Tema 2

Administración de bases de datos. Funciones y responsabilidades. Administración de servidores de correo electrónico. Protocolos de Correo electrónico.

Guion-resumen

1. **Administración de bases de datos**
   - 1.1. Arquitecturas de un sistema gestor de base de datos
   - 1.2. Entorno de un sistema gestor de base de datos
   - 1.3. Clasificación de los sistemas gestores de bases de datos
   - 1.4. Seguridad
2. **Funciones y responsabilidades**
3. **Administración de servidores de correo electrónico**
   - 3.1. Componentes de un sistema de correo electrónico
   - 3.2. El correo electrónico y el sistema de nombres de dominio
   - 3.3. Tareas de administración
   - 3.4. Tipos de servidores de correo electrónico
4. **Protocolos de correo electrónico**
   - 4.1. Simple Mail Transfer Protocol, SMTP
   - 4.2. Post Office Protocol, POP
   - 4.3. Internet Message Access Protocol (IMAP)

1. Administración de bases de datos

Las bases de datos son una parte muy difundida y muy presente en las organizaciones que trabajan con información en general y en las de tecnologías de la información en particular. Operaciones cotidianas como transacciones bancarias, reservas de espectáculos, de viajes, búsquedas en sitios web, comparativas de precios o compras en plataformas online están basadas en consultas a bases de datos y nos hacen nuestra vida profesional y personal más sencilla. Y no se trata únicamente de consultas sino de actualizaciones: descontar una entrada o un recambio de automóvil que se han vendido precisa de una actualización en las bases de datos de cada sistema de gestión de base de datos respectivo.

Otro entorno en el que las bases de datos se han convertido en imprescindibles es el de los medios de comunicación online. Su crecimiento exponencial en los últimos años exige una actualización de noticias casi inmediata. Formatos tradicionales como el texto y las imágenes se mezclan con otros más novedosos como el audio y el vídeo, incluso posibilitando su flujo continuo (en inglés, *streaming*). Las páginas web que muestran la información consultan las bases de datos que contienen las noticias a publicar y también todo este universo multimedia.

Más aún, los sistemas de información geográfica almacenan mapas, imágenes de satélite o datos proporcionados por sensores meteorológicos que posteriormente habrá que presentar previo procesamiento. La presencia casi masiva de tecnologías analíticas en tiempo real es utilizada en muchas organizaciones para controlar sus procesos de fabricación, financieros o de producción.

De manera no muy diferente al de otras disciplinas de rápido crecimiento, los estándares apenas alcanzan a mantener la velocidad del desarrollo vertiginoso de las tecnologías de bases de datos.

Como consecuencia, se ha generado un gran número de productos comerciales, cada cual con el sello específico de su fabricante: son los conocidos como Sistemas Gestores de Base de Datos (SGBD), conjuntos de programas que proporcionan la capacidad de gestionar datos y su almacenamiento y proceso en una base de datos. Además, se han producido diversos modelos de bases de datos, desde el jerárquico o el de red, pasando por el orientado a objetos, el relacional o el de objetos-relacional.

En la práctica, la mayor parte de los motores de búsqueda que utilizan los usuarios para acceder a la información se basan en la arquitectura relacional y utilizan el lenguaje de consultas SQL o alguna de sus variaciones para operar con los datos pero en los últimos años, con la explosión del concepto nube y del software como servicio, estamos asistiendo al auge de un nuevo modelo, no relacional, que parece estar dando respuesta a los problemas de alta escalabilidad que presentan los modelos relacionales, pues a medida que aumenta la complejidad se hacen menos intuitivos y más pesados a la hora de devolver peticiones y ejecutar consultas que son cada vez más complejas.

Estos sistemas se conocen como NoSQL. Algunos de los productos desarrollados son MongoDB, Cassandra, BigTable, Hadoop, entre otros.

El mayor reto o desafío a la hora de implantar una base de datos es el de diseñar correctamente su estructura. Además, una elección errónea a la hora de decidirse por el sistema gestor de base de datos que se desea utilizar, la no aplicación de las mejores prácticas para organizar los datos requeridos, el desconocimiento de la organización y de sus flujos de trabajo podrían terminar convirtiendo en pesadísima una tarea que estaba destinada a hacer más competitiva a la organización.

