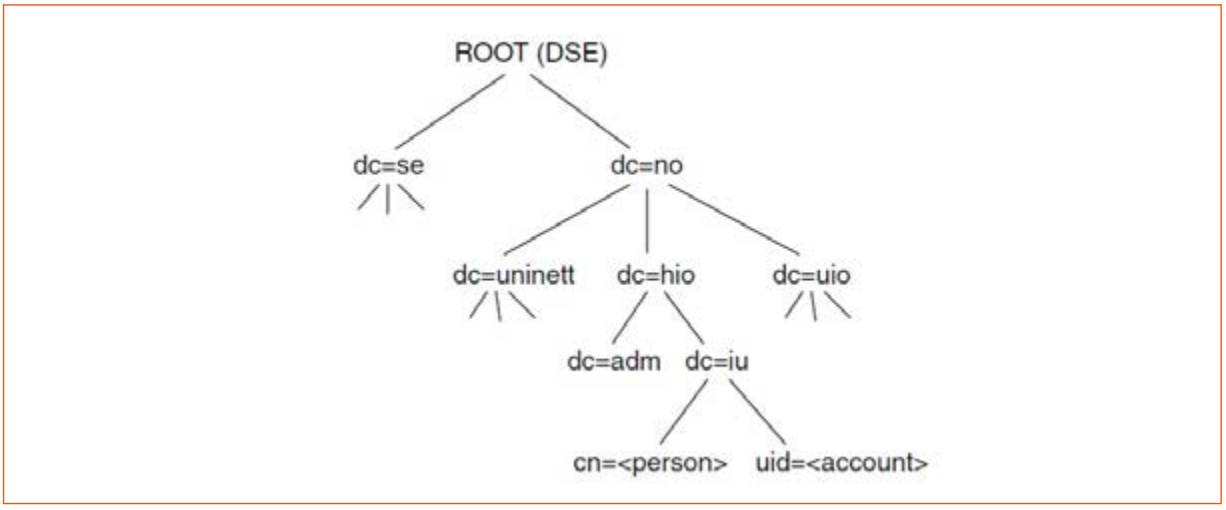

Figura 8. Ejemplo de modelo de jerarquía de datos para el DIT (*Directory Information Tree*) de iu.hio.no. Recuperado de Burgess (2004).

3.2.2. Open LDAP

OpenLDAP es la implementación de LDAP de referencia para sistemas tipo Unix. Se puede acceder a la información del directorio a través de una variedad de agentes, y se puede agregar a la lista de servidores de nombres de Unix a través del archivo `nsswitch.conf` y Módulos PAM.

El archivo de configuración `slapd.conf` determina el espacio de nombre local, así como la protección de identidad y contraseña del administrador de la base de datos. Aquí se puede configurar mucho, relacionado con la seguridad y las extensiones de esquema. **Para empezar, es importante establecer una contraseña para el administrador de la base de datos.** Esta contraseña debe cifrarse manualmente y pegarse en el archivo de configuración.

El comando `slappasswd` mezcla las contraseñas en cadenas ASCII, para que puedan agregarse al archivo `slapd.conf`.

```
/usr/local/sbin/slappasswd
New Password:
Repeat New Password:
SSHAkDPIIA9KR5LVQthcv+zJmzpC+GVYQ4Jj
```

Un ejemplo de un archivo de configuración puede lucir como el siguiente:

```
# See slapd.conf(5) for details on configuration options.
# This file should NOT be world readable.
#
include /usr/local/etc/openldap/schema/core.schema
pidfile /usr/local/var/slapd.pid
argsfile /usr/local/var/slapd.args
database bdb
suffix "dc=iu,dc=hio,dc=no"
rootdn "cn=Manager,dc=iu,dc=hio,dc=no"
# Cleartext passwords, especially for the rootdn, should
# be avoid. See slappasswd(8) and slapd.conf(5) for details.
# Use of strong authentication encouraged.
#rootpw secret
rootpw SSHAkDPIIA9KR5LVQthcv+zJmzpC+GVYQ4Jj
# The database directory MUST exist prior to running slapd AND
# should only be accessible by the slapd/tools. Mode 700 recommended.
directory /usr/local/var/openldap-data
# Indices to maintain
index objectClass eq
```

Nótese que la contraseña se verifica al agregar datos a la base de datos. Los errores de credenciales de contraseña se informan si la contraseña es incorrecta o si los componentes de DC en el sufijo son

incorrectos. Todo en la especificación tiene que ser correcto. Piense en estos como nombre de usuario y contraseña. El servidor se puede iniciar de la siguiente manera:

```
system# /usr/local/libexec/slapd -h "ldap://0.0.0.0"
```

3.2.3. Microsoft Active Directory (AD)

Las primeras versiones de Microsoft Windows estaban limitadas por un modelo de infraestructura de *host* plano que hacía difícil organizar y administrar los hosts de Windows racionalmente. *Active Directory* (AD) es el servicio de directorio introducido e integrado en Windows 2000. Reemplaza el modelo de dominio utilizado en NT4 y se basa en conceptos de X.500. Es compatible con LDAP, y utiliza DNS como sistema de nombres y Kerberos para la autenticación de usuarios y recursos. En el *software* de red original de Windows, los nombres se basaban en *software* propietario como WINS. Windows ha adoptado cada vez más estándares abiertos como DNS, y ha elegido el modelo de nombres DNS para la integración LDAP.

El área de partición LDAP más pequeña en AD se denomina **dominio** para proporcionar un punto de partida para los usuarios de NT4. El esquema en AD difiere ligeramente del modelo de información X.500. Las clases auxiliares no existen como clases independientes, sino que se incorporan a clases estructurales. Como resultado, las clases auxiliares no se pueden buscar, y no se pueden agregar de forma dinámica o independiente. Otras diferencias incluyen el hecho de que todos los RDN deben tener un valor único y que las reglas de coincidencia no se publican para inspección por parte de los agentes; las reglas de búsqueda están ocultas.

AD recopila usuarios, grupos, computadoras y políticas de sistemas operativos bajo un único paraguas, centralizando y simplificando la administración del sistema. Es una de las razones principales por las que Microsoft Windows se ha afianzado en muchas empresas. Los servidores que ejecutan AD se denominan **Controladores de Dominio**. Un controlador de dominio AD autentica y autoriza a todos los usuarios y computadoras en una red de tipo de dominio de Windows, **asignando** y **aplicando políticas de seguridad** para todas las computadoras e instalando o actualizando *software*. Por ejemplo, cuando un usuario inicia sesión en una computadora que forma parte de un dominio de Windows, AD verifica la contraseña enviada y determina si el usuario es un administrador del sistema u otro tipo de usuario, con otros privilegios (Pediapress, 2012).

La estructura de un *Active Directory* es una disposición jerárquica de información sobre objetos. Los objetos se dividen en dos grandes categorías:

- **Recursos:** por ejemplo, impresoras.
- **Actores de seguridad:** cuentas y grupos de usuarios, o computadoras.

Los actores de seguridad tienen asignados identificadores de seguridad únicos (*Security ID*, SID). Cada objeto representa una entidad única, ya sea un usuario, una computadora, una impresora o un grupo, y sus atributos. Ciertos objetos pueden contener otros objetos. Un objeto se identifica de forma exclusiva por su nombre y tiene un conjunto de atributos (las características y la información que representa el objeto) definidos por un **esquema**, que también determina los tipos de objetos que se

pueden almacenar en AD. El **objeto esquema** permite a los administradores ampliar o modificar el esquema cuando sea necesario. Sin embargo, debido a que cada objeto esquema es parte integral de la definición de objetos de AD, desactivar o cambiar estos objetos básicamente puede cambiar o interrumpir una implementación del AD. Los cambios de esquema se propagan automáticamente por todo el sistema. Una vez creado, un objeto solo se puede desactivar, no eliminar. Cambiar el esquema generalmente requiere planificación (Pediapress, 2012).

En la arquitectura de *Microsoft Active Directory* se resaltan los siguientes componentes y conceptos (Pediapress, 2012):

- **Sitio (*Site*):** un objeto de sitio en *Active Directory* representa una ubicación geográfica que aloja redes.

- **Bosques, árboles y dominios (*Forests, trees* y *domains*):** el marco de *Active Directory* que contiene los objetos se puede ver en varios niveles. **El bosque**, **el árbol** y **el dominio** son las divisiones lógicas en una red de AD. En una implementación, los **objetos se agrupan en dominios**. Los objetos para un solo dominio se almacenan en una única base de datos (que se puede replicar). Los dominios se identifican por su estructura de nombres DNS, el espacio de nombres. **Un árbol** es una colección de uno o más dominios y árboles de dominio en un espacio de nombres contiguo, vinculados en una jerarquía de confianza transitiva. En el tope de la estructura se encuentra **el bosque**. Un bosque es una colección de árboles que comparten un catálogo global común, un esquema de directorio, una estructura lógica y una configuración de directorio. El bosque representa el límite de seguridad dentro del cual los usuarios, computadoras, grupos y otros objetos son accesibles.

- **Unidades organizacionales (OU):** los objetos contenidos dentro de un dominio se pueden agrupar en unidades organizacionales (*Organizational Units*, OU). Las OU pueden proporcionar jerarquía a un dominio, facilitar su administración y parecerse a la estructura de la organización en términos administrativos o geográficos. Las OU pueden contener otras OU: **los dominios son contenedores en este sentido**. Microsoft recomienda usar OUs en lugar de dominios para estructurar y simplificar la implementación de políticas y administración. La unidad organizativa es el nivel recomendado para aplicar **políticas de grupo** (*Group Policy*), que son objetos de AD denominados formalmente **Objetos de Política de Grupo** (*Group Policy Objects*, GPO), aunque las políticas también se pueden aplicar a dominios o sitios. Las OU son una abstracción para el administrador y no funcionan como contenedores. **El dominio subyacente es el verdadero contenedor.** No es posible, por ejemplo, crear cuentas de usuario con un nombre de usuario idéntico (*sAMAccountName*) en OUs separadas, así como *fred.staff-ou.domain* y *fred.student-ou.domain*, donde *staff- ou* y *student-ou* son las OUs, y *fred* sería el nombre de usuario (repetido). Esto es así porque *sAMAccountName*, un atributo del objeto de usuarios, debe ser único dentro del dominio entero, incluyendo todas las OUs. Sin embargo, dos usuarios en unidades organizativas diferentes pueden tener el mismo nombre común (*Common Name*, CN), el primer componente del nombre distinguido (*Distingueshed Name*, DN) del usuario. Por lo tanto, desde el punto de vista del DN, las unidades organizativas funcionan como contenedores.

La división de la infraestructura de información de una organización en una jerarquía de uno o más dominios y OUs de nivel superior es una decisión clave. Los modelos comunes son **por unidad de negocio**, **por ubicación geográfica**, **por servicio de IT** o por tipo de objeto e híbridos de estos. Las unidades organizacionales deben estructurarse principalmente para **facilitar la delegación administrativa** y, en segundo lugar, para **facilitar la aplicación de la política de grupo (GPO)**. Aunque las OUs forman un límite administrativo, **el único límite de seguridad verdadero es el bosque mismo** y el administrador de cualquier dominio debe ser acreditado en todos los dominios del bosque.

Físicamente, la información del (AD) se almacena en uno o más controladores de dominio (DC) interconectados entre sí. Cada DC tiene una copia de la información del AD. Los servidores que se unen al AD que no son controladores de dominio se denominan "Servidores Miembros".

La base de datos del AD está organizada en particiones, cada una con tipos de objetos específicos y siguiendo un patrón de replicación específico. AD sincroniza los cambios mediante la **replicación multimaestro**. Microsoft a menudo se refiere a estas particiones como **contextos de nombres**. La partición *Esquema* contiene la definición de clases de objetos y atributos dentro del bosque. La partición *Configuración* contiene información sobre la estructura física y la configuración del bosque (como la topología del sitio). Ambas se replican a todos los controladores de dominio en el bosque. La partición *Dominio* contiene todos los objetos creados en ese dominio y se replica solo en los controladores de dominio dentro de su dominio. Entonces, por ejemplo, un usuario creado en el Dominio X solo figuraría en los controladores de dominio del Dominio X. Un subconjunto de objetos en la partición de dominio se replica en controladores de dominio que están configurados como **catálogos globales** (*Global Catalog*, GC). Los servidores del catálogo global (GC) proporcionan una lista global de todos los objetos en el bosque. Los servidores del catálogo global se replican a sí mismos todos los objetos de todos los dominios y, por lo tanto, proporcionan una lista global de objetos en el bosque. Sin embargo, para minimizar el tráfico de replicación y mantener pequeña la base de datos del GC, solo se replican los atributos seleccionados de cada objeto. Esto se llama el conjunto de atributos parciales (*Partial Attribute Set*, PAS). El PAS puede modificarse cambiando el esquema y marcando los atributos para la replicación al GC. Las versiones anteriores de Windows usaban NetBIOS para comunicarse. *Active Directory* está totalmente integrado con DNS y requiere TCP/IP-DNS. Para ser completamente funcional, el servidor DNS debe admitir registros de recursos SRV o registros de servicio.

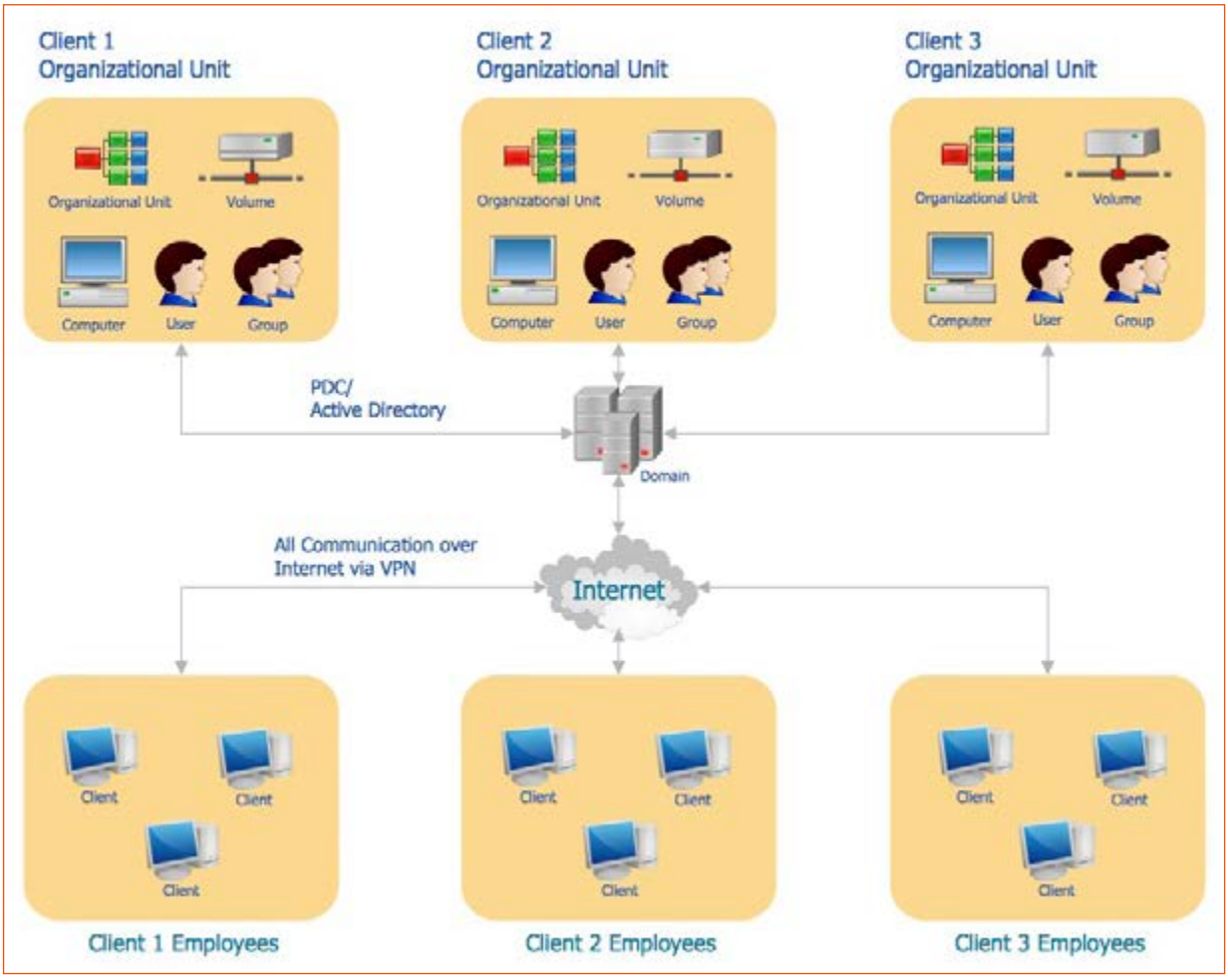

Figura 9. Arquitectura de un dominio en Active Directory. Adaptado de https://conceptdraw.com/

3.3. Integración de sistemas Unix/Linux y Windows: Samba

Samba es una solución de *software* libre para lograr que los sistemas de archivos Unix estén disponibles para los sistemas operativos Windows. Windows NT utiliza un sistema de intercambio de archivos de red basado en su propio protocolo SMB (*Server Message Block*). Samba es un servicio basado en demonio de Unix que hace que los discos de Unix sean visibles para Windows NT. Samba asigna nombres de usuario, por lo que para usar Samba se necesita una cuenta con el mismo nombre en el servidor NT y en el servidor Unix. Mapea los nombres de usuario textualmente, sin mucha seguridad. La configuración de Samba está en estilo Unix, editando el archivo de texto `/etc/smb.conf`. A continuación, se muestra un archivo de ejemplo. Se debe tener especial cuidado con la línea "*host allow*" que restringe el acceso a los discos a direcciones IP específicas, como envoltorios TCP (Burgess, 2004).

```
[global]
printing = bsd
printcap name = /etc/printcap
load printers = yes
guest account = nobody
invalid users = root
```

```
workgroup = UNIX
hosts allow = 128.39.

[homes]
comment = Home Directories
browseable = no
read only = no
create mode = 0644

[printers]
comment = All Printers
browseable = no
path = /tmp
printable = yes
public = no
writable = no
create mode = 0644
```

Una vez que el servidor Samba está activo, los discos se configuran para su uso en Windows usando el comando `net use`, por ejemplo, de la siguiente manera:

```
C:\> net use F: \\host\directory
```

Este ejemplo asigna el directorio nombrado en el *host* nombrado a la letra de unidad NT **F:**. El problema inverso de montar sistemas de archivos NT en un *host* Unix funciona solo para GNU/Linux con comandos como el siguiente:

```
system% smbmount //nthost/directory /mountpoint -U administrator
```

3.3.1. Autenticación *Active Directory* (AD) con Samba

La gente del proyecto Samba ha hecho grandes avances para proporcionar soporte de *Active Directory* para entornos Unix/Linux. Con la ayuda de Samba, los sistemas Linux pueden unirse a un dominio de *Active Directory* y permitir el acceso al sistema mediante cuentas definidas en AD que no tienen entradas en el archivo `/etc/passwd`. Los UID de Linux se derivan de sus identificadores de usuario análogos en Windows, conocidos como identificadores de seguridad o SID. Aprovechando PAM, se puede crear automáticamente un directorio de inicio para un usuario que aún no tiene uno. El sistema de integración incluso permite que el comando `passwd` cambie la contraseña AD de un usuario. Toda esta magia de integración de Windows es manejada por un componente de Samba llamado **winbind** (Nemeth, Snyder, Hein, & Whaley, 2011).