Se ha comprobado que resulta innegable la ventaja que supone utilizar un sistema gestor de base de datos, pero pueden existir condicionantes que pongan alguna duda a la hora de comprometerse en su adquisición, instalación, configuración y mantenimiento:

— Puede suponer una inversión muy elevada en software, hardware y formación para los administradores.

— Exigencia de una inversión en seguridad, recuperación y garantía de la integridad que la organización puede no permitirse.

— Existencia en la organización de bases de datos, sencillas pero con muy pocas posibilidades de sufrir cambios y que no precisan de un entorno mayor.

— Falta de una necesidad de acceso multiusuario a los datos.

Existen organizaciones que optan por no utilizar un sistema gestor de base de datos de propósito general. En algunas la decisión se ha producido porque existen dificultades con la gestión de archivos y formatos propietarios de las aplicaciones (por ejemplo en aplicaciones de diseño asistido por ordenador, CAD). En otras porque, internamente, el software estaba ya optimizado para un acceso jerárquico a los datos, permitiendo un acceso de forma muy eficaz (por ejemplo, en aplicaciones de sistemas de geolocalización o en sistemas de comunicaciones). En ambos casos, el uso de un sistema gestor de base de datos genérico no representaría una ventaja.

1.1. Arquitecturas de un sistema gestor de base de datos

Los sistemas gestores de bases de datos no han sido insensibles a los cambios tecnológicos, a las tendencias, a la computación distribuida y a la irrupción de los servicios web. Se ha pasado de sistemas de arquitectura monolítica –en el que todo el software estaba integrado– a sistemas modulares de arquitectura cliente-servidor.

— **Arquitectura de tres esquemas.**

  También denominada arquitectura ANSI/SPARC, tiene como objetivo separar las bases de datos de las aplicaciones de usuario.

  Esta separación permite a los usuarios una visualización de los niveles del esquema de un sistema de base de datos: el **externo** (de los usuarios), el **conceptual** (del sistema) y el de **almacenamiento interno** para diseñar la base de datos.

— **Arquitectura cliente/servidor centralizada.**

Se trata de la evolución de la arquitectura centralizada, en la que toda la funcionalidad del sistema gestor de la base de datos se realizaba en un sistema central. A medida que aumentaba la capacidad de cómputo en los terminales de los usuarios empezaron a desarrollarse las arquitecturas cliente/servidor.

— **Arquitecturas cliente/servidor básicas.**

Su objetivo es la definición de sistemas especializados con funcionalidades muy específicas. Las máquinas cliente proporcionarán las interfaces necesarias para permitir el acceso a esos sistemas.

— **Arquitecturas cliente/servidor de 2 capas.**

En esta arquitectura los componentes software están repartidos en dos sistemas, cliente y servidor. Su simplicidad y compatibilidad con los sistemas existentes en el momento de su desarrollo impulsaron su éxito.

Los primeros componentes que cambiaron de la parte servidor a la parte cliente fueron las aplicaciones y la interfaz de usuario. La conectividad entre ambas partes la garantiza el estándar ODBC (Open Database Connectivity) pues proporciona una API (Application Programming Interface) que posibilita que las aplicaciones del lado del cliente contacten con el sistema gestor de base de datos siempre que ambas partes cuenten con el software instalado, por supuesto.

— **Arquitecturas de 3 capas y "n" capas.**

La aparición de la web obligó a redefinir los roles de clientes y servidores. La respuesta del mercado fue la arquitectura de 3 capas.

Consiste en añadir una capa intermedia entre la parte del cliente y la del servidor del sistema gestor de la base de datos. En algunas ocasiones, -en función de la aplicación-, se denomina servidor de aplicaciones y en otras servidor web.

En esta arquitectura, las interfaces y algunos aspectos específicos de las aplicaciones residen en la parte cliente mientras que el servidor intermedio aceptará, procesará y remitirá solicitudes y comandos al servidor de base de datos. Posteriormente transmite el resultado desde la base de datos a los clientes para su procesamiento final.