Active Directory adopta y extiende varios protocolos estándar, especialmente **LDAP** y **Kerberos**. En un intento por lograr el nirvana de la administración de sistemas, Microsoft desafortunadamente ha sacrificado el cumplimiento de los protocolos originales, creando una red llena de dependencias propietarias de RPC (*Remote Procedure Call*).

Para emular el comportamiento de un cliente de AD, *winbind* se conecta a PAM, NSS y Kerberos. Convierte las solicitudes de autenticación y de información del sistema en los formatos apropiados específicos de Microsoft. Desde el punto de vista de Unix, AD es solo otra fuente de información de directorio LDAP y datos de autenticación Kerberos.

Se deben completar las siguientes tareas de configuración antes de que un sistema Unix/Linux pueda ingresar a un dominio en *Active Directory* (Nemeth, Snyder, Hein, & Whaley, 2011):

- Instalar Samba con soporte para *Active Directory* y conversión de identidad.
- Configurar el conmutador de servicio de nombres, `nsswitch.conf`, para utilizar winbind como fuente de información de usuario, grupo y contraseña.
- Configurar PAM para atender las solicitudes de autenticación a través de winbind.
- Configurar *Active Directory* como un "reino" de Kerberos.

Los clientes AD de Unix/Linux también deben usar controladores AD para atender sus solicitudes de DNS y configurar sus relojes con NTP. Se debe asegurar también de que el nombre de dominio completo del sistema se encuentre en `/etc/hosts`. En algunos casos, también es necesario agregar la dirección IP del controlador de dominio a `/etc/hosts`. Sin embargo, esto último no es recomendable si se está utilizando *Active Directory* para el servicio DNS.

winbind es una excelente opción para sistemas Linux, pero Unix ha quedado en gran medida fuera de combate. Se ha reportado en sitios Unix que implementaron con éxito el mismo esquema general descrito aquí, pero cada sistema tiene algunas advertencias, y la integración tiende a no ser tan limpia como en Linux.

Samba se incluye de manera predeterminada en muchas de las distribuciones de Linux, pero algunas distribuciones no incluyen los **servicios de mapeo de identidad** necesarios para una implementación completa del cliente AD. Es posible que esos componentes se puedan configurar correctamente si está compilando desde el código fuente, por lo que se recomienda instalar paquetes binarios si están disponibles para su distribución. Los componentes de Samba, por otro lado, deben ser lo más actualizados posibles. La integración de AD es una de las características más nuevas de Samba, por lo que descargar el código fuente más reciente del sitio **samba.org** puede evitar errores frustrantes.

Si se compila Samba desde el código fuente, se recomienda configurar con los módulos compartidos `idmap_ad` e `idmap_rid`. El argumento apropiado para el script ./configure es:

```
--with-shared-modules=idmap_ad,idmap_rid
```

Se debe compilar e instalar Samba con la secuencia familiar `make`, `sudo make install`. Cuando se instala correctamente, la biblioteca **winbind** se deposita en `/lib`:

```
system$ ls -l /lib/libnss_winbind.so.2
-rw-r--r-- 1 root root 21884 2009-10-08 00:28 /lib/libnss_winbind.so.2
```

El demonio **winbind** se detiene y se inicia mediante los procedimientos normales del sistema operativo. Debe reiniciarse después de los cambios en `nsswitch.conf`, `smb.conf` o el archivo de configuración de Kerberos, `krb5.conf`. No es necesario iniciarlo hasta que estos otros servicios se hayan configurado.

3.3.2. Configurando Kerberos para Integración con *Active Directory*

Kerberos es infame por su configuración compleja, particularmente en el lado del servidor. Afortunadamente, solo se necesita en este caso configurar el lado del cliente de Kerberos, que es una tarea mucho más fácil. El archivo de configuración es `/etc/krb5.conf`.

Primero, se debe verificar que el nombre de dominio completo del sistema se haya incluido en `/etc/hosts` y que NTP esté utilizando un servidor de *Active Directory* como referencia de tiempo. Luego, editar `krb5.conf` para agregar el reino como se muestra en el siguiente ejemplo, donde se usa el dominio AD "ULSAH.COM".

```
[logging]
default = FILE:/var/log/krb5.log
[libdefaults]
clockskew = 300
default_realm = ULSAH.COM
kdc_timesync = 1
ccache_type = 4
forwardable = true
proxiable = true
[realms]
ULSAH.COM = {
kdc = dc.ulsah.com
admin_server = dc.ulsah.com
default_domain = ULSAH
}
[domain_realm]
.ulsah.com = ULSAH.COM
ulsah.com = ULSAH.COM
```

Varios valores son de interés en el ejemplo anterior. Se permite un sesgo de reloj de 5 minutos a pesar de que la hora se establece a través de NTP. Esto da cierta holgura en caso de un problema de NTP. El reino (*realm*) predeterminado se establece en el dominio AD, y el centro de distribución de claves (*Key Distribution Center*, KDC), se configura como un controlador de dominio AD. `krb5.log` puede ser útil para la depuración.

Se puede solicitar un *ticket* del controlador de *Active Directory* ejecutando el comando **kinit**. Se debe especificar una **cuenta de usuario de dominio válida**. "Administrator" suele ser una buena prueba, pero cualquier cuenta servirá. Cuando se solicite, se ha de escribir la contraseña del dominio.

```
system$ kinit administrator@ULSAH.COM
Password for administrator@ULSAH.COM: <password>
```

Con el commando **klist** se puede observar el ticket Kerberos:

```
system$ klist
Ticket cache: FILE:/tmp/krb5cc_1000
Default principal: administrator@ULSAH.COM
Valid starting Expires Service principal
10/11/09 13:40:19 10/11/09 23:40:21 krbtgt/ULSAH.COM@ULSAH.COM
renew until 10/12/09 13:40:19
Kerberos 4 ticket cache: /tmp/tkt1000
klist: You have no tickets cached
```

Si se muestra un *ticket*, la autenticación fue exitosa y se ha configurado Kerberos correctamente. En este caso, el *ticket* es válido por 10 horas y puede renovarse por 24 horas (se puede usar el comando **kdestroy** para invalidar el *ticket*).

3.3.3. Samba como un miembro de dominio de *Active Directory*

Al igual que otros componentes de Samba, **winbind** se configura en el archivo `smb.conf`. Se configura Samba como miembro de dominio AD con la opción `security = ads`.

Una configuración de ejemplo se muestra a continuación. Se establece el reino Kerberos y se apunta la autenticación Samba al controlador de dominio. También se configura la asignación de identidad del usuario con las opciones de `idmap` de `smb.conf`. Tener en cuenta que la configuración de recursos compartidos individuales en `smb.conf` es independiente de la configuración de los servicios de autenticación de AD. Aquí solo se muestra la autenticación de dominio.

```
[global]
security = ads
realm = ULSAH.COM
password server = 192.168.7.120
workgroup = ULSAH
winbind separator = +
idmap uid = 10000-20000
idmap gid = 10000-20000
winbind enum users = yes
winbind enum groups = yes
template homedir = /home/%D/%U
template shell = /bin/bash
client use spnego = yes
client ntlmv2 auth = yes
encrypt passwords = yes
winbind use default domain = yes
restrict anonymous = 2
```

De particular interés es la opción `winbind use default domain`. Si se usan varios dominios de AD, este valor debería ser **no**. Sin embargo, si se usa solo un dominio, establecer este valor en **sí** permite omitir el dominio durante la autenticación (por ejemplo, se puede usar "ben" en lugar de "ULSAH\ben"). Además, el valor del `windbind separator` especifica una alternativa a la barra diagonal inversa cuando se escriben los nombres de usuario. El valor del parámetro `workgroup` debe ser el nombre corto del dominio. Un dominio como "linux.ulsah.com" usaría LINUX como el valor del **workgroup**.

Después de configurar Samba, reiniciar los servicios **Samba** y **winbind** para que esta nueva configuración surta efecto.

Finalmente es hora de unir el sistema al dominio. Se puede usar la herramienta de red proporcionada por Samba, que toma prestada su sintaxis del comando de Windows del mismo nombre. **net** acepta varios protocolos para comunicarse con Windows. Se usa la opción `ads` para apuntar a *Active Directory*.

Se puede revisar la existencia de un ticket ejecutando **klist** (o solicitando uno nuevo con **kinit**). Luego usar el siguiente comando para unirse al dominio:

```
system$ sudo net ads join -S DC.ULSAH.COM -U administrator
Enter administrator's password: <password>
Using short domain name -- ULSAH
Joined 'SYSTEM' to realm 'ulsah.com'
```

En el ejemplo se ha especificado un servidor AD, **dc.ulsah.com**, en la línea de comando (no estrictamente necesario) y la cuenta de administrador. De forma predeterminada, AD agrega el nuevo sistema a la unidad organizativa del equipo de la jerarquía de dominio. Si el sistema aparece en la unidad organizativa (OU) de equipos dentro de la herramienta Usuarios y equipos de AD de Windows, la operación de unión de dominio se realizó correctamente. También se puede examinar el estado del sistema con el comando `net ads status`. En la página de manual del comando **net** se pueden ver opciones adicionales, incluidas las **operaciones de búsqueda LDAP**.

La configuración del conmutador de servicio de nombres es simple. Los archivos del sistema `passwd` y `group` siempre deben consultarse primero, pero luego se puede atajar a *Active Directory* a través de **winbind**. Estas entradas en `nsswitch.conf` hacen el truco:

```
passwd: compat winbind
group: compat winbind
shadow: compat winbind
```

Una vez que se ha configurado NSS, se puede probar la resolución de usuarios y grupos de AD con el comando **wbinfo**. Usando **wbinfo -u** se puede ver una lista de los usuarios del dominio y **wbinfo -g** para ver grupos. El comando `getent passwd` muestra las cuentas de usuario definidas en todas las fuentes, en formato `/etc/passwd`:

```
system$ getent passwd
root:x:0:0:root:/root:/bin/bash
```

```
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
...
bwhaley:x:10006:10018::/home/bwhaley:/bin/sh
guest:*:10001:10001:Guest:/home/ULSAH/guest:/bin/bash
ben:*:10002:10000:Ben Whaley:/home/ULSAH/ben:/bin/bash
krbtgt:*:10003:10000:krbtgt:/home/ULSAH/krbtgt:/bin/bash
```

La única forma de distinguir a los usuarios locales de las cuentas de dominio es por el ID de usuario (UID) y por la ruta ULSAH en el directorio de inicio, lo cual es evidente en las últimas tres entradas anteriores. Si el sitio tiene múltiples dominios o si la opción `winbind use default domain` no está establecida, el nombre de dominio corto se antepone a las cuentas de dominio (por ejemplo, `ULSAH\ben`).

3.3.4. Configuración de PAM

En este punto, el sistema se ha configurado para comunicarse con *Active Directory* a través de Samba, pero la autenticación no se ha configurado. Configurar PAM para realizar la autenticación a través de *Active Directory* es un poco complicado, principalmente porque los detalles difieren ampliamente entre las distribuciones de Linux.

La idea general es configurar **winbind** como un módulo de autenticación para todos los servicios que deberían tener soporte para *Active Directory*. Algunas distribuciones, como **Red Hat**, configuran convenientemente todos los servicios en un solo archivo. Otros, como Ubuntu, utilizan en varios archivos. La Tabla 10 enumera los archivos apropiados en varias distribuciones de Linux de ejemplo.

Tabla 10
Archivos de configuración de PAM para soporte de winbind

Distribución	Autenticación	Sesión
ubuntu	`common-account, common-auth, sudo`	`common-session`
SUSE	`common-auth, common-password, common-account`	`common-session`
Red Hat	`system-auth`	`system-auth`

Nota. Adaptado de Nemeth, Snyder, Hein, & Whaley (2011).

Para habilitar la autenticación **winbind**, añadir al principio de cada archivo:

```
auth sufficient pam_winbind.so
```

Una excepción es el archivo **common-password** de SUSE, en el que se debe reemplazar la palabra clave `auth` por `password`:

```
password sufficient pam_winbind.so
```

PAM puede crear directorios de inicio automáticamente si no existen cuando un nuevo usuario (en el sistema) inicia sesión. Dado que los usuarios de *Active Directory* no se agregan mediante el comando

useradd estándar, que normalmente es responsable de crear directorios de inicio, esta característica es bastante útil. Se debe agregar la siguiente línea en el archivo de configuración de sesión de PAM como se indica en la Tabla 10:

```
session required pam_mkhomedir.so umask=0022 skel=/etc/skel
```

Con esta configuración, PAM crea directorios de inicio con permisos octales 755 y con perfiles de cuenta copiados de `/etc/skel`.

También es posible que se desee restringir el acceso al sistema local a los usuarios que se encuentran en un grupo particular de *Active Directory*. Para hacerlo, se debe agregar la siguiente línea al archivo de configuración de sesión de PAM:

```
session required /lib/security/$ISA/pam_winbind.so use_first_pass
require_membership_of=unix_users
```

Aquí, solo los usuarios del grupo AD **unix_users** pueden iniciar sesión.

3.4. Computación en la nube (*Cloud Computing*)

La computación en la nube (*Cloud Computing*) es la entrega de la capacidad informática y de almacenamiento como un servicio a una comunidad de usuarios. El nombre proviene del uso de un símbolo en forma de nube en los diagramas de sistemas como una abstracción de la complejidad que puede contener una red o una infraestructura tecnológica. La computación en la nube confía los servicios con datos, *software* y computación de un usuario a través de una red.

Existen diferentes modelos de servicios de computación en la nube, de los que resaltan los siguientes (Pediapress, 2012):

- Infraestructura como Servicio (*Infrastructure as a Service*, IaaS)
- Plataforma como Servicio (*Platform as a Service*, PaaS)
- Software como Servicio (*Software as a Service*, SaaS)

Por ejemplo, al usar el *software* como servicio (SaaS), los usuarios alquilan *software* de aplicación y bases de datos alojado en una infraestructura externa a las estaciones de trabajo (comúnmente en Internet, o "la nube"). Los proveedores de la nube administran la infraestructura y las plataformas en las que se ejecutan las aplicaciones.

Los usuarios finales acceden a aplicaciones basadas en la nube a través de un navegador web o una aplicación liviana de escritorio o móvil, mientras que el *software* comercial y los datos del usuario se almacenan en servidores en una ubicación remota. Los defensores de la computación en la nube afirman que permite a las empresas poner en funcionamiento sus aplicaciones más rápido, con una capacidad de administración mejorada y menos esfuerzos de mantenimiento, y le permite a IT ajustar más rápidamente los recursos para satisfacer la demanda comercial fluctuante e impredecible.

La computación en la nube se basa en el intercambio de recursos para lograr coherencia y economías de escala similares a las de una empresa de servicios públicos (como la red eléctrica) a través de una red (generalmente Internet). La base de la computación en la nube es el concepto más amplio de infraestructura convergente y servicios compartidos. La disponibilidad ubicua de redes de alta capacidad, computadoras de bajo costo y dispositivos de almacenamiento, así como la adopción generalizada de virtualización de *hardware*, arquitectura orientada a servicios, informática autónoma y de servicios públicos han llevado a un tremendo crecimiento de la computación en la nube. Las organizaciones están cambiando los modelos con *hardware* y *software* como activos propiedad de la compañía, a **modelos basados en servicios por uso**, de tal manera que el cambio proyectado a la informática resultará en un crecimiento dramático en los productos de IT en algunas áreas y reducciones significativas en otras áreas.

En los últimos años los modelos de computación en la nube se están popularizando cada vez más, en la medida en que aumentar los anchos de banda de acceso a las redes, y las instalaciones de infraestructura y servicios en premisas (*on-premises*) se hacen más costosas y difíciles de administrar. **La ingeniería de la nube** (*Cloud Engineering*) viene a ser la aplicación de disciplinas de ingeniería a la computación en la nube. Aporta un enfoque sistemático a las preocupaciones de alto nivel de comercialización, estandarización y gobierno, en la concepción, desarrollo, operación y mantenimiento de sistemas informáticos en la nube. Es una nueva metodología multidisciplinaria que abarca contribuciones de diversas áreas, como sistemas, *software*, web, rendimiento, información, seguridad, plataforma, riesgo e ingeniería de calidad.

3.4.1. Características de la computación en la nube

La computación en la nube presenta las siguientes características clave (Wikiwand.com, s.f.):

- **La agilidad** mejora con la capacidad de los usuarios para reaprovisionar recursos de infraestructura tecnológica.

- **Accesibilidad a las interfaces de programación de aplicaciones (*Application Programming Interfaces*, API)** al *software* que permite que las máquinas interactúen con el *software* en la nube de la misma manera que la interfaz de usuario facilita la interacción entre humanos y computadoras. Los sistemas de computación en la nube generalmente usan APIs basadas en REST.

- **El costo** se afirma que se reduce y, en un modelo de entrega en la nube pública, el gasto de capital o de propiedad se convierte en gasto operativo. Esto se supone reduce las barreras de entrada, ya que la infraestructura generalmente es proporcionada por un tercero y no se requiere comprar *hardware* especial para la ejecución de tareas informáticas poco frecuentes. Los precios en este modelo de "informática de servicios" son granulares, con opciones basadas en el uso, y se requieren menos habilidades de IT para la implementación in la organización.

- **La independencia la ubicación y del dispositivo** permite a los usuarios acceder a los sistemas utilizando un navegador web, independientemente de su ubicación o qué dispositivo están utilizando (por ejemplo, PC o un teléfono móvil inteligente). Como la infraestructura está fuera del sitio (generalmente proporcionada por un tercero) y se accede a través de Internet, los usuarios pueden conectarse desde cualquier lugar.

- **La tecnología de virtualización** permite compartir servidores y dispositivos de almacenamiento físicos, aumentando la eficiencia de utilización de estos recursos. Las aplicaciones se pueden migrar fácilmente de un servidor físico a otro.

- **La naturaleza mutiusuario y multiempresa** permite compartir recursos y costos en un gran grupo de usuarios, lo que permite:

  - **Centralización de la infraestructura** en ubicaciones con costos más bajos (como bienes raíces, electricidad, etc.).

  - **La capacidad de carga máxima aumenta** (los usuarios no necesitan diseñar los niveles de carga más altos posibles).

  - **Mejoras en la utilización** y la eficiencia para sistemas que a menudo solo se utilizan del 10 al 20%.

- **La confiabilidad** mejora si se usan múltiples sitios redundantes, lo que hace que la computación en la nube bien diseñada sea adecuada para la continuidad del negocio y la recuperación ante desastres.

- **Escalabilidad y elasticidad** a través del aprovisionamiento dinámico "bajo demanda" (*on-demand*) de recursos sobre una base de autoservicio de alta granularidad y casi en tiempo real, sin que los usuarios tengan que diseñar cargas máximas.

- **Monitorización del rendimiento**, construyendo arquitecturas consistentes y desacopladas, utilizando servicios web como interfaces del sistema.

- **La seguridad** podría mejorar debido a la centralización de los datos, el aumento de los recursos centrados en la seguridad, etc. Pero las preocupaciones pueden persistir sobre la pérdida de control sobre ciertos datos confidenciales y la falta de seguridad para los núcleos almacenados. La seguridad es a menudo tan buena o mejor que otros sistemas tradicionales, en parte porque los proveedores pueden dedicar recursos para resolver problemas de seguridad que muchos clientes no pueden permitirse. Sin embargo, la complejidad de la seguridad aumenta considerablemente cuando los datos se distribuyen en un área más amplia o en un mayor número de dispositivos y en sistemas multi-empresariales que están siendo compartidos por usuarios no relacionados. Además, el acceso de los usuarios a los registros de auditoría de seguridad puede ser difícil o imposible. Las instalaciones en **la nube privada** están en parte motivadas por el deseo de los usuarios de mantener el control sobre la infraestructura y evitar perder el control de la seguridad de la información.

- **El mantenimiento** de las aplicaciones de computación en la nube es más fácil, ya que no necesitan instalarse en la computadora de cada usuario y pueden accederse desde diferentes lugares.

3.4.2. Modelos de servicios computación en la nube

Como se mencionó anteriormente, los proveedores de computación en la nube ofrecen sus servicios de acuerdo con tres modelos fundamentales: **Infraestructura como servicio** (IaaS), **Plataforma como servicio** (PaaS) y **Software como servicio** (SaaS) donde IaaS es el más básico, y cada uno de modelos superiores se abstrae a partir de los detalles de los modelos inferiores (ver Figura 10).

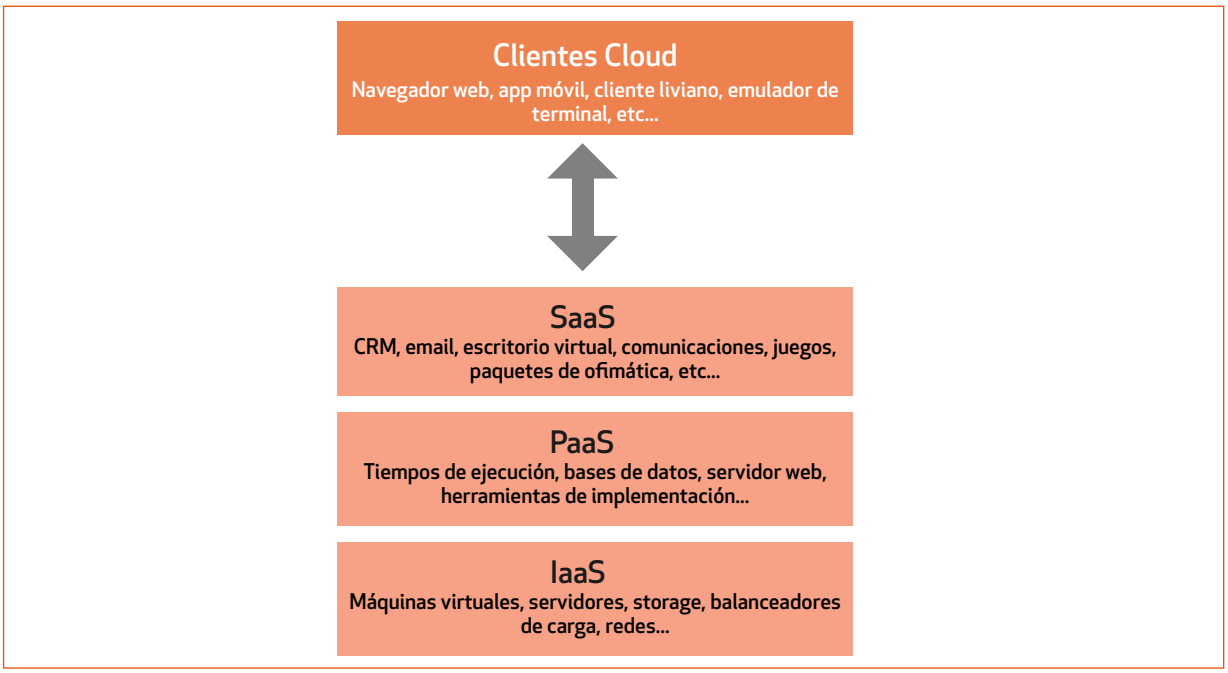

Figura 10. Niveles de los modelos de servicios en computación en la nube. Adaptado de http://www.wikiwand.com/

Infraestructura como servicio (IaaS)

En este modelo de servicio en la nube, los proveedores ofrecen computadoras, tanto físicas como en máquinas virtuales, y otros recursos de infraestructura. Las máquinas virtuales se ejecutan como invitados por un hipervisor, como XEN, KVM, VMWare o Hyper-V. La gestión de grupos de hipervisores por el sistema de soporte operativo en la nube permite escalar para admitir una gran cantidad de máquinas virtuales. Otros recursos en las nubes IaaS incluyen bibliotecas de imágenes de máquinas virtuales, almacenamiento (*storage*) en bloques y basado en archivos, *firewalls*, balanceadores de carga, direcciones IP, VLANs y paquetes de *software*. Los proveedores de nube de IaaS suministran estos recursos bajo demanda desde sus grandes conjuntos de recursos instalados en centros de datos. Para una conectividad WAN se puede usar Internet, se pueden configurar redes privadas virtuales dedicadas sobre nubes de operadores de redes.

Para desplegar sus aplicaciones, los usuarios de la nube luego instalan imágenes del sistema operativo en las máquinas, así como su *software* de aplicación. En este modelo, es el usuario de la nube el responsable de aplicar parches y mantener los sistemas operativos y el *software* de la aplicación alojados en la infraestructura del proveedor IaaS. Dichos proveedores suelen facturar los servicios de IaaS en función de la computación de servicios públicos, es decir, el costo reflejará la cantidad de recursos asignados y consumidos.

IaaS no se refiere a una máquina que hace todo el trabajo, sino simplemente a una instalación dada a las empresas que ofrece a los usuarios el espacio de almacenamiento adicional en servidores y centros de datos. Ejemplos de proveedores IaaS incluyen: *AWS CloudFormation* (y servicios subyacentes como *Amazon EC2*, *Amazon RDS* o *Amazon S3*), *Rackspace Cloud*, y *Google Compute Engine*.

Plataforma como servicio (PaaS)

En el modelo PaaS, los proveedores de la nube ofrecen una plataforma informática que generalmente incluye un sistema operativo, entornos de ejecución de lenguajes de programación, base de datos y servidor web. Los desarrolladores de aplicaciones pueden desarrollar y ejecutar sus soluciones de software en una plataforma en la nube sin el costo y la complejidad de comprar y administrar las capas subyacentes de *hardware* y *software*. Con algunas ofertas de PaaS, la computadora subyacente y los recursos de almacenamiento se escalan automáticamente para satisfacer la demanda de la aplicación, de modo que el usuario de la nube no tenga que asignar recursos manualmente. Ejemplos de PaaS incluyen: *Amazon Elastic Beanstalk*, *Heroku*, *EngineYard*, *Google App Engine* y *Microsoft Azure*.

***Software* como servicio (PaaS)**

En este modelo, los proveedores de la nube instalan y operan *software* de aplicación en la nube y los usuarios de la nube acceden al *software* desde los clientes de la nube. Los usuarios de la nube no administran la infraestructura y la plataforma en la nube en la que se ejecuta la aplicación. Esto elimina la necesidad de instalar y ejecutar la aplicación en las propias computadoras del usuario de la nube, lo que simplifica el mantenimiento y el soporte. **Lo que hace que una aplicación en la nube sea diferente de otras aplicaciones es su elasticidad.** Esto se puede lograr mediante la clonación de tareas en múltiples máquinas virtuales en tiempo de ejecución para satisfacer la demanda de trabajo cambiante. Los balanceadores de carga distribuyen el trabajo sobre el conjunto de máquinas virtuales. Este proceso es discreto para el usuario de la nube que solo ve un único punto de acceso. Para dar cabida a una gran cantidad de usuarios de la nube, las aplicaciones en la nube pueden ser multi-empresa, es decir, cualquier máquina sirve a más de una organización de usuarios de la nube. Es común referirse a tipos especiales de *software* de aplicación basado en la nube con una convención de nomenclatura similar, por ejemplo: escritorio como servicio (*Desktop as a Service*, DaaS), proceso comercial como servicio (*Business Process as a Service*, BPaaS), Ambiente de pruebas como servicio (*Test Environment as a Service*, TEaaS), Comunicación como servicio (*Communication as a Service*, CaaS), entre otros. El modelo de precios para aplicaciones SaaS suele ser una **tarifa plana mensual o anual por usuario**. Ejemplos de proveedores SaaS incluyen: *Google Apps*, *Quickbooks Online* y *Salesforce.com*.

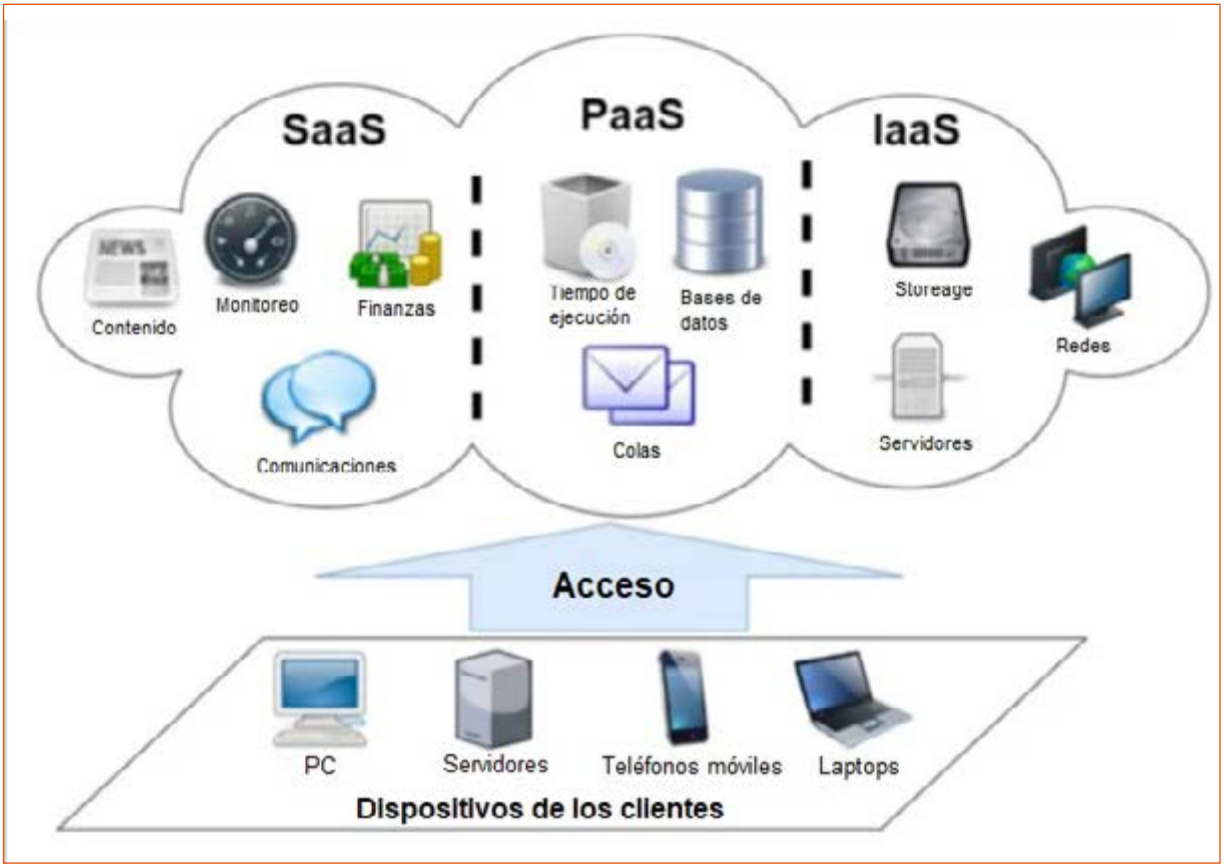

Figura 11. Servicios de computación en la nube. Adaptado de Chin-Ling, Chiang, & Lin (2020).

3.4.3. Modelos de implementación de computación en la nube

Dependiendo de la localización y propiedad de los recursos sobre los cuáles se despliegan los servicios de computación en la nube, puede haber diferentes modelos, los cuales se explican a continuación.

Nube Pública (*Public Cloud*)

Un proveedor de servicios pone a disposición del público en general las aplicaciones, el almacenamiento y otros recursos de la nube pública, generalmente a través de Internet. Estos servicios son gratuitos u ofrecidos en un modelo de pago por uso. En general, los proveedores de servicios de nube pública como **Amazon Web Service**, **Microsoft** y **Google** poseen y operan la infraestructura y ofrecen acceso solo a través de Internet (no se ofrece conectividad directa). En este sentido son claras las ventajas, al liberar a la organización de la adquisición y mantenimiento de infraestructura de forma local, siendo esta provista de forma "elástica" o por uso, con esquemas de costes bastante convenientes.

Por otro lado, se tiene como desventaja la dependencia de los servicios en línea y la necesidad de acceso a través de Internet. Las empresas deben tener unas buenas infraestructuras de conectividad para evitar problemas de conexión. Se debe estudiar bien la integración de trabajar en la nube con los sistemas propietarios que deben coexistir al mismo tiempo. Se ha de prever que el proveedor pueda garantizar la obtención de todos los datos almacenados en la nube para poder disponer de ellos en el menor tiempo posible, así como la seguridad de los mismos (Descom.es, s.f.).