En el caso de que la capa del sistema gestor de base de datos se dividiese en dos capas, por ejemplo un servidor web y un servidor de base de datos, la arquitectura resultante contaría con 4 capas. Si se realiza alguna división adicional en las capas existentes entre el usuario y los datos almacenados, obtendríamos arquitecturas de *"n"* capas. En estas arquitecturas cualquiera de sus capas puede ejecutarse en plataformas o sistemas independientes. Las sofisticadas plataformas de gestión empresarial (ERPs o CRMs) son ejemplos de este tipo de arquitectura.

1.2. Entorno de un sistema gestor de base de datos

Los sistemas gestores de base de datos cuentan con diversos componentes software pero también es preciso detallar aquellos componentes del sistema operativo con el que interactúan:

— **Módulos.**

En una arquitectura cliente/servidor, es habitual que la funcionalidad se reparta entre el módulo cliente y el módulo servidor. El primero suele ejecutarse en un sistema local y mediante interfaces más o menos sencillas e intuitivas permite la interacción del usuario con la base de datos. El segundo se encarga de la gestión del almacenamiento, del acceso y de la búsqueda de los datos.

Normalmente tanto la base de datos como el catálogo del sistema gestor de base de datos se almacenan en el disco del sistema que, principalmente, está controlado por su sistema operativo. El módulo administrador de los datos controlará el acceso a la información guardada en el disco, forme parte de la base de datos o del catálogo.

Un compilador procesará las definiciones del esquema almacenando las descripciones –los metadatos– en el catálogo del sistema gestor de base de datos. El catálogo resulta de gran importancia pues incluye nombre y tamaño de archivos y de los elementos de datos, información sobre las restricciones, las correspondencias entre esquemas y otra información que los módulos del sistema gestor de base de datos necesita. Estos módulos buscarán en el catálogo la información que precisan.

— **Utilidades del sistema gestor de base de datos.**

El administrador de la base de datos cuenta habitualmente con un conjunto de utilidades que le ayudan a resolver ciertas tareas:

- **Carga de datos.** La existencia de numerosos sistemas gestores de base de datos ha convertido en común la carga de datos entre sistemas. Estas herramientas suelen contar con capacidades de conversión para preparar y adaptar las descripciones internas de los datos.
- **Copia de seguridad.** Estas utilidades crean copias de la base de datos en otros soportes para, en caso de necesidad, poder restaurar dicha base de datos. Las organizaciones deben contar con una estrategia de copia seguridad, definiendo el tipo –completo, diferencial, incremental– y su programación.
- **Reorganización del almacenamiento de la base de datos.** En ocasiones puede ser conveniente una reorganización para mejorar el rendimiento.
- **Monitorización del rendimiento.** La obtención de estadísticas resulta de gran utilidad al administrador. Su análisis le servirá para adoptar decisiones, tanto de forma proactiva como reactiva.

— **Herramientas para los diseñadores.**

En la fase de diseño de la base de datos es muy común apoyarse en las herramientas *CASE* (acrónimo en inglés para *Computer-Aided Software Engineering*).

Otra herramienta particularmente utilizada en grandes organizaciones y que puede ayudar a las tareas del administrador es el sistema de diccionario de datos, que aumenta las prestaciones del catálogo almacenando información sobre descripción de aplicaciones, decisiones de diseño o información de usuarios.

— **Entornos de desarrollo de aplicaciones.**

Incluyen servicios que ayudan en numerosos aspectos del diseño de la base de datos, de las interfaces gráficas, actualizaciones y consultas o, incluso, en el desarrollo de una aplicación. Algunos ejemplos podrían ser el entorno Eclipse o *Visual Studio*.

— **Software de comunicaciones.**

Su función es garantizar la conectividad de usuarios remotos con el sistema gestor de base de datos. El sistema compuesto por el sistema gestor de base de datos y el sistema de comunicación se denomina sistema DB/DC.