Nube Privada (*Private Cloud*)

La nube privada es la infraestructura de la nube operada únicamente para una sola organización, ya sea administrada internamente o por un tercero y alojada interna o externamente. La realización de un proyecto de nube privada requiere un nivel y un grado de compromiso significativos para virtualizar el entorno empresarial, y requerirá que la organización evalúe las decisiones sobre los recursos existentes. Cuando se hace bien, puede tener un impacto positivo en un negocio, pero cada uno de los pasos del proyecto plantea problemas de seguridad que deben abordarse para evitar vulnerabilidades graves. Han atraído críticas porque los usuarios aún tienen que comprar, construir y administrar los recursos y, por lo tanto, no hay en principio beneficios como la descarga en las tareas de gestión, básicamente desaprovechando el modelo económico que hace que la computación en la nube sea un concepto tan atractivo.

Nube Híbrida (*Hybrid Cloud*)

La nube híbrida es una composición de dos o más nubes (privadas o públicas) **que siguen siendo entidades únicas, pero están unidas**, ofreciendo los beneficios de múltiples modelos de implementación. Al utilizar la arquitectura de "nube híbrida", las empresas y los individuos pueden obtener grados de tolerancia a fallas combinados con usabilidad local inmediata sin depender de la conectividad a Internet. La arquitectura de nube híbrida requiere recursos locales e infraestructura de nube basada en servidor (remoto) fuera del sitio.

Las nubes híbridas combinan los servicios locales con los ofrecidos externamente por una nube pública. Las empresas pueden dividir sus servicios para mantener unos en modo local y privado dentro de la empresa y aprovechar otros servicios que la nube pública le ofrece y que pueden ser aprovechados donde se necesiten. La nube híbrida es un paso intermedio para pasar o crear nuevos servicios en la nube pública. Por ejemplo, es muy común que muchas empresas utilicen **servicios de almacenamiento en la nube pública** para almacenar ciertos datos que considere que necesita tener de forma accesible externamente (Descom.es, s.f.).

En la Figura 12 se observa la idea de la nube híbrida, como **la interconexión** de una nube privada con una nube pública, implementando servicios controlados y mantenidos por la empresa, combinados con servicios ofrecidos por proveedores de nubes públicas.

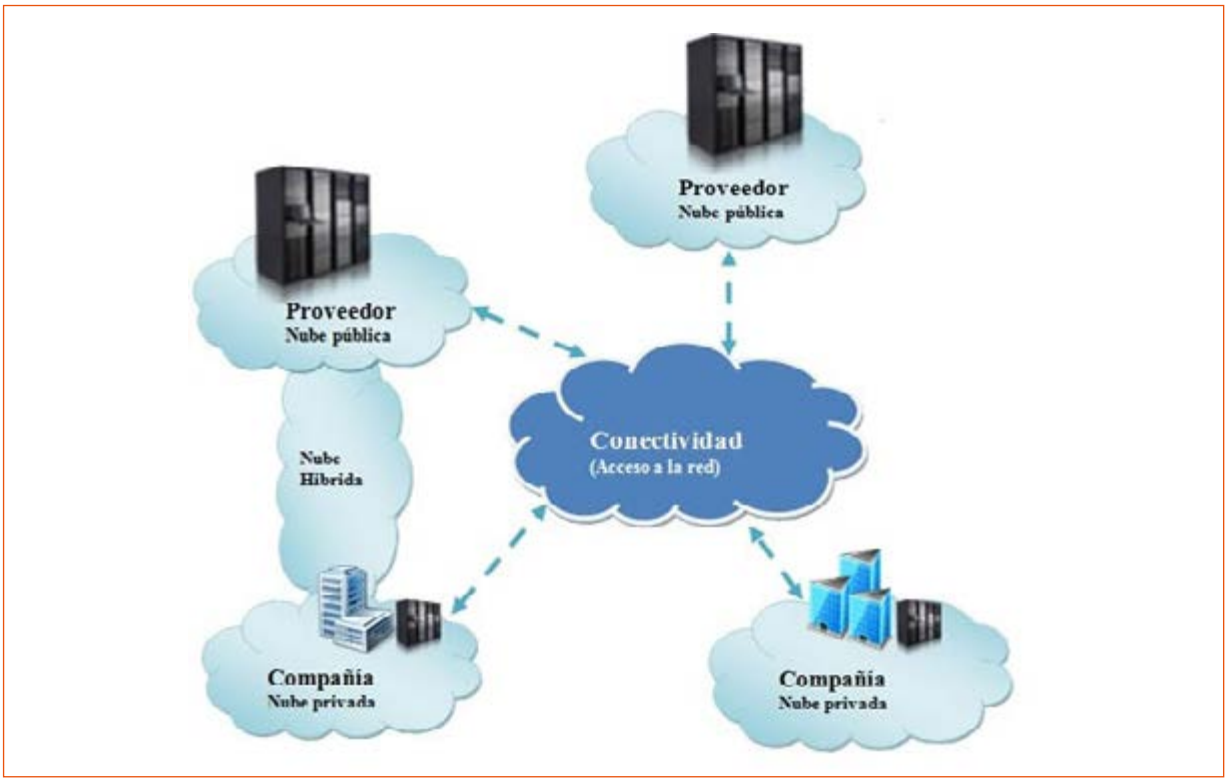

Figura 12. Modelos de implementación de computación en nube: pública, privada e híbrida. Recuperado de https://www.descom.es/

Tema 4. Copias de seguridad y recuperación

Todos los administradores de sistemas odian la tarea de realizar **copias de seguridad o respaldos**. Son procesos por lo general costosos. Los servicios se ejecutan más lentamente (o se bloquean) cuando se realizan copias de seguridad de los servidores. Por otro lado, a los clientes les encantan las restauraciones. De hecho, la eventual necesidad de realizar las restauraciones son el porqué de la realización de copias de seguridad.

Poder restaurar datos perdidos es una parte crítica de cualquier sistema o entorno de trabajo. Son muchas las razones que pueden ocasionar la pérdida de datos: Pérdida casual; los equipos fallan; los usuarios los eliminan por error (o con intención); los jueces confiscan todos los documentos relacionados con una demanda que se almacenaron en las computadoras en una fecha determinada; los accionistas requieren la tranquilidad de saber que un desastre natural u otro no hará que su inversión no tenga valor; etc. Los datos también se corrompen por error, a propósito, o por radiación electromagnética. Las copias de seguridad son como un seguro: se paga por ellas, aunque se espera no necesitarlas nunca. En realidad, son una necesidad.

En este capítulo se estudian los aspectos principales a tomar en cuenta para la realización de copias de seguridad y restauración de datos en la administración de sistemas, sin enfocarse en comandos específicos de algún sistema operativo.

4.1. Aspectos básicos: Planificación

La administración del sistema de copia de seguridad y restauración debe comenzar determinando el resultado final deseado y trabajando hacia atrás desde allí. El resultado final es la capacidad de restauración deseada del sistema. Las restauraciones se solicitan por varias razones, y las razones que se aplican a un entorno específico afectan decisiones adicionales, como la **creación de políticas y cronogramas** para las copias de seguridad y los procesos de restauración. En este sentido, se puede comenzar estableciendo **pautas corporativas** para definir **acuerdos de nivel de servicio** (*Service Level Agreement*, SLA) para ejecutar restauraciones basadas en las necesidades del sitio, las cuales se convierten en políticas de seguridad a partir de las cuales se definen los cronogramas de respaldo. Una guía para este proceso puede ser la siguiente (Limoncelli, Hogan, & Chalup, 2007):

- Las pautas corporativas definen la terminología y dictan los mínimos y requisitos para los sistemas de recuperación de datos.
- El SLA define los requisitos para un sitio o aplicación en particular y se guía por las pautas corporativas.
- La política documenta la implementación del SLA en términos generales.
- El procedimiento describe cómo se implementará la política.
- La programación detallada muestra qué disco se respaldará y cuándo. Esto puede ser estático o dinámico. Esta suele ser la política traducida del español a la configuración del *software* de respaldo.

Solo después todas las definiciones anteriores, se puede desplegar el sistema de respaldo y restauración. Los sistemas de respaldo modernos tienen dos componentes clave: **automatización** y **centralización**.

La Tabla 11 resume las razones más comunes por las que se requiere restauración de datos en una organización.

Tabla 11
Razones comunes de restauración de datos

Razón de restauración	Descripción
Eliminación accidental de archivos	El usuario ha borrado accidentalmente uno o más archivos y necesita restaurarlos. La razón más común para solicitar una restauración es recuperarse de la eliminación accidental de archivos. Los sistemas de almacenamiento modernos pueden hacer que este tipo de restauración sea una función de autoservicio. Aún mejor, los sistemas sofisticados que proporcionan instantáneas no solo se encargan de esto sin requerir que el administrador del sistema participe en cada restauración, sino que también pueden afectar positivamente el entorno de trabajo del usuario.
Fallas de discos	El segundo tipo de razón de restauración está relacionado con la falla del disco, o cualquier falla de hardware o software que resulte en una **pérdida total o parcial del sistema de archivos**. Una falla de disco causa dos problemas: **pérdida de servicio** y **pérdida de datos**. En sistemas críticos, como el comercio electrónico y los sistemas financieros, la tecnología de discos RAID debe implementarse de modo que las fallas del disco no afecten el servicio, con la posible excepción de una pérdida de rendimiento.

Razón de restauración	Descripción
Archivo	Por razones comerciales, se debe hacer una copia instantánea (*snapshot*) de todos los datos de manera regular por razones de recuperación ante desastres, legales o fiduciarias. Las políticas corporativas pueden requerir que se sea capaz de reproducir todo el entorno con una granularidad de un trimestre, medio año o año completo en caso de desastres o demandas judiciales. Se desean las siguientes características para un adecuado archivo de los datos: • *Full-backup* • Almacenamiento fuera del sitio. • Si los archivos son parte de un plan de recuperación ante desastres, pueden aplicarse políticas o leyes especiales. Se debe asegurar que las herramientas necesarias para restaurar el archivo y la documentación requerida se almacenan junto con el archivo.

Nota. Adaptado de Limoncelli, Hogan, & Chalup (2007)

4.1.1. *Full-backup*, respaldo incremental, y respaldo diferencial

Se utiliza el término *full-backup* (copia de seguridad completa) para significar una copia de seguridad completa de todos los archivos en una partición. En **Unix** se le llama a esto una **copia de seguridad de nivel 0**. El término **respaldo incremental** se refiere a copiar todos los archivos que han cambiado desde la última ejecución de una copia de seguridad, sea esta un *full-backup* u otra incremental. En Unix a esto se le conoce como una **copia de seguridad de nivel 1**. Las copias de seguridad incrementales crecen con el tiempo. Es decir, si se realiza una copia de seguridad completa el domingo y una copia de seguridad incremental cada día de la semana siguiente, la cantidad de datos de la copia de seguridad debería crecer cada día porque la copia de seguridad incremental del martes incluye todos los archivos de la copia de seguridad del lunes, así como qué ha cambiado desde entonces. La copia de seguridad incremental del viernes debe incluir todos los archivos que formaron parte de las copias de seguridad del lunes, martes, miércoles y jueves, además de lo que cambió desde la copia de seguridad del jueves. Algunos sistemas realizan una copia de seguridad incremental que recopila todos los archivos modificados desde una copia de seguridad incremental particular en lugar de la última copia de seguridad completa. Tomando prestada la terminología de Unix, se llamarían en este caso **copias de seguridad incrementales de nivel 2** si contienen archivos modificados **desde el último nivel 1**, o **nivel 3** si contienen archivos modificados **desde el último nivel 2**, y así sucesivamente.

Finalmente se tiene el concepto de **respaldo diferencial**, que es el caso específico de una copia de los datos creados y modificados **desde la última copia de seguridad completa**.

4.1.2. Cronograma de respaldos

Un cronograma de respaldos debe específico y enumerar los detalles de las particiones de los hosts se respaldan y en qué momento. Aunque es raro que cambie frecuentemente un SLA, el cronograma de respaldos sí que podría cambiar con frecuencia. Muchos administradores de sistemas eligen especificar el cronograma directamente mediante la configuración del *software* de respaldo.

Comúnmente, las copias de seguridad deben realizarse **todos los días hábiles**. Incluso si la empresa experimenta una falla de disco no redundante y fallaron las copias de seguridad del último día, no se perderían más de 2 días de datos. Como las copias de seguridad completas (*full-backup*) demoran mucho más que las incrementales, típicamente se programan para los viernes por la noche y se dejan ejecutándose todo el fin de semana. De domingo a jueves por la noche, se realizan copias de seguridad incrementales.

En general, podría decirse que la frecuencia con la cual deben realizarse los respaldos depende de dos velocidades que compiten (Burgess, 2004):

- La velocidad a la que se producen nuevos datos
- La tasa esperada de pérdidas o fallos

Para la mayoría de los sitios, una copia de seguridad diaria es suficiente. En una zona de guerra, donde el riesgo de bombardeo es una amenaza en cualquier momento, podría ser necesario respaldar más a menudo. La mayoría de las organizaciones no producen grandes cantidades de datos todos los días. Sin embargo, otras organizaciones, como los laboratorios de investigación, recopilan datos automáticamente de instrumentos que serían prácticamente imposibles de recuperar. En ese caso, la importancia de la copia de seguridad sería aún mayor. Por supuesto, hay límites con qué frecuencia es posible hacer una copia de seguridad. La copia de seguridad es un proceso que requiere muchos recursos.

4.1.3. Automatización de respaldos

No automatizar las copias de seguridad es peligroso y una mala idea. Cuanto más se automatiza, más se elimina la posibilidad de error humano. Las copias de seguridad son aburridas y, si no están automatizadas, no se realizarán de manera confiable. Y si no se hacen correctamente, será muy vergonzoso tener que enfrentar la pregunta de un CEO: *¿Pero por qué no hubo copias de seguridad?* (Limoncelli, Hogan, & Chalup, 2007).

Se pueden automatizar principalmente dos aspectos del procedimiento de copias de seguridad: **los comandos** y **los cronogramas**. Cuando no había automatización los comandos individuales se escribían a mano cada vez que se realizaban copias de seguridad. A menudo, las copias de seguridad se iniciaban en el último turno antes del fin de la jornada para ejecutarse durante la noche. El horario era simple. Había poco o nada de inventario, excepto las etiquetas en las cintas. El primer paso en la automatización fueron los *scripts* que simplemente replicaban los comandos que antes se escribían manualmente, pero aun decidiendo qué datos respaldar y cuándo, lo cual todavía era una tarea humana. Pronto, se implementaron en ***software*** los algoritmos para automatizar cronogramas de respaldo. Dichos algoritmos han ido mejorando con el tiempo, ofreciendo horarios dinámicos que van más allá del alcance de lo que los humanos pueden cumplir razonablemente.

Tradicionalmente, las copias de seguridad se han hecho **de discos a cintas** (que son relativamente baratas y trasladables), pero la copia de seguridad en cinta es incómoda y difícil de automatizar a menos que puedan permitirse robots especializados para cambiar y administrar las cintas. Para sitios pequeños también es posible realizar la duplicación de disco. El disco es barato, mientras que los

operadores humanos son caros. Muchos sistemas de archivos modernos (por ejemplo, **DFS**) son capaces de duplicar automáticamente el disco en tiempo real.

Muchas de las estrategias o técnicas de automatización de respaldos están basadas en la instalación y configuración de agentes que estarán instalados y configurados en los hosts o servidores cuyos datos se desean respaldar. Dichos agentes manejarán las unidades de datos a respaldar en diferentes niveles, a saber:

- **A nivel de archivo:** cualquier cambio en un archivo causará que el mismo sea respaldado por completo.
- **A nivel de bloques:** sólo los bloques que cambian en un archivo son respaldados. Se requiere un procesamiento especial para detectar los cambios de bloques.

El resultado del respaldo de los agentes se puede almacenar en las unidades de discos locales de los *hosts* donde se ejecutan, o en unidades externas de almacenamiento centralizadas a través de la red, en cuyo caso convendría que los agentes trabajen a nivel de bloques, para realizar respaldos más granulares y disminuir el impacto en el tráfico de red minimizando la cantidad de datos a transferir. La posibilidad de realizar respaldos en red permite la centralización de la administración de las copias de seguridad, lo cual evidentemente resultará más beneficioso de cara a la protección de los datos y a la eficiencia de la administración, pero aumenta los costes y la complejidad. La evolución de los sistemas de almacenamiento (*storage*) ha permitido que esto sea una realidad, con esquemas cada vez más interesantes, que potencian la automatización y administración centralizada de las copias de seguridad, y de su eventual restauración luego de algún fallo. Algunos de estos esquemas se exponen en el siguiente apartado, de modo resumido, pues este tema tan extenso se escapa de los detalles a ofrecer en este material.

4.2. Centralización de la administración de respaldos: almacenamiento en red

Otro objetivo de diseño fundamental de los sistemas de respaldo y recuperación modernos es **la centralización**. Las copias de seguridad deben centralizarse porque son caras e importantes. Hacer las inversiones correctas puede distribuir el coste del sistema de copia de seguridad y restauración en muchos sistemas.

Sin centralización, se debe conectar una unidad de almacenamiento a cada máquina que se deba respaldar, y se debe pagar a alguien para que realice el proceso de respaldo en cada máquina. El resultado: pagar por muchas piezas caras de hardware y una gran cantidad de trabajo físico. **Los sistemas de respaldo basados en red** como son los sistemas de almacenamiento (*storage*) permiten conectar equipos de respaldo de gran capacidad a hosts que se contactan con otros para iniciar respaldos. Los sistemas de almacenamiento y respaldo en red han evolucionado gracias al desarrollo de redes de alta velocidad y fiabilidad, mayormente basadas en transmisiones sobre fibra óptica.