1.3. Clasificación de los sistemas gestores de bases de datos

Existen numerosos sistemas gestores de bases de datos y a veces no es sencilla su categorización. Una posible clasificación atendería a los siguientes criterios:

— **Según el modelo de datos en el que está basado.**

- **Modelo de datos relacional.** Presenta la base de datos como una colección de tablas, cada una de las cuales puede guardarse como un archivo separado.
- **Modelo de datos de objetos.** Define la base de datos como objetos, propiedades y operaciones. Si los primeros cuentan con una estructura y comportamiento iguales pertenecerán a una clase, que se organizan en jerarquías. Los procedimientos definidos para las operaciones son los métodos.
- **Modelo de datos objeto-relacional.** Se trata de una ampliación del modelo de datos que incluye más objetos y capacidades.
- **Modelo de datos jerárquico.** Pertenece a un modelo de datos muy importante, histórico, que ahora se denomina heredado. Representa los datos como estructuras en forma de una jerarquía de árboles que muestra un conjunto de registros relacionados.
- **Modelo de datos de red.** También pertenece al modelo de datos heredados. Representa los datos como tipos de registros.

- **Modelo de datos no relacional.** Habitualmente denominado sistema *NoSQL*, se ha convertido en tendencia desde el comienzo del siglo XXI por su capacidad de almacenamiento de datos sin normalizar y su gran capacidad de escalado.
- **Modelo de datos de *XML*** (*eXtended Markup Language*). Está considerado como el estándar para el intercambio de datos a través de Internet. Utiliza estructuras jerárquicas en árbol. Los datos se representan como elementos. El uso de etiquetas permite su anidamiento en estructuras muy complejas.

El modelo relacional sigue siendo el más comúnmente utilizado en los sistemas comerciales.

— **Según el número de usuarios soportado.**

- Sistemas de un **único usuario**.
- Sistemas **multiusuario**, soportado por la inmensa mayoría de sistemas gestores de bases de datos.

— **Según el número de sitios sobre los que se distribuyó la base de datos.**

- **Centralizado**, los datos están almacenados en un único sistema.
- **Distribuido**, tanto base de datos como software asociado puede estar repartido entre varios sistemas.
- **Homogéneos**, utilizan el mismo software de gestión de base de datos en varios sitios.
- **Federados**, los sistemas gestores de base de datos cuentan con cierta autonomía aunque se acoplan en sistemas homogéneos.

— **Según su propósito.**

- De propósito **general**.
- De propósito **especial**. Existen determinados factores, por ejemplo el rendimiento, que pueden conducir a la construcción de un sistema gestor de base de datos para una aplicación específica. Como ejemplos destacan los antiguos sistemas de gestión de reservas aéreas o directorios telefónicos.

— **De código libre o comerciales**.

Probablemente los más conocidos y utilizados en la primera categoría serían MySQL y PostgreSQL, mientras que en la segunda categoría encontramos a Oracle o Microsoft SQL Ser*ver*, por ejemplo.

1.4. Seguridad

Al igual que en otros entornos de tecnologías de la información, la seguridad en la administración de las bases de datos se ha convertido en un factor imprescindible. Se trata, además, de una disciplina transversal que comprende varios aspectos:

— **Legales**. Es necesario adecuarse a la legislación vigente en cuanto al derecho de acceso a la información propiedad de una organización.

— **De políticas**. Hay que conocer qué políticas existen sobre el tipo de información que puede mostrarse tanto a nivel de gobierno de un Estado, como institucional o de una organización.

— **La situación del sistema de información**. Puede ser necesario relacionar aspectos de seguridad del sistema gestor de base de datos con otros niveles, como el de la situación de seguridad del hardware o del software que lo sustenta para garantizar de este modo una homogeneidad en las medidas implantadas.

— **Clasificación de la seguridad**. Existen distintas categorías, por ejemplo desde alto secreto a no clasificado, pasando por secreto y confidencial. Resulta vital lograr una clasificación de la seguridad de los datos de una organización para definir las medidas a adoptar.

Las amenazas a los datos suelen provocar su pérdida pero también su degradación. Conviene no olvidar que conceptos comúnmente aceptados como característicos de las bases de datos, -integridad, disponibilidad y confidencialidad-, pueden verse parcial o totalmente afectados en el caso de sufrir algún ataque.

— **Pérdida de la integridad**. El concepto de integridad exige que la información se encuentre protegida ante modificaciones no autorizadas, sean accidentales o intencionadas. Si no es posible recuperar la integridad perdida, puede producirse una toma de decisiones, que, presumiblemente sería errónea, inexacta y probablemente, fraudulenta.

— **Pérdida de la disponibilidad**. Los datos han de estar disponibles, bien para un usuario, bien para un software que cuente con privilegios de acceso.

— **Pérdida de confidencialidad**. Se pierde la confidencialidad cuando se produce un acceso no autorizado a los datos. Las consecuencias de esta situación son múltiples: desde económicas por destrucción o robo hasta legales. No es asunto menor el desprestigio que puede suponer para una organización que sus datos sean accedidos sin autorización.

Las soluciones más habituales para evitar estas amenazas se resumen en las siguientes medidas de control:

— Control de accesos.

— Control de daños.

— Control de flujo.

— Cifrado de las comunicaciones.

El administrador de la base de datos cuenta con todas las posibilidades de administración. Estos **privilegios** le permiten:

— **Crear cuentas.** Esta acción permite añadir una nueva cuenta y contraseña de acceso al sistema gestor de base de datos, tanto a un usuario como a un grupo de ellos.

— **Otorgar privilegios.** Concede permisos a las cuentas.

— **Revocar privilegios.** Cancela permisos previamente concedidos a las cuentas.

— **Asignar el nivel de seguridad.** En función de los niveles de seguridad definidos, el administrador podrá asignar cuentas de usuario al nivel adecuado.

La realización eficiente de estas cuatro tareas permite una situación segura, tanto en el acceso al sistema gestor de base de datos como en la seguridad de las acciones que en él se realicen, gracias a la concesión y retirada de privilegios. Finalmente, es considera obligatoria una asignación de nivel de seguridad siempre de acuerdo a la estructura de la organización.

2. Funciones y responsabilidades

Como en cualquier otro servicio en el que varios usuarios precisan acceder a los mismos recursos, la administración de una base de datos exige la presencia de una o varias figuras implicadas en su diseño, uso y mantenimiento. Naturalmente, este sería el caso de organizaciones de tamaño medio y grande donde existen numerosas bases de datos y además de gran tamaño y que son accedidas por cientos o miles de usuarios. En las pequeñas organizaciones o en aquellas que únicamente cuentan con bases de datos locales –que no son compartidas-, todas esas funciones las suele realizar alguno de los usuarios. También es habitual que la fase de diseño e implementación así como los primeros pasos en su administración se contraten a una empresa externa para, posteriormente, recaer en algún empleado de la organización.

Si establecemos que nos encontramos en una organización que sí cuenta con numerosas bases de datos y que son accedidas por muchos usuarios y de distinta naturaleza, podríamos establecer los siguientes perfiles:

— **Administradores**. La responsabilidad prioritaria de un administrador de base de datos es la administración de los recursos relacionados con los datos, esto es, la base de datos como tal y el sistema gestor de base de datos y su software asociado. En terminología inglesa este rol es conocido como *Database Administrator* (DBA). Se responsabilizará de:

  • Establecer el acceso a la base de datos.

  • Coordinación y monitorización de su utilización.

- Adquisición de los recursos hardware y software precisos.
- Garantizar tiempos de respuesta adecuados.
- Garantizar un nivel de seguridad para los datos y la infraestructura tecnológica que los soporta.