Existen básicamente dos modelos principales de almacenamiento en red: el NAS (*Network Attached Storage*) y el SAN (*Storage Area Network*). En términos generales, un **NAS es un equipo de**

almacenamiento individual que opera sobre los **archivos** de datos, mientras que **un SAN es una red local de múltiples dispositivos** que operan en **bloques** de disco.

Un NAS incluye un *hardware* dedicado que a menudo es el cerebro que se conecta a una red LAN. Este servidor NAS autentica clientes y gestiona las operaciones de archivos de la misma manera que los servidores de archivos tradicionales, a través de protocolos de red bien establecidos como NFS y SMB / CIFS. Para reducir el costo en comparación con los servidores de archivos tradicionales, los dispositivos NAS suelen ejecutar un sistema operativo embebido en el mismo *hardware*. Por otro lado, una SAN utiliza generalmente conexiones de fibra y conecta un conjunto de dispositivos de almacenamiento que son capaces de compartir datos entre sí a bajo nivel (Ferrando Lavila, 2017).

El administrador de una red doméstica o de pequeña empresa puede conectar un dispositivo NAS a su LAN. El NAS mantiene su propia dirección IP igual que los ordenadores y otros dispositivos de la red. Utilizando un *software* que normalmente se suministra con el dispositivo NAS, los administradores de la red pueden configurar copias de seguridad automáticas o manuales y copias de archivos entre el NAS y todos los otros dispositivos conectados. Un NAS suele disponer de mucha capacidad de almacenamiento, de hasta unos varios terabytes. Los administradores pueden añadir más capacidad de almacenamiento a la red mediante la instalación de dispositivos adicionales NAS, aunque cada NAS funciona de manera independiente. **Para infraestructuras de gran tamaño se pueden requerir mucha más capacidad en *terabytes* de almacenamiento centralizado o muchas operaciones de transferencia de archivos a alta velocidad.** Cuando la instalación de un ejército de muchos dispositivos NAS no es una opción práctica, los administradores pueden instalar en cambio un único SAN que contiene un conjunto de discos de alto rendimiento para proporcionar la escalabilidad y el rendimiento necesarios. En este caso, estos administradores requieren un conocimiento y capacitación especializada para configurar y mantener las redes SAN (Ferrando Lavila, 2017).

Una SAN, como se ha mencionado, más que un dispositivo es una red, sobre la que se comparten directamente unidades de almacenamiento sobre canales de alta velocidad basados en fibra. Uno de los protocolos dominantes para transmisiones de SAN es el Canal de Fibra (*Fibre-Channel*), aunque existe una nueva tendencia a utilizar protocolos LAN como *Ethernet* para la implementación de SAN, con lo que se estarían usando IP también para esta tecnología.

En la Figura 13 se ilustran las tecnologías de almacenamiento en redes de área local NAS y SAN, donde se puede observar que un NAS está directamente conectado a la LAN para ser accedido por los *host* y compartir a través de éste archivos, mientras que la SAN es una red aparte sobre la cual los servidores y otros dispositivos con tarjetas y puertos *Fibre-Channel* estarían compartiendo su almacenamiento. Se destaca aquí la posibilidad de colocar en la SAN un **equipo especial de almacenamiento para los respaldos**, que son controlados a través de *software* especial desde alguno de los servidores, los cuales a su vez estarían recibiendo los datos a ser respaldados desde los clientes de red. Se destaca en esta figura, además, la posibilidad, cada vez más común, de realizar almacenamiento y respaldo en internet con soluciones basadas en la nube.

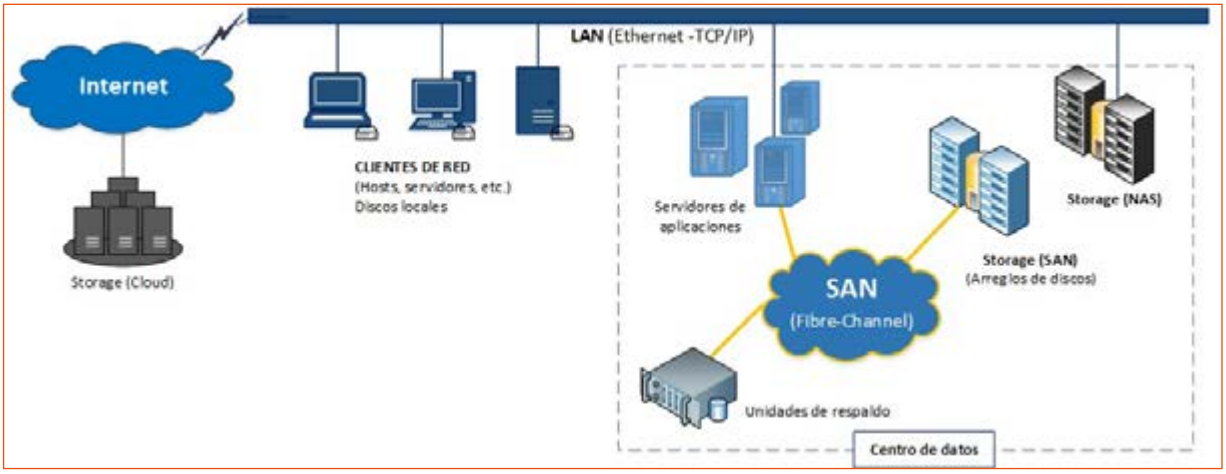

Figura 13. Almacenamiento en red con NAS y SAN. Elaboración propia.

La Tabla 12 enumera algunas diferencias básicas entre NAS y SAN, tomando en cuenta la forma de tratamiento y acceso de los datos, y el tipo de cableado o red subyacente.

Tabla 12
Diferencias entre NAS y SAN

	NAS	SAN
Tipo de datos	Archivos compartidos	Datos a nivel de bloque
Cableado utilizado	LAN	Fibra óptica
Clientes principales	Usuarios finales	Servidores de aplicaciones
Acceso a disco	A través del dispositivo NAS	Acceso directo

Nota. Adaptado de Ferrando Lavila (2017).

4.3. Restauración

La restauración se refiere a la recuperación de los datos previamente respaldados para su colocación en la ubicación primaria, o los discos de producción. El proceso puede ser de "cinta a disco", de "disco a disco", o de "nube a disco". El proceso de restauración normalmente formará parte de un proceso más general de recuperación, cuyos pasos generales se muestran en el diagrama de la Figura 14. El proceso de la recuperación de desastres se estudia en el próximo apartado.

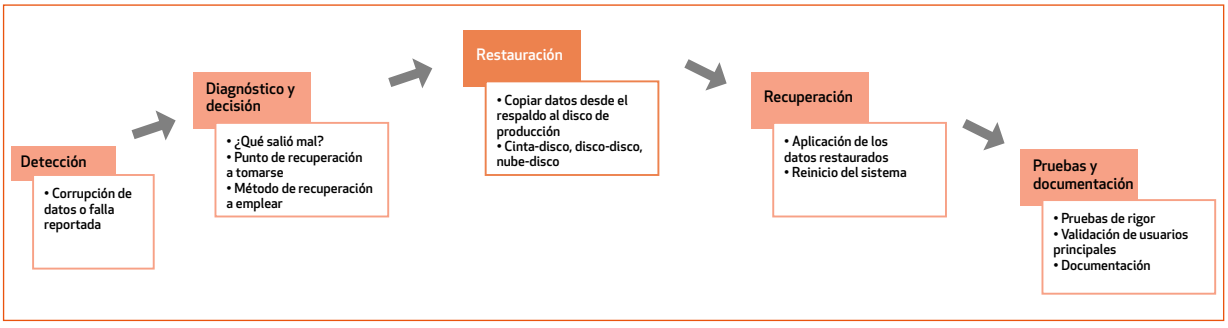

Figura 14. Proceso general de recuperación de fallos en sistemas. Elaboración propia.

4.4. Recuperación

La capacidad de recuperación de la pérdida presupone que se tienen suficientes piezas del sistema para reconstruirlo, en caso de desastre. Aquí es donde entra en juego el principio de redundancia. Si se ha hecho un trabajo adecuado de copia de seguridad del sistema, incluida información especial sobre su configuración de *hardware*, no se pierden datos, pero aún se podría perder tiempo valioso.

Los planes de recuperación pueden ser útiles siempre que no sean meramente ejercicios burocráticos. Por lo general, una lista de chequeo sería suficiente, siempre que el equipo de administración del sistema esté familiarizado con los detalles de la configuración local. Un error común en una organización grande, que se garantiza que generará fricciones, es hacer suposiciones injustificadas sobre un departamento local. **La delegación puede ser una estrategia valiosa en la lucha contra el tiempo.** Si hay suficientes administradores de sistemas locales que conocen los detalles de cada parte de la red, a esas personas les tomará menos tiempo tomar las decisiones apropiadas e implementar el plan de recuperación. Sin embargo, la delegación también abre a la posibilidad de inconsistencias: se debe asegurar que aquellos a quienes se delega la responsabilidad de los planes de recuperación estén bien capacitados.

Cuando se produce una pérdida, se deben recuperar archivos de las copias de seguridad. Una de las grandes ventajas de un esquema de duplicación de disco es que los usuarios pueden encontrar copias de seguridad de sus propios archivos sin tener que involucrar a un administrador. Para recuperaciones de archivos más grandes, es más eficiente que un administrador del sistema se encargue de la tarea. Restaurar desde una copia de seguridad en cinta es una tarea mucho más complicada. En primer lugar, se debe localizar la cinta (o cintas) correctas que contienen las versiones apropiadas de los archivos respaldados. Esto implica tener un sistema de almacenamiento, leer etiquetas y comprender cualquier secuencia del incremento que se utilizó para realizar el volcado. Es un asunto que consume mucho tiempo. Una de las incomodidades de las copias de seguridad incrementales es que hacer copias de seguridad de los archivos puede implicar cambiar varias cintas para reunir todos los archivos. Además, hay que tomar en cuenta lo que sucedería si las cintas no estuvieran debidamente etiquetadas o si se sobrescriben por accidente.

Prepararse para una recuperación es prepararse para el peor escenario: la destrucción del sitio. Se debe determinar la **cantidad de datos que se perderían** y **cuánto tiempo** tomaría para volver a poner en funcionamiento el sistema. Se debe incluir en los cálculos, por ejemplo, el tiempo que tomaría adquirir un nuevo *hardware*. Aquí aparecen los conceptos de **Tiempo Objetivo de Recuperación** (*Recovery Time Objective*, RTO) y un **Punto Objetivo de Recuperación** (*Recovery Point Objective*, RPO), que son parámetros a establecerse en el plan de recuperación de servidores o sistemas de archivos específicos. Cuando estos parámetros están disponibles, proporcionan una valiosa orientación.

El RTO representa la cantidad máxima de tiempo que la empresa puede tolerar esperar a que se complete una recuperación. Los RTO típicos para datos de usuario varían de horas a días. Para los servidores de producción, los RTO pueden variar de horas a segundos (Nemeth, Snyder, Hein, & Whaley, 2011).

Un RPO indica cómo de reciente se requiere que sea una copia de seguridad para la restauración, e influye en la granularidad en la que deben conservarse las copias de seguridad. Dependiendo de la frecuencia con la que cambie el conjunto de datos y de lo importante que sea, un RPO puede variar de semanas, a horas o segundos. Las copias de seguridad en cinta claramente no pueden satisfacer los RPO cuasi instantáneos, por lo que tales requisitos generalmente implican grandes inversiones y dispositivos de almacenamiento especializados ubicados en múltiples centros de datos (Nemeth, Snyder, Hein, & Whaley, 2011).

En la Figura 15 se muestra el proceso general de recuperación y reinicio de un sistema luego de una situación de fallo. Todas las modificaciones de los datos ocurridas desde el último respaldo (última imagen buena conocida) hasta el momento de ocurrencia del fallo se perderán, y debe estar dentro del rango de la máxima cantidad de datos que la organización o el dueño de los datos está dispuesto a perder, lo cual es justamente la definición del RPO. Una vez que el fallo ocurre, y luego de su detección, se activa un proceso de análisis para activar el plan de recuperación con la mejor estrategia posible, definida en el plan de recuperación. Para lograr le recuperación se debe ejecutar entonces el proceso de restauración de los datos (y otros componentes), a recuperarse desde las copias de seguridad (o componentes de respaldo, *spares*). La recuperación consistirá entonces en la reunión de los datos y componentes respaldados para finalmente reiniciar el sistema. El tiempo que transcurre desde que inicia el fallo hasta la recuperación y reinicio del sistema debe estar en los límites del RTO establecido en el plan de recuperación, que es el tiempo máximo a tolerar con el sistema fuera de funcionamiento. Obviamente, el medio de almacenamiento empleado para los respaldos va a incidir en el cumplimiento o no del RTO establecido en el plan, sabiendo que, por ejemplo, los tiempos de restauración desde unidades de discos serán considerablemente más cortos que los tiempos de restauración desde cintas. Todo esto habrá que colocarlo en la balanza de Costes vs. Requerimientos.

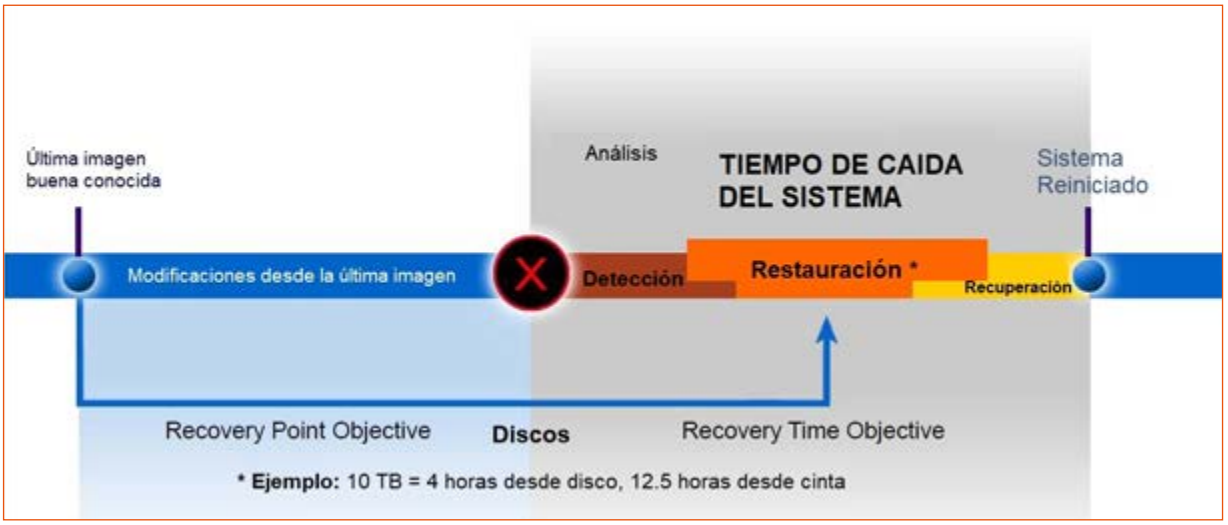

Figura 15. Proceso de recuperación y los parámetros involucrados. Elaboración propia.

Tema 5. Automatización y gestión de servicios

En el mundo de la administración de sistemas son parte esencial las tareas o rutinas de configuración, monitorización y mantenimiento. Muchas de estas tareas pueden ser repetitivas, por lo que la posibilidad de automatización siempre será bienvenida.

La automatización elimina el riesgo asociado con el error humano, como la configuración manual incorrecta. Eliminar esto puede disminuir el tiempo de inactividad y aumentar la confiabilidad. La automatización de la infraestructura puede aumentar la rapidez de las instalaciones y configuraciones, y tiene como objetivo proporcionar visibilidad para ayudar a otros equipos de toda la empresa a trabajar de manera rápida y eficiente. La necesidad de automatización se ha vuelto progresivamente más clara a medida que los sitios crecen y la complejidad de la administración aumenta.

La mayoría de las herramientas de administración de sistemas desarrolladas en la industria se basan en la idea de **interfaces de control** (interacción entre el administrador y la máquina para realizar cambios manuales) o en la **clonación de sistemas** de referencia existentes (duplicación). Se ven interfaces gráficas de usuario de complejidad creciente, pero rara vez se presta atención seria al comportamiento autónomo o automático (Burgess, 2004).

Han existido en la industria herramientas de administración de sistemas como Tivoli, de IBM que está basada en los estándares CORBA y X/Open. Tivoli es un producto comercial, anunciado como un sistema de gestión completo para ayudar tanto en la logística de la gestión de la red como en una

variedad de problemas de configuración. Como con la mayoría de las herramientas comerciales de administración de sistemas, aborda los problemas de administración desde el punto de vista de la comunidad empresarial. La fortaleza del *software* de gestión como Tivoli está en su enfoque integral de gestión. Se basa en comunicaciones cifradas e interrelaciones cliente-servidor para proporcionar funcionalidad, incluida la distribución de *software* y la ejecución de *scripts*. Tivoli puede activar *scripts*, pero los *scripts* en sí mismos son un enlace débil. No se proporcionan herramientas especiales, los programas son esencialmente *scripts de shell* con todos los problemas habituales. Entre las tareas que se pueden ejecutar en sistemas como Tivoli se encuentran:

- Ejecutar scripts de forma manual, cuando sea necesario.
- Programar tareas con una función similar a cron (Unix/Linux).
- Ejecutar una acción como respuesta a un evento, como ejecutar una tarea en un conjunto de *hosts*, o copiar paquetes.

HP OpenView es otro producto comercial basado en protocolos de control de red como SNMP (*Simple Network Management Protocol*). Openview tiene como objetivo proporcionar un sistema de gestión de configuración común para impresoras, dispositivos de red, sistemas Windows y HPUX. Desde una ubicación central, los datos de configuración pueden enviarse a través de la red de área local utilizando el protocolo SNMP. Una de sus ventajas es un enfoque consistente para la gestión de servicios de red.

Son muchos los protocolos que se han propuesto en la industria para implementar las funciones de gestión de redes y sistemas desde los diferentes *softwares* de administración, destacando los modelos de SNMP (a explicarse más adelante) para las funciones específicas de monitoreo y control de los dispositivos de red y otros de naturaleza estática (los dispositivos más debajo en la jerarquía de las operaciones, por lo que SNMP normalmente se referencia como una *South Bound Interface*, SBI), y los modelos de **CORBA** y **CFEngine** para la llamada **inteligencia de red**, donde los protocolos se usan administración avanzada con automatización y reportes a los sistemas de soporte a operaciones (*Operation Support System*, OSS) más dinámicos y complejos, y que estarían en la parte superior de la jerarquía de operaciones, por lo que estos protocolos normalmente pertenecen a las llamadas *North Bound Interfaces*, NBI.

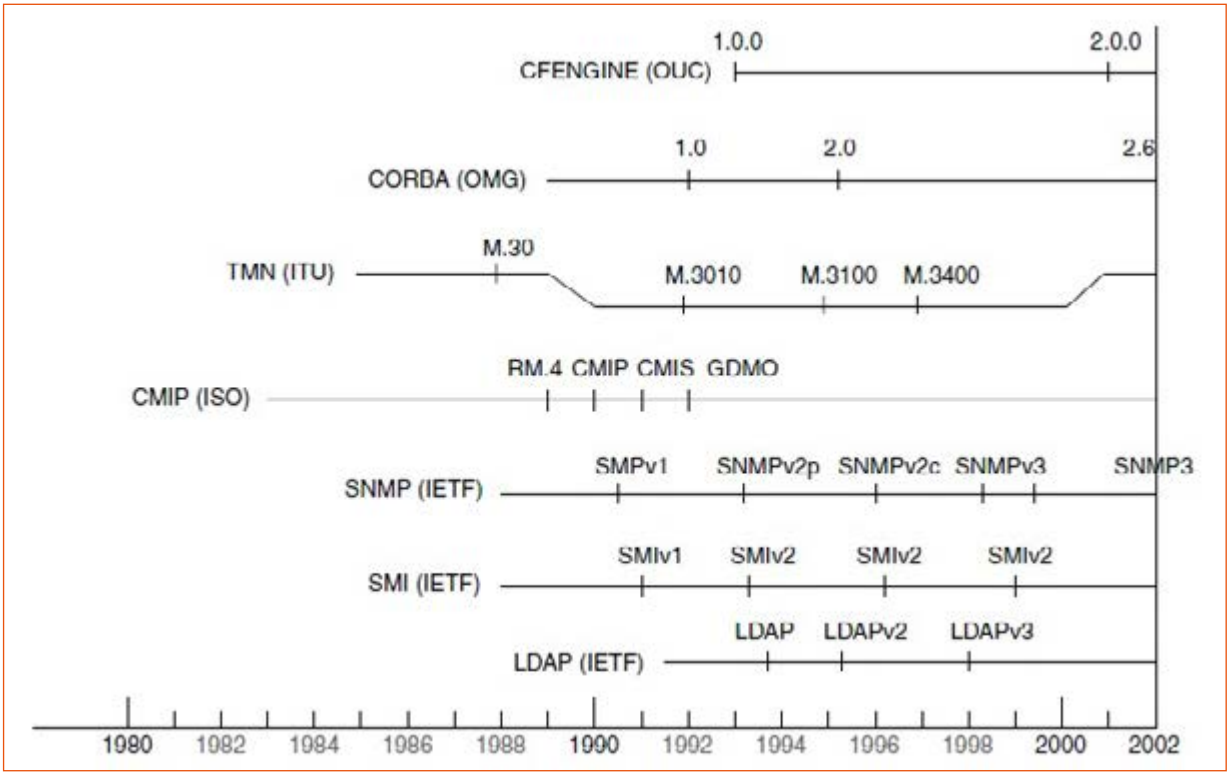

Figura 16. Evolución de algunas tecnologías de gestión de redes y sistemas. Recuperado de Burgess (2004).

5.1. Automatización y programación de tareas

La programación de tareas, tanto humana como automática, puede desempeñar un papel importante en el buen funcionamiento de un sistema humano-computadora. Si las reparaciones llegan demasiado tarde, se pueden causar problemas innecesarios. Si las reparaciones se programan con demasiada frecuencia, puede ser un desperdicio y menoscabar otras tareas. La capacidad de un sistema de administración para ejecutar tareas en momentos predeterminados es pieza fundamental para mantener el control sobre un sistema dinámico y cambiante, de forma eficiente.

5.1.1. Estrategias de programación de tareas

Un aspecto importante de la gestión de la configuración en administración de sistemas se refiere a cómo se programan las tareas de gestión, tanto en respuesta a eventos específicos como como una cuestión general del mantenimiento del sistema. La **administración basada en políticas**, que emplea agentes o robots de *software*, es un método administrativo que escala a un gran número de *hosts* en sistemas distribuidos, precisamente porque cada *host* es esencialmente responsable de su propio estado de configuración. Sin embargo, las interdependencias de tales máquinas en red significan que la gestión de la configuración debe reflejar las propiedades globales de la red, como las delegaciones y las funciones del servidor. Esto presenta desafíos especiales. Por lo tanto, es importante tener propiedades de programación predecibles.

Los **lenguajes de configuración basados en políticas** asocian la ocurrencia de eventos o condiciones específicos, con las respuestas que debe realizar un agente. `Cfengine` es una herramienta que logra

esto en gran medida, clasificando el estado de un host, en el momento de la invocación, en varias cadenas de identificadores. Algunos de estos representan el momento de la invocación, otros la naturaleza del ambiente, etc.

La programación toma muchas formas, como la programación bajo demanda, la programación de producción, la programación multiprocesador, etc. Puede tener lugar en cualquier medida de tiempo, espacio u otra etiqueta de cobertura adecuada.

Las dos clases principales de programación de tareas son la **programación dinámica** y la **programación estática**. La programación dinámica puede cambiar su propio patrón de ejecución, mientras que la estática está completamente predeterminada. En general, resolver problemas de programación estática se considera de dificultad "no polinómica" (es decir, no hay un algoritmo conocido que pueda hacer esto en tiempo polinómico). Esto implica asignar los vértices (tareas) de un gráfico acíclico dirigido a un conjunto de recursos, de modo que se minimice el tiempo total para procesar todas las tareas. El tiempo total para procesar todas las tareas generalmente se conoce como el *makepan* (Burgess, 2004).

A menudo, un objetivo adicional es lograr un *makespan* corto a la vez que se minimiza el uso de los recursos. Tales problemas de optimización de objetivos múltiples implican compromisos y compensaciones complejas, y las buenas estrategias de programación se basan en una comprensión detallada y profunda del dominio del problema específico. La mayoría de los enfoques pertenecen a la familia de algoritmos de programación de **listas de prioridades**, diferenciados por la forma en que las prioridades de las tareas se asignan al conjunto de recursos. Tradicionalmente, se han empleado métodos heurísticos en la búsqueda de soluciones de alta calidad. Durante los últimos años, la heurística se ha combinado con técnicas de búsqueda modernas, como el algoritmo del recocido simulado, y los algoritmos genéticos (Centeno & Velásquez, 2016).

5.1.2. El servicio `cron` (Unix/Linux)

Los sistemas Unix/Linux tiene un demonio de tiempo llamado `cron` (cronómetro). Éste lee un archivo de configuración llamado `crontab` que contiene una lista de comandos de *shell* para ejecutar a intervalos de tiempo regulares. En los sistemas modernos basados en Unix/Linux, cada usuario puede crear y editar un archivo *crontab* usando el comando

```
crontab -e
```

Este comando inicia un editor de texto que permite editar el archivo. El contenido del archivo `crontab` de un usuario se puede enumerar en cualquier momento con el comando `crontab -l`. El formato de un archivo crontab es una serie de líneas de la forma:

```
Minutes  hours  day      month  weekday         Shellcommand
0-59      0-23   1-31  1-12   Mon-Sun
```

Se puede usar un asterisco o una estrella * como comodín, indicando "cualquiera". Por ejemplo:

```
# Run script every weekday morning Mon-Fri at 3:15 am:
15 3 * * Mon-Fri /usr/local/bin/script
```

Un típico archivo *crontab* para el usuario ***root*** puede ser el siguiente:

```
#
# The root crontab
#
0 2 * * 0,4 /etc/cron.d/logchecker
5 4 * * 6 /usr/lib/newsyslog
0 0 * * * /usr/local/bin/cfwrap /usr/local/bin/cfdaily
30 * * * * /usr/local/bin/cfwrap /usr/local/bin/cfhourly
```

La primera línea se ejecuta a las 2:00 a.m. los domingos y miércoles, la segunda a las 4:05 los sábados; la tercera se ejecuta todas las noches a las 00:00 horas y la línea final se ejecuta una por hora en cada media hora.

5.1.3. Programación de tareas con `schtasks` (Microsoft Windows)

El servicio de programación de tareas de Windows es similar a `cron`. De forma predeterminada, solo el administrador tiene acceso al servicio de programación. Todos los trabajos iniciados por el servicio de programación se ejecutan con los mismos derechos del usuario que ejecuta el servicio (normalmente el administrador).

El controlador de dominio coordina el servicio de programación de tareas para todos los *hosts* en un dominio, por lo que el **nombre de *host*** en el que se ejecutará un trabajo por lotes puede ser un argumento para el comando de programación.

Si bien existe un asistente de Microsoft Windows que ayuda a crear gráficamente tareas programadas, el comando `schtasks` es ideal para situaciones como las siguientes:

- Manipular tareas en scripts por lotes.
- Controlar y cree tareas en máquinas en red sin tener que iniciar sesión en ellas.
- Tareas de creación/sincronización en lotes en múltiples máquinas.
- Usar en aplicaciones personalizadas para comunicarse con el Programador de Tareas en lugar de tener que hacer llamadas a través de la API.

En los siguientes ejemplos de programación de tareas con `schtasks` se muestra cómo crear y luego modificar una tarea de ejecución diaria (How To Geek, s.f.):

Ejecutar el archivo por lotes "C:\RunMe.bat" todos los días a las 9:00 AM. El nombre asignado a la tarea es "My Task":

```
schtasks /create /SC DAILY /TN "My Task" /TR "C:\RunMe.bat" /ST 09:00
```

Modificar la tarea "My Task" para ejecutarse ahora a las 14:00 h, todos los días:

```
schtasks /Change /TN "My Task" /ST 14:00
```

Crear la tarea "My Task" para ejecutar "C:\RunMe.bat"el primer día de cada mes:

```
schtasks /Create /SC MONTHLY /D 1 /TN "My Task" /TR "C:RunMe.bat" /ST 14:00
```

Crear la tarea "My Task" para ejecutar "C:\RunMe.bat" el primer día de cada mes:

```
schtasks /Create /SC WEEKLY /D MON,TUE,WED,THU,FRI /TN "My Task" /TR "C:RunMe.
bat" /ST 14:00
```

Crear la tarea "My Task" para ejecutar "C:\RunMe.bat" cada día de semana a las 2:00 PM:

```
schtasks /Create /SC WEEKLY /D MON,TUE,WED,THU,FRI /TN "My Task" /TR "C:RunMe.
bat" /ST 14:00
```

Eliminar la tarea "My Task":

```
schtasks /Delete /TN "My Task"
```

Como es de suponer, estos comandos (`cron` y `schtasks`) pueden llegar a ser bastante complejos y ser muy útiles para la funcionalidad de programación de tareas. Para ver todas sus opciones y parámetros, se puede acceder a sus manuales o visualizar las ayudas interactivas en la línea de comandos.

5.2. Monitorización del sistema

Las herramientas de monitorización por lo general funcionan haciendo que un proceso demonio (*daemon*) recopile información de auditoría básica, establezca un límite en un parámetro dado y active una alarma si el valor excede los parámetros aceptables. Las alarmas pueden enviarse por correo, pueden enrutarse a una pantalla GUI o incluso pueden enrutarse a un localizador de administrador del sistema.

5.2.1. Gestión de redes con SNMP

El **Protocolo Simple de Administración de Redes** (*Simple Network Management Protocol*, SNMP) es un protocolo diseñado para hacer precisamente lo que dice su nombre. Se generó en 1987 como un protocolo de monitoreo de puerta de enlace simple, pero se extendió rápidamente y se convirtió en un estándar para el monitoreo de redes, principalmente basadas en la pila de protocolos de Internet, TCP/IP. SNMP fue diseñado para ser lo suficientemente pequeño y simple como para poder funcionar incluso con piezas menores de tecnología de red como puentes e impresoras.

El modelo de gestión de configuración es particularmente adecuado para dispositivos no interactivos como impresoras e infraestructura de red estática que requieren una configuración esencialmente estática durante largos períodos de tiempo. Desde entonces, SNMP se ha extendido, con menos éxito, a algunos aspectos de la administración del *host*, como las estaciones de trabajo y la configuración del servidor de red de la PC; sin embargo, el modelo estático de configuración es menos apropiado aquí, ya que los usuarios perturban constantemente los servidores de manera impredecible y, combinado con el registro de seguridad poco impresionante de las primeras versiones, esto ha desalentado su uso para la administración del *host* (Burgess, 2004).

La arquitectura tradicional de SNMP se basa en dos entidades: gestores (*managers*) y agentes (*agents*). Los gestores SNMP ejecutan aplicaciones de administración, mientras que los agentes SNMP median el acceso a las variables de administración. Estas variables contienen valores simples y se organizan en grupos de escalares de variables de un solo valor o en tablas conceptuales de variables de varios valores. El conjunto de todas las variables o parámetros en un sistema gestionado se denomina MIB (*Management Information Base*).

SNMP ha sido criticado a menudo por la débil seguridad de sus agentes, que están configurados de forma predeterminada con una contraseña (o nombre de comunidad) de texto claro llamada *public* de forma predeterminada. La versión 3 del protocolo SNMP finalmente se acordó y se publicó en diciembre de 2002 para abordar estos problemas, utilizando métodos de cifrado robustos.

SNMP admite tres operaciones en dispositivos: **leer** (*read*), **escribir** (*write*) y **notificar** (*traps*). La consola de administración puede leer y modificar las variables almacenadas en un dispositivo y emitir notificaciones de eventos especiales. El acceso SNMP está mediado por un proceso servidor (*Daemon*) en cada nodo de *hardware* (*agent SNMP*), que normalmente se comunica por UDP en los puertos 161 y 162. Los sistemas operativos modernos a menudo ejecutan demonios o servicios SNMP que anuncian su estado a un administrador con capacidad SNMP. Los servicios están protegidos por una contraseña bastante débil que se llama la **cadena de la comunidad**, mencionada anteriormente. Debido a que SNMP es básicamente un protocolo de "consultas" para valores simples, su éxito depende de manera crucial de la capacidad de las bases de información de administración o MIB para caracterizar correctamente el estado de los dispositivos y cómo los agentes traducen los valores de MIB en acciones reales. Para monitorear la carga de trabajo (por ejemplo, estadísticas de carga en las interfaces de red o alertas de agotamiento de papel en una impresora), este modelo es claramente bastante bueno. De hecho, incluso las herramientas alojadas en *host* Unix/Linux (`ps`, `top`, `netstat`, etc.) utilizan este enfoque para consultar tablas de recursos en sistemas más complejos. Además, en el caso de los dispositivos de red tontos, cuyo comportamiento está esencialmente fijado por algunos parámetros o listas de ellos (impresoras, conmutadores, etc.), este modelo incluso cumple con el desafío de la configuración razonablemente bien.

Las variables que existen en una MIB se definen formalmente en los llamados módulos MIB (RFC 1213) que están escritos en un lenguaje llamado SMI (*Structure of Management Information*). El SMI proporciona un espacio de nombres genérico y extensible para identificar variables MIB. Debido a la falta de recursos de estructuración de datos de nivel superior, los módulos MIB a menudo aparecen como un mosaico de definiciones de variables individuales en lugar de una estructura de clases como en Java u otros lenguajes orientados a objetos. Una solicitud SNMP especifica la información que desea leer/escribir dando el nombre de una instancia de la variable para leer o escribir en una solicitud. Este nombre se asigna en el módulo MIB formal. Existen módulos MIB estándar para tablas de traducción de direcciones, estadísticas TCP/IP, etc. Estos tienen parámetros predeterminados que pueden ser alterados por SNMP: nombre del sistema, ubicación y contacto humano; estado de la interfaz (arriba/abajo), *hardware* y dirección IP, estado IP (puerta de enlace de reenvío) IP TTL, dirección IP siguiente salto, edad y máscara de ruta IP, estado TCP, estado del vecino, habilitación de *traps* SNMP, etc. Las MIB entonces están alojadas en los **dispositivos administrados** donde funcionan los **agentes SNMP**, mientras que los **gestores SNMP** se alojan normalmente en las **estaciones de monitoreo** desde donde se realizan las consultas a los agentes. La Figura 17 resume está arquitectura con sus componentes básicos.