La capacitación para convertirse en DBA suele estar marcada por los distintos fabricantes. Por ejemplo, en el caso de Microsoft, su producto es Microsoft SQL Server. Cuenta con un itinerario formativo dividido en niveles (Principiante, Asociado y Experto). Como suele ser habitual en esa organización, facilitan la posibilidad de obtener la certificación (Microsoft Technology Associate, MTA; Microsoft Certified Solutions Associate, MCSA y Microsoft Certified Solutions Expert, MCSE, respectivamente). El caso de Oracle es similar pues proporciona tanto itinerario formativo como la posibilidad de certificación. También se estructura en niveles, que capacitan para la administración al rendimiento o la seguridad. Para la certificación, la denominación es Oracle Certified Associate (OCA), Professional (OCP), Master (OCM), Specialist (OCS) y Expert Program (OCEP). En el caso de los sistemas gestores base de datos de código abierto, como MySQL o PostgreSQL, resulta algo más complejo encontrar tanto itinerarios de formación como certificaciones. Análogamente a otros entornos de software libre, la capacitación se logra de forma muy autodidacta y la experiencia se gana con el trabajo diario.

— **Diseñadores**. Su responsabilidad es la de identificar los datos que se guardarán en la base de datos así como de la elección de las estructuras necesarias tanto para representarlos como para almacenarlos. Estas tareas suelen implementarse antes de comenzar el proceso de carga de los datos.

  Para abordar estas tareas ha de existir un trabajo previo de recogida de requisitos por parte de los diseñadores, que se habrán reunido con los usuarios, de modo que el diseño final satisfaga sus necesidades. Suelen desarrollar vistas de las bases de datos diseñadas que, posteriormente, son analizadas e integradas con otras vistas de otros grupos de usuarios. Finalmente, el diseño de la base de datos debería soportar los requisitos planteados por todos los grupos de usuarios.

  No es inusual que los diseñadores formen parte del equipo de administradores de la base de datos, pues facilita la comunicación, el intercambio de ideas y la ejecución final del diseño presentado.

— **Analistas de sistemas**. Determinan tanto los requisitos de los usuarios finales (sobre todo para los denominados usuarios paramétricos) así como las especificaciones de desarrollo para ciertas transacciones, como son consultas y actualizaciones estándar.

— **Programadores de aplicaciones**. Implementan las especificaciones de desarrollo dándole forma de programas. Posteriormente verificarán, documentarán y mantendrán las transacciones.

— **Usuarios finales**. Son las personas que precisan acceso a las bases de datos para consultar, actualizar y emitir informes. Existen numerosos perfiles de usuarios finales, que podrán cambiar en función de cada organización. Una posible clasificación sería:

  • **Usuarios paramétricos**. En otras clasificaciones se les denomina Principiantes. Representan la gran mayoría de usuarios que acceden a las bases de datos. Suelen limitarse a consultas y actualizaciones estándar (por ejemplo consultas para comprobar la disponibilidad de una entrada de teatro, una reserva de hotel, etc.). Para esta labor suelen utilizar interfaces intuitivas aunque podrían utilizan también lenguajes estándar de consulta a bases de datos.

  • **Usuarios casuales**. Precisan un acceso puntual a la base de datos, aunque sí podrán necesitar informaciones diferentes en cada acceso. Estos usuarios suelen acceder mediante el uso de lenguajes sofisticados de consulta de bases de datos. Su perfil suele ser el de administradores de nivel medio o el de usuarios avanzados.

  • **Usuarios finales de perfil sofisticado**. En esta categoría suelen estar presentes roles como analistas de datos, científicos o ingenieros, todos ellos muy familiarizados con el SGBD.

  • **Usuarios finales independientes**. Son capaces de mantener sus propias bases de datos mediante el uso de programas a medida. Dichos programas suelen proporcionar interfaces cómodas que facilitan la tarea.

El nivel de conocimiento del sistema gestor de base de datos que cada tipología de usuario alcanzará es muy variado. Es muy común que los usuarios finales independientes se conviertan en auténticos expertos del software que utilizan, mientras que los usuarios paramétricos no precisan de mucho conocimiento del SGBD para realizar sus tareas, si acaso familiarizarse con la interfaz con la que efectúan consultas. En el otro extremo se encuentran los usuarios finales de perfil sofisticado, que exprimen la gran mayoría de las prestaciones del SGBD para satisfacer sus, con frecuencia, grandes demandas de datos.