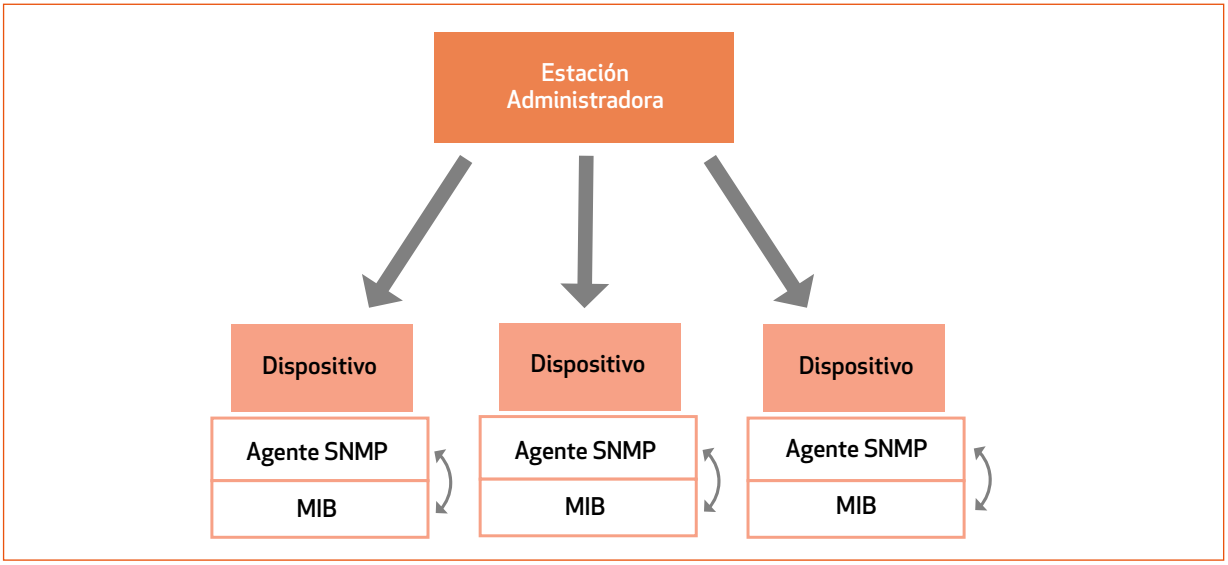

Figura 17. Componentes básicos de la arquitectura de SNMP. Elaboración propia.

Aunque SNMP funciona bastante bien para monitorear grupos de dispositivos estáticos, su éxito en la administración de *hosts*, incluidas estaciones de trabajo y servidores, es más cuestionable. Incluso sin tener en cuenta los problemas de escalabilidad que puedan darse, **el modelo MIB es muy difícil de adaptar a los *hosts* donde los usuarios interactúan constantemente con el sistema**. Los *hosts* ya no son dispositivos básicamente predecibles con un estado aproximadamente constante; su estado cambia dinámicamente en respuesta a la interacción del usuario y en respuesta a los servicios que realizan. La filosofía de gestión de SNMP, es decir, comunicarse con un agente para leer y escribir, solo es efectiva cuando la tasa de cambio de estado de un dispositivo es lenta, es decir, cuando el dispositivo cambia aproximadamente a la misma tasa que la tasa de cambio de política sí mismo. Además, la complejidad de las operaciones que puede llevar a cabo un agente SNMP tonto, basado solo en las instrucciones del "botón pulsador" de MIB y una comunicación bilateral con una interfaz de administración, es limitada. Por lo tanto, **en la práctica, SNMP solo se puede utilizar para detectar problemas, no para repararlos**. Esto no es tanto un problema en el modelo, sino en la forma en que se implementa. Si los agentes SNMP contuvieran inteligencia integrada, el modelo de comunicación podría conservarse. Se está entonces en el territorio de sistemas de agentes más inteligentes como **cfengine** y **PIKT**.

A pesar de las deficiencias de SNMP para las operaciones de *host*, muchos sistemas operativos definen sus propias MIB para la recopilación de datos de rendimiento del sistema e incluso para parámetros de configuración esenciales. Algunos sistemas comerciales de gestión de redes, como *OpenView* de Hewlett Packard, funcionan leyendo y escribiendo sobre la MIB utilizando la tecnología SNMP cliente-servidor. La mayoría de las variantes de Unix/Linux, Novell y Windows ahora también admiten SNMP. Sus MIB se pueden usar incluso para recopilar información, como los nombres de los usuarios que han iniciado sesión. Esta información no es particularmente relevante para el problema de la gestión de recursos e incluso puede considerarse un riesgo de seguridad, de no tomarse las medidas adecuadas.

SNMP ha ganado con los años mucha popularidad para monitorear *hardware* de red (enrutadores y conmutadores, etc.), pero como cualquier base de datos de información pública, también puede ser abusado por los atacantes de red, y es un objetivo principal para la vulneración de muchos sitios. Para

protegerse frente a esta situación se suelen deshabilitar los servicios SNMP por completo en los *hosts*, usándolo solo para monitorear el *hardware* de transporte de red, o creando reglas en los *firewalls* para rechazar mensajes SNMP provenientes de redes externas.

A pesar de sus limitaciones, SNMP sigue siendo el protocolo de elección para la administración de la mayoría del *hardware* de red, y se han desarrollado muchas herramientas para consultar y administrar dispositivos habilitados para SNMP.

5.2.2. Sistemas de *logging* y auditorías

El núcleo del sistema operativo comparte recursos y ofrece servicios. Se le puede pedir que mantenga listas de transacciones que se han llevado a cabo para que luego se puedan revisar y ver exactamente qué sucedió en un momento dado. Esto se llama *logging* (registro) o auditoría. La auditoría se ha convertido en un problema relacionado con la seguridad. Las organizaciones tienen miedo de los robos de los *hackers* del sistema y quieren poder rastrear las actividades del sistema para poder mirar hacia atrás y descubrir la identidad de un *hacker*.

Si bien el registro continuo de actividades en el sistema puede consumir muchos recursos, para algunas organizaciones la auditoría es importante. Un uso para la auditoría es el llamado "no repudio" o "no negación". Si se registra todo en un sistema, los usuarios no pueden retroceder y afirmar que no hicieron algo: todo está allí en el registro. El no repudio es una característica de seguridad que alienta a los usuarios a ser responsables de sus acciones.

Syslog es un estándar para el registro de datos de la computadora. Separa el *software* que **genera mensajes del sistema** que los almacena y el ***software* que los informa y analiza**. *Syslog* se puede utilizar para la gestión de sistemas informáticos y la auditoría de seguridad, así como para mensajes informativos, de análisis y de depuración generalizados. Es compatible con una amplia variedad de dispositivos (como impresoras y enrutadores) y receptores en múltiples plataformas. Debido a esto, *Syslog* se puede utilizar para integrar datos de registro de diferentes tipos de sistemas en un repositorio central.

Los mensajes se refieren a una instalación (`auth, authpriv, daemon, cron, ftp, lpr, kern, mail, news, syslog, user, uucp, local0, ..., local7`) y se les asigna una gravedad (`Emergency, Alert, Critical, Error, Warning, Notice, Info or Debug`) por parte del remitente del mensaje.

Tema 6. Soporte al usuario y formación

Todos los usuarios requieren ayuda en algún momento. El hecho de que los usuarios normales no sean usuarios privilegiados significa que ocasionalmente deben confiar en un "superusuario" o administrador para solucionar un problema que esté fuera de su control. Si se quiere distinguir entre usuarios privilegiados y no privilegiados, es menester proveer a los usuarios el servicio de soporte o *help-desk* (Burgess, 2004).

La cantidad de soporte que se ofrece a los usuarios es una cuestión de políticas. Se tiene la opción entre apoyar a los usuarios directamente o invertir tiempo en formarlos y hacerlos autosuficientes. ¿Cuál de estas dos estrategias es más conveniente? Depende de la naturaleza del problema. En casi todos los casos se necesitan ambas estrategias. Por lo tanto, se busca una mezcla de lo siguiente:

- Capacitación de usuarios.
- Ayuda a los usuarios.
- Documentar y proporcionar las respuestas a las preguntas frecuentes.

La proporción del tiempo dedicado a cada uno debe elegirse como política. El tiempo de los administradores de sistemas generalmente es escaso, aunque una mayor automatización está aportando tiempo para la concentración en problemas de nivel superior, como el soporte. La capacidad de soporte en un sistema depende de su tamaño en relación con los recursos de personal

disponible. Dar soporte al *hardware* y el *software* significa corregir errores, actualizar y tal vez proporcionar capacitación o servicios de asistencia telefónica o remota. Algunos sistemas de *help-desk* por correo electrónico pueden ayudar en la organización de los servicios de soporte, pero son principalmente herramientas de seguimiento de tareas. A veces, los *hosts* y los paquetes de *software* se etiquetan como no compatibles para enfatizar a los usuarios que no tendrán soporte si insisten en usar esas instalaciones.

La provisión de un servicio a los usuarios sugiere la necesidad de controles de calidad. Esto se remonta al principio central de previsibilidad en la gestión: **un resultado controlado y verificado le permite a uno confiar en la efectividad del procedimiento y, por lo tanto, en la estabilidad a largo plazo del sistema humano-computadora**. Las listas de chequeo son una ayuda algorítmica útil para asegurar resultados predecibles. Una **lista de verificación básica para servicios de soporte al usuario** puede consistir en los siguientes puntos (Burgess, 2004):

- ¿Se ha leído la solicitud del usuario correctamente?
- ¿Se entiende la solicitud?
- ¿La solicitud está en línea con las políticas de soporte de la organización?
- ¿Se tiene la competencia para atender la solicitud?
- Agendar la solicitud. La respuesta rápida mitiga la frustración de los usuarios.

Por otro lado, otros autores han sugerido modelos para la estandarización del soporte al usuario, como el **modelo de 9 pasos** sugerido por (Limoncelli T. , 1999) y que se resume en la Tabla 13. En dicho modelo se resalta el aspecto de la comunicación con los usuarios, con detalles como el saludo inicial, la información durante proceso, y la involucración en las verificaciones. Esto redundará con toda seguridad en la calidad percibida por los usuarios en el proceso de soporte.

Tabla 13
Modelo de 9 pasos para la asistencia al usuario

Modelo de 9 pasos para soporte al usuario
SALUDO
1. Saludo. ¿Cómo puedo ayudarle?
IDENTIFICACIÓN DEL PROBLEMA
2. Identificar el problema 3. Refinar y expresar el problema 4. Verificar el problema
CORRECCIÓN
5. Listar soluciones propuestas 6. Seleccionar la solución 7. Ejecución de la solución

Modelo de 9 pasos para soporte al usuario
VERIFICACIÓN
8. Auto chequeo
9. Chequeo del usuario

Nota. Adaptado de Limoncelli (1999)

6.1. Marcos de trabajo para soporte al usuario

Los procesos de soporte al usuario han evolucionado hacia **marcos de trabajo** (*frameworks*) reconocidos por la industria. Muchas organizaciones que realmente desean controlar y formalizar este aspecto en la administración de sus sistemas, tratan de implementar dichos marcos de trabajo.

El marco de trabajo más importante de la industria es el ITIL (*IT Infrastructure Library*). ITIL es un marco de mejores prácticas para la prestación de servicios de IT. El enfoque sistemático de ITIL para la gestión de servicios de IT puede ayudar a las empresas a gestionar los riesgos, fortalecer las relaciones con los clientes, establecer prácticas rentables y crear un entorno de IT estable que permita el crecimiento, la escalabilidad y la gestión del cambio.

ITIL fue desarrollado por la Agencia Central de Telecomunicaciones y Computación del gobierno británico (*Central Computer and Telecommunications Agency*, CCTA) durante la década de 1980. La primera versión de ITIL consistió en más de 30 libros, desarrollados y lanzados a lo largo del tiempo, que codificaban las mejores prácticas en tecnología de la información acumuladas de muchas fuentes (incluidas las mejores prácticas de los proveedores) alrededor del mundo. IBM, por ejemplo, dice que su serie de cuatro volúmenes sobre conceptos de gestión de sistemas, *Un Sistema de Gestión para Sistemas de Información*, conocido como los Libros Amarillos (*The Yellow Books*), proporcionó información vital en los libros originales de ITIL. En abril de 2001, la CCTA, junto con varias otras agencias, ingresaron a la Oficina de Comercio Gubernamental (*Office of Government Commerce*, OGC), que ahora se conoce como la Oficina del Gabinete. El OGC adoptó el proyecto como parte de su misión de trabajar con el sector público del Reino Unido como catalizador para lograr la eficiencia, la relación calidad-precio en actividades comerciales y un mayor éxito en la entrega de programas y proyectos. El objetivo no era crear un producto patentado que pudiera comercializar; más bien, fue para **reunir las mejores prácticas** que podrían ayudar con lo que el gobierno reconoció que era una dependencia creciente dentro del gobierno de IT combinada con una dolorosa falta de procedimientos estándares que aumentaban los costos y permitían que los errores se perpetuaran. Rápidamente se hizo evidente que la distribución de estas mejores prácticas beneficiaría tanto a las organizaciones del sector público como al privado. Con los años, se reconoció la credibilidad y utilidad de ITIL, y en 2005 sus prácticas contribuyeron y se alinearon con el estándar de Gestión de Servicios ISO/IEC 20000, el primer estándar internacional para la gestión de servicios de IT, el cual se basa en el estándar británico BS15000. Desde 2013, ITIL es propiedad de **Axelos**, una empresa conjunta entre la Oficina del Gabinete y Capita. Axelos otorga a las empresas la licencia para usar el marco ITIL, al tiempo que administra las actualizaciones y los cambios en el proceso. Sin embargo, para usar ITIL internamente, las organizaciones no necesitan una licencia. **ITIL v3** se lanzó en 2011, bajo la Oficina del Gabinete,

con actualizaciones de la versión de 2007 publicada bajo la OGC. En 2018, Axelos anunció **ITIL-4**: una revisión importante de todo el marco y el mayor cambio desde que se publicó ITIL v3 en 2007. ITIL 4, que comenzó a implementarse en el primer trimestre de 2019, ofrece una versión más ágil, flexible y personalizable de ITIL que se actualiza para empresas modernas. La última versión fomenta menos silos, más colaboración, comunicación en todo el negocio e integración ágil y DevOps en las estrategias de ITSM (*IT Service Management*) (White & Greiner, 2019).

ITIL-4, que se lanzó en 2019, mantiene el mismo enfoque en la automatización de procesos, mejorando la gestión del servicio e integrando el departamento de IT en el negocio. Sin embargo, también actualiza el marco para adaptarse a la nuevas tecnologías, herramientas y *software*. Desde la última actualización de ITIL, el departamento de IT ha pasado a ser parte integral de cada negocio y el nuevo marco se adapta a esto al ser más ágil, flexible y colaborativo.

ITIL-4 contiene **nueve principios rectores**, que cubren la **gestión del cambio organizacional**, la **comunicación**, la **medición** y las **métricas**. Estos principios incluyen (White & Greiner, 2019):

- Centrarse en el valor.
- Diseñar por experiencia.
- Comenzar donde se está.
- Trabajar holísticamente.
- Progresar iterativamente.
- Observar directamente.
- Ser transparente.
- Colaborar.
- Mantener la simplicidad.

Si bien ITIL no es el único marco de trabajo para soporte al usuario de tecnología en las organizaciones, sí es el más popular, y el que se está convirtiendo en un estándar de facto en este campo. El gráfico de la Figura 18 se muestra una evidencia de ello, en comparación con otros marcos de trabajos, de los que se destacan KCS (*Knowledge-Centered Support/Service*), MOF (*Microsoft Operations Framework*), ISO 9000 y COBIT, donde ITIL y MOF están asociados más directamente a procesos asociados con los servicios de tecnologías de información, mientras los demás tratan de procesos de negocio en general, incluyendo los procesos de los servicios de tecnología.

Figura 18. Marcos de trabajo para soporte técnico. Recuperado de https://www.joetheitguy.com/

En resumen, **la administración de servicios de IT basada en ITIL ayuda a las organizaciones de cualquier tamaño en todo el mundo a maximizar y optimizar el valor comercial mediante la tecnología de la información**. Las mejores prácticas de ITIL cubren **la gestión** de las siguientes áreas de ITSM:

- Incidentes
- Problemas
- Cambios
- Nivel de servicio
- Continuidad
- Configuración
- Liberación o publicación
- Capacidad
- Financiera
- Disponibilidad
- Seguridad
- Soporte técnico
- Conocimiento

Para los objetivos de este material, habría que enfocarse en aspectos de ITIL como la **gestión de incidentes** y el **soporte técnico**. El objetivo principal de la gestión de incidentes es resolver la disrupción de un servicio cuanto antes para continuar con las operaciones. Debido al hecho de que una interrupción menor puede tener un gran impacto en una organización, los incidentes deben ser solucionados inmediatamente. Le gestión de incidentes usualmente involucra soporte de nivel uno, el cual incluye:

- Identificación del incidente.
- Registro del incidente.
- Categorización del incidente.
- Priorización del incidente.
- Diagnóstico inicial.
- Paso a nivel dos (si es necesario).
- Resolución del incidente.
- Cierre del incidente.
- Comunicación con el usuario durante el tiempo del incidente.

Las plataformas para soporte y documentación basadas en ITIL deberán tener entonces las funcionalidades que permitan seguir este ciclo para gestión de incidentes, esquematizado en la Figura 19.

Figura 19. Proceso de gestión de incidentes propuesto por ITIL-v3. Recuperado de Maldonado (2017).

6.2. Plataformas de soporte y documentación

Las mejores prácticas de administración de servicios de ITIL cubren muchas áreas de la administración de servicios de IT (ITSM), incluida la **administración de incidentes**, la **administración de configuración** y la **administración del soporte técnico** (*helpdesk*). Una plataforma de soporte compatible con ITIL flexible y asequible es clave para abordar estos objetivos.

La mejor plataforma de soporte no solo está etiquetada como "certificada por ITIL", sino que sus funcionalidades principales están diseñadas para ser flexibles y ajustables para adaptarse a los procesos de ITSM de acuerdo con los estándares de ITIL.

En la Tabla 14 se listan algunas de las plataformas de ITSM compatibles con ITIL más populares, describiendo algunas de sus características principales.

Tabla 14
Listado de plataformas ITSM

Plataforma ITSM	Descripción
Atlassian Jira Service Desk	Jira Service Desk ayuda a recibir, rastrear, administrar y resolver solicitudes de clientes en todos los departamentos. Las solicitudes se pueden enviar por correo electrónico, un centro de ayuda que se puede personalizar para las necesidades del negocio o mediante un widget incorporado. El *software* viene preparado para manejar la automatización, los SLA, los informes en tiempo real y los procesos certificados por ITIL, como la gestión de incidentes, problemas y cambios.
BMC Remedyforce	BMC Remedyforce ofrece gestión de incidentes, problemas, cambios y nivel de servicio, y también incluye autoservicio, descubrimiento y gestión de activos. Puede configurar y personalizar el sistema para entregar informes y administrar problemas de IT específicos. Las características incluyen *chatbots* que se integran con aplicaciones como Skype, Slack y aplicaciones móviles, informes de tablero, la capacidad de configurar la plataforma con codificación o scripts. Se integra con la famosa plataforma de CRM **Salesforce.com**.
SolarWinds Service Desk	SolarWinds adquirió Samanage Service Desk el año pasado, reemplazando el servicio ITSM anterior de SolarWinds, **Web Help Desk**. SolarWinds Service Desk ofrece todas las funciones estándar de manejo de tickets, automatización, informes y administración. También incluye herramientas para la gestión del nivel de servicio, un portal de autoservicio para usuarios, funciones de detección de riesgos y *software* CMDB. También ofrece integración con varias herramientas como Active Directory, Dropbox, Google Apps, Jira, Slack, Zendesk y más.
Zendesk	Zendesk es un sistema de tickets de IT integral y fácil de usar que es fácil de implementar y mantener. La *suite* incluye soporte, guía, chat y Talk: se puede comprar la *suite* completa o comprar cada servicio individualmente. El soporte ayuda a rastrear y resolver las necesidades de soporte del cliente. La guía ayuda a entregar respuestas de autoservicio a los clientes. El chat le permite interactuar en tiempo real con los clientes. Talk ayuda a brindar asistencia telefónica o por otro medio de conexión con los clientes.

Nota: Adaptado de White, S. (2020)

Glosario

API (*Application Programming Interface*)

Una API es un conjunto de funciones y procedimientos que permiten la creación de aplicaciones que acceden a las características o datos de un sistema operativo, aplicación u otro servicio.

CIFS (*Common Internet File System*)

El protocolo CIFS es un dialecto del protocolo SMB (*Server Message Block*). Accede a la utilidad de la red de Windows y recursos compartidos para permitir el intercambio de archivos de protocolo SMB que soporten CIFS.

CMOS (*Complementary Metal Oxide Semiconductor*)

El CMOS es una parte física de la placa base de una computadora: es un chip de memoria que alberga las configuraciones y funciona con la batería incorporada. El CMOS se reinicia y pierde todas las configuraciones personalizadas en caso de que la batería se quede sin energía. Además, el reloj del sistema se reinicia cuando el CMOS pierde energía.

CPU (*Central Processing Unit*)

La unidad central de procesamiento (CPU) de una computadora es una pieza de *hardware* que lleva a cabo las instrucciones de un programa de computadora. Realiza las operaciones aritméticas, lógicas y de entrada/salidas básicas de un sistema informático. La CPU a veces también se conoce como la unidad central de procesador, o procesador para abreviar.

DevOps (*Development and Operations*)

DevOps (acrónimo inglés *Development* -desarrollo- y *Operations* -operaciones-) es una práctica de ingeniería de *software* que tiene como objetivo unificar el desarrollo de *software* (*Dev*) y la operación del *software* (*Ops*). La principal característica del movimiento DevOps es defender enérgicamente la automatización y el monitoreo en todos los pasos de la construcción del *software*, desde la integración, las pruebas, la liberación hasta la implementación y la administración de la infraestructura. DevOps apunta a ciclos de desarrollo más cortos, mayor frecuencia de implementación, lanzamientos más confiables, en estrecha alineación con los objetivos comerciales.

DFS (*Distributed File System*)

El Sistema de archivos distribuido (*Distributed File System* DFS) proporciona funciones con la capacidad de agrupar lógicamente los recursos compartidos en varios servidores y vincular los recursos

compartidos de forma transparente en un único espacio de nombres jerárquico. DFS organiza los recursos compartidos en una red en una estructura similar.

Discos ATA (*Advanced Technology Attachment*)

ATA es un tipo de unidad de disco que integra el controlador de la unidad directamente en la unidad misma. Las computadoras pueden usar discos duros ATA sin un controlador específico para soportar el disco.

Discos SCSI (*Small Computer System Interface*)

SCSI es un conjunto de estándares de interfaz paralelos desarrollados por el *American National Standards Institute* (ANSI) para conectar impresoras, unidades de disco, escáneres y otros periféricos a las computadoras. SCSI (pronunciado "skuzzy") es compatible con todos los principales sistemas operativos.

FireWire (*IEEE 1394*)

IEEE 1394 (*Firewire*) es un tipo de conexión para diversas plataformas, destinado a la entrada y salida de datos en serie a gran velocidad. Suele utilizarse para la interconexión de dispositivos digitales como cámaras digitales y videocámaras a computadoras. Existen cuatro versiones de 4, 6, 9 y 12 pines. En el mercado doméstico su popularidad ha disminuido entre los fabricantes de hardware, y se ha sustituido por la interfaz USB en sus versiones 2.0 y 3.0.

Firmware

El *firmware* es un programa de *software* o un conjunto de instrucciones programadas para trabajar de forma específica en un dispositivo de *hardware*. Proporciona las instrucciones necesarias sobre cómo se comunica el dispositivo con el otro *hardware* de la computadora. El *firmware* generalmente se almacena en la ROM *flash* de un dispositivo de *hardware*.

FTP (*File Transfer protocol*)

FTP es un protocolo de red para la transferencia de archivos entre sistemas conectados a una red TCP/IP, basado en la arquitectura cliente-servidor. Desde un equipo cliente se puede conectar a un servidor para descargar archivos desde él o para enviarle archivos, independientemente del sistema operativo utilizado en cada equipo.

GECOS

GECOS en Linux también se conoce como “campo de comentarios para el usuario”. El comentario o el campo GECOS se utiliza con fines informativos en el registro del usuario en el archivo /etc/passwd, para indicar información adicional del usuario como el nombre completo, número de oficina, número de extensión telefónica o el número de teléfono de casa. GECOS viene del inglés " *General Electric Comprehensive Operating System*". GECOS pasó a llamarse GCOS cuando la división de grandes sistemas de **General Electric** fue vendida a la empresa **Honeywell**.

GNU/Linux

GNU es un acrónimo recursivo de *GNU's Not Unix!*, elegido porque el diseño de GNU es similar a Unix, pero difiere de éste por ser *software* libre y no contener código Unix. El proyecto GNU incluye un núcleo del sistema operativo, GNU Hurd, que fue el enfoque original de la *Free Software Foundation* (FSF).

GUI (*Graphical User Interface*)

La interfaz gráfica de usuario (GUI) es una forma de interfaz de usuario que permite a los usuarios interactuar con dispositivos electrónicos a través de iconos gráficos e indicadores de audio, como la notación primaria, en lugar del usuario basado en texto, interfaces, etiquetas de comandos escritos o navegación de texto.

Hot-Plugging

Funcionalidad por la cual los componentes de la computadora se pueden reemplazar sin tener que apagar el sistema.

LDAP (*Lightweight Directory Access Protocol*)

LDAP significa **Protocolo Ligero de Acceso a Directorios**. Como su nombre indica, es un protocolo liviano cliente-servidor para acceder a los servicios de directorio, específicamente a los servicios de directorio basados en X.500. LDAP se ejecuta sobre TCP/IP u otros servicios de transferencia orientados a la conexión

Memoria RAM (*Random Access Memory*)

La memoria RAM o memoria de acceso aleatorio es una forma de memoria de computadora que se puede leer y cambiar en cualquier orden, generalmente utilizada para almacenar datos de trabajo y código de máquina mientras está en funcionamiento. No es una memoria de almacenamiento permanente como los discos duros, sino la memoria donde se ejecutan los procesos de la máquina con su espacio de memoria propios.

NFS (*Network File System*)

El *Network File System* (Sistema de Archivos de Red), es un protocolo para compartición de archivos, creado por Sun Microsystems para SunOS en 1984.

NIS (*Network Information Service*)

El servicio de información de red, o NIS (originalmente llamado *Yellow Pages* o YP), es un protocolo de servicio de directorio cliente-servidor para distribuir datos de configuración del sistema, como nombres de usuario y host, entre computadoras en una red informática. Sun Microsystems desarrolló el NIS. La tecnología tiene licencia para prácticamente todos los demás proveedores de Unix.

RPC (*Remote Procedure Call*)

En computación distribuida, la llamada a procedimiento remoto (*Remote Procedure Call*, RPC) es un programa que utiliza una computadora para ejecutar código en otra máquina remota sin tener que preocuparse por las comunicaciones entre ambas. El protocolo que se utiliza para esta llamada es un gran avance sobre los *sockets* de Internet usados hasta el momento. De esta manera el programador no tiene que estar pendiente de las comunicaciones, estando estas encapsuladas dentro de las RPC.

SMB (*Server Message Block*)

SMB es un protocolo de red que permite compartir archivos, impresoras, etcétera, entre nodos de una red de computadoras que usan el sistema operativo Microsoft Windows. Este protocolo pertenece a la capa de aplicación en el modelo OSI. SMB fue desarrollado originalmente por IBM, pero la versión más común es la modificada ampliamente por Microsoft.

USB (*Universal Serial Bus*)

USB es un bus de comunicaciones que sigue un estándar que define los cables, conectores y protocolos usados en un bus para conectar, comunicar y proveer de alimentación eléctrica entre computadoras, periféricos y dispositivos electrónicos.

UPS (*Uninterruptible Power Supply*)

Un UPS o "fuente de alimentación ininterrumpida" es un dispositivo que proporciona batería de respaldo cuando falla la energía eléctrica o cae a un nivel de voltaje inaceptable. Los sistemas UPS pequeños proporcionan energía durante unos minutos; suficiente para apagar la computadora de manera ordenada, mientras que los sistemas más grandes tienen suficiente batería durante varias horas.

WWW (*World Wide Web*)

La *World Wide Web* (WWW) o red informática mundial es un sistema de distribución de documentos de hipertexto o hipermedia interconectados y accesibles a través de Internet. Con un navegador web, un usuario visualiza sitios web compuestos de páginas web que pueden contener textos, imágenes, vídeos u otros contenidos multimedia, y navega a través de esas páginas usando hiperenlaces.

Enlaces de interés

Sitio de CNET en español

CNET es un sitio web especializado en tecnología que ofrece información, herramientas, consejos y reseñas sobre diversos aspectos que pueden ser útiles para el administrador de sistemas. Su sitio Web principal está en:

https://www.cnet.com/es/

Se destaca el artículo sobre ajustes de seguridad en el sistema operativo Windows 10.

https://www.cnet.com/es/como-se-hace/windows-10-ajustes-de-seguridad/

Sitio Web de Soporte de Microsoft

Se resalta el articulo con reseña de configuración de la seguridad en el sistema operativo Microsoft Windows, en español.

https://support.microsoft.com/es-cl/help/4013263/windows-10-stay-protected-with-windows-security#:~:text=Para%20obtener%20acceso%20a%20ellas,Protecci%C3%B3n%20contra%20virus%20y%20amenazas.

Linux Proffesional Institute (LPI)

Linux Professional Institute (LPI) es la organización estándar de certificación global y de apoyo profesional para profesionales de código abierto. Con más de 175,000 titulares de certificaciones, es el primer y más grande organismo de certificación de fuente abierta y Linux neutral del proveedor. LPI cuenta con profesionales certificados en más de 180 países, ofrece exámenes en varios idiomas para las certificaciones LPIC para administración de sistemas Unix/Linux.

https://www.lpi.org/es/

Linux Advanced Routing & Traffic Control HOWTO

Enlace al documento muy popular de redes y enrutamiento avanzado en sistemas Linux.

https://www.tldp.org/HOWTO/pdf/Adv-Routing-HOWTO.pdf

Socket programming – IBM Knowledge Center

Sección del sitio IBM Knowledge Center dedicada a la programación de *sockets* en redes TCP/IP.

https://www.ibm.com/support/knowledgecenter/ssw_ibm_i_72/rzab6/rzab6soxoverview.htm

IT World

Sitio Web con noticias y tendencias en el mundo de la tecnología de la información (IT).

https://www.itworld.com/

Bibliografía

Burgess, M. (2004). *Principles of Network and System Administration* (2da ed.). John Wiley & Sons, Ltd.

Centeno, M., & Velásquez, R. (Marzo de 2016). *Un algoritmo metaheurístico de recocido simulado para el 3AP-Axial*. Revista SABER.

Chin-Ling, C., Chiang, M.-L., & Lin, C.-B. (21 de Febrero de 2020). *The High Performance of a Task Scheduling Algorithm Using Reference Queues for Cloud- Computing Data Centers*. MDPI Electronics. Obtenido de https://www.mdpi.com/2079-9292/9/2/371/htm

ConceptDraw.com. (s.f.). *Active Directory Diagrams Solution*. Recuperado el 22 de junio de 2020, de https://www.conceptdraw.com/solution-park/computer-active-directory

Descom.es. (s.f.). *Nube pública o privada*. Recuperado el 27 de Junio de 2020, de https://www.descom.es/blog/cloud/nube-publica-o-privada.html

Ferrando Lavila, J. (08 de noviembre de 2017). *Comparativa SAN vs NAS - Infordisa*. Recuperado el 07 de mayo de 2020, de https://www.infordisa.com/es/comparativa-san-vs-nas/#:~:text=Tanto%20SAN%20(Storage%20Area%20Network,operan%20en%20bloques%20de%20disco.

How To Geek. (s.f.). *How to Create, Modify and Delete Scheduled Tasks from the Command Line.* Recuperado el 20 de Junio de 2020, de https://www.howtogeek.com/51236/how-to-create-modify-and-delete-scheduled-tasks-from-the-command-line/

joetheitguy.com. (marzo de 2018). ITSM Basics: *The Top 13 "Approaches" Used by IT Service Desks.* Recuperado el 07 de mayo de 2020, de https://www.joetheitguy.com/itsm-basics-top-13-approaches-used-service-desks/

Limoncelli, T. (1999). *Deconstructing user requests and the nine step model. Proceedings of the Thirteenth Systems Administration Conference (LISA XIII)*. USENIX Association: Berkeley, CA.

Limoncelli, T., Hogan, C., & Chalup, S. (2007). *The practice of system and network administration* (2da ed.). Addison-Wesley.

Maldonado, D. (noviembre de 2017). *¿Cuál es la diferencia entre gestión de incidentes y gestión de problemas?* Recuperado el 17 de mayo de 2020, de http://www.icorp.com.mx/blog/gestion-de-incidentes-gestion-problemas/

Nemeth, E., Snyder, G., Hein, T., & Whaley, B. (2011). *Unix and Linux System Administration Handbook* (4ta ed.). Prentice Hall.

Pediapress. (2012). *System Administration: Configure, Deploy, Maintain and Audit*. Pediapress.

PNGWave. (s.f.). *Serial ATA Hard Drives and others*. Recuperado el mayo de 2020, de https://www.pngwave.com/png-clip-art-kuhfz

Schaumann, J. (2019). *Principles of System Administration*. Recuperado en abril de 2020 de Netmeister.org: https://www.netmeister.org/book

White, S. (enero de 2020). *Top 14 ITSM tools for 2020*. Recuperado el 23 de junio de 2020, de CIO.com: https://www.cio.com/article/3304243/top-itsm-tools.html?upd=1594829364609

White, S., & Greiner, L. (enero de 2019). *What is ITIL? Your guide to the IT Infrastructure Library. CIO.com*. Recuperado el 07 de mayo de 2020, de https://www.cio.com/article/2439501/infrastructure-it-infrastructure-library-itil-definition-and-solutions.html

Wikiwand.com. (s.f.). *Microsoft Windows - Wikiwand*. Recuperado el mayo de 2020, de Wikiwand.com: https://www.wikiwand.com/en/Microsoft_Windows

Agradecimientos

Autor
Wilfredo J. Torres Moya