Debido a la complejidad de un sistema gestor de base de datos, podemos encontrar otra tipología adicional de usuarios asociados con el funcionamiento de su entorno, tanto en el software como en el sistema:

— **Diseñadores e implementadores del SGBD**. Su función es la de diseñar e implementar tanto los módulos que lo componen como las interfaces. El SGBD ha de interactuar con el sistema operativo y con los compiladores de los posibles lenguajes de programación. La enorme complejidad de un SGBD ha de atender a un gran número de aspectos críticos:

  • El catálogo.

  • El lenguaje de consulta.

  • El procesamiento de la interfaz.

- La gestión de los *buffers.*
- El control de la concurrencia.
- La seguridad de los datos.
- La recuperación de los datos.

— **Desarrolladores de herramientas**. Con mucha frecuencia los SGBD no cuentan con herramientas de alto nivel que permitan a sus administradores el modelado, el diseño de la base de datos, el del sistema o la monitorización de su rendimiento. Suelen suministrarse por separado y en ocasiones por empresas desarrolladores de software que son independientes a los fabricantes de SGBD. En el momento de su implantación, los técnicos encargados necesitarán acceso al SGBD para realizar comprobaciones y validaciones.

— **Administradores de sistemas, técnicos, operadores y personal de mantenimiento**. Su tarea es la administración del sistema que soporta el SGBD. Mantienen el entorno software y hardware.

3. Administración de servidores de correo electrónico

Junto al servicio *World Wide Web* (www), el servicio de correo electrónico es, posiblemente, el más ampliamente utilizado ocupando una importante porción del tráfico de Internet y se ha constituido desde hace años como un elemento de comunicación imprescindible en cualquier organización. Si se compara con otros servicios que sí han visto una modificación sustancial, el correo electrónico no ha sufrido demasiados cambios. Sin embargo, sí que está sufriendo en mayor cantidad toda la problemática de seguridad. Son innumerables los ataques y la diversidad de los mismos que se centran en los servidores de correo electrónico y en el propio contenido del correo. Por ello, garantizar la seguridad en la recepción y emisión de este medio de comunicación se ha convertido en uno de los grandes desafíos para los administradores, tanto de correo como de seguridad.

Ninguna organización puede arriesgarse a que su servicio de correo electrónico sea inseguro. La presencia de virus o de correo basura (*spam*) puede significar un gran daño y un alto coste, tanto económico como incluso de prestigio. Conviene no olvidar que también una organización puede convertirse en generadora de correo basura (*spammer*) si no configura correctamente su servicio de correo electrónico y permite el *Open Relay*, esto es que cualquier usuario de Internet pueda utilizarlo para enviar su correo. Esta configuración permite el envío de correos masivos y no deseados. Muchos proveedores de servicios de Internet (*ISP*) rechazan el correo procedente de retransmisiones abiertas por lo que una organización sin su servicio de correo correctamente configurado podría dejar sin correo a sus usuarios. Además, es más que probable que aquellos servidores de correo generadores de correo basura sean incluidos en listas negras. Estas listas son consultadas por los sistemas de los proveedores de servicios de Internet antes de emitir correo, por lo que podrían denegar cualquier correo del servidor de una organización que estuviese incluido.

De lo anteriormente expresado puede deducirse que es obligatorio contar con un administrador o equipo de administradores que garanticen una eficiente administración del servicio de correo electrónico. Como en cualquier otro servicio de administración, la experiencia es un factor de incalculable valor. Por tanto, la formación continua de los administradores se convierte en casi imprescindible. Es muy habitual que esta formación sea autodidacta. No obstante, también en el caso del correo electrónico, los distintos fabricantes cuentan con ofertas formativas unidas a las certificaciones que permitirán a las organizaciones garantizarse la presencia de profesionales formados en las distintas herramientas.

En el caso de Microsoft, su producto de servidor de correo es Microsoft Exchange Server, actualmente en la versión 2016. La certificación ofrecida se denomina Microsoft Certified Solutions Expert (MCSE). Con ella el fabricante trata de validad la capacidad de los administradores en la implementación y administración de un entorno de mensajería empresarial con Exchange Server.

En el caso de los entornos basados en UNIX y Linux, el camino formativo es ofrecido por los distintos fabricantes de distribuciones. Oracle para su producto Solaris o Red Hat para su plataforma Red Hat Enterprise Linux cuentan con itinerarios formativos. Igualmente, ofrecen la posibilidad de certificación. En otras distribuciones Linux resulta más complicado y a menudo la única posibilidad con la cuentan los administradores es la autoformación debido a la escasez de posibilidades formativas más o menos regladas. A cambio, estas distribuciones suelen contar con una comunidad de usuarios muy activos y colaborativos, lo que permite estar el día en todo tipo de detalles que afectan al producto.

3.1. Componentes de un sistema de correo electrónico

En líneas generales, un sistema de correo electrónico está constituido por los siguientes componentes:

— **Agente de transferencia de correo** (en inglés *Mail Transfer Agent*, MTA). Está considerado como el componente más importante. Es responsable de la transferencia de correo electrónico a través del protocolo SMTP, tanto en un sistema como hacia el exterior del sistema o de la organización. Debe garantizar la recepción de los mensajes, tanto a otros servidores de correo como a los buzones de los usuarios. Además, se encarga del encaminamiento del correo entre diversos sistemas. Entre sus funciones se encuentra:

  - Debe ser capaz de detectar si el correo es local y, en caso afirmativo, ceder el control a un cliente de correo local.
  - Si el correo es remoto, debe ser capaz de reescribir las direcciones de correo del destinatario y remitentes del correo de manera que sean compatibles con el sistema remoto y con el agente de transporte.
  - Debe ser capaz de reconocer alias, así como de manejar ficheros de reenvío (*forwarding*).
  - El agente de transferencia de correo debe ser altamente configurable ya que cada sistema tiene sus propios usuarios, confi-

guraciones, requerimientos, condicionantes, etc. Es preciso garantizar que no existan limitaciones para ningún usuario.

- Gestión de la seguridad. Conviene recordar que se trata de establecer una conexión con un sistema remoto, cuya configuración se desconoce. Es necesario procurar que se autentifiquen tanto el origen como el destinatario del mensaje, así como seleccionar la ruta óptima
- El encaminamiento del correo debe ser rápido, de confianza y que consuma pocos recursos. El sistema debe garantizar que el correo llegue a su destino o sea rechazado como no válido pero no debe perderse

— **Agente de usuario de correo** (en inglés *Mail User Agent*, MUA). Conocido habitualmente como cliente de correo. Es una interfaz de usuario que permite editar, componer y enviar correo local. Algunos ejemplos son Mail, Pine, Outlook, Thunderbird o Eudora. Algunas MUAs pueden utilizar técnicas de túnel para remitir correo a través de pasarelas punto a punto así como otras características más avanzadas, por ejemplo gestionar listas.

— **Agentes de acceso**. Suelen encargarse de realizar funciones intermedias entre MTAs y MUAs, como clasificación, distribución o verificación. Son los siguientes:

- **Agente de envío de correo** (en inglés *Mail Delivery Agent,* MDA). Su misión es la distribuir y clasificar mensajes en la máquina local para que, posteriormente, sean accedidos por un cliente de correo. Un agente MDA es llamado por un agente MTA para realizar su función. En ocasiones, los clientes de correo ya cuentan con funciones integradas de MDA.
- **Agente de entrega de correo** (en inglés *Mail Submission Agent*, MSA). Su principal misión es la de verificar los mensajes entrantes en un MTA, descargando de trabajo al agente MTA en grandes flujos de correo y posibilitando la realización de controles de seguridad más efectivos.

3.2. El correo electrónico y el sistema de nombres de dominio

El protocolo SMTP es capaz de gestionar el reenvío de mensajes entre distintos sistemas siempre que el emisor conozca el sistema destino a quien enviárselos.

El sistema de nombres de dominio (en inglés, *Domain Name System,* DNS) es de vital importancia en la administración del correo electrónico. Es utilizado tanto por los clientes como por los servidores. Incluso en el caso de que la organización no administre los servidores DNS, es necesario un amplio conocimiento de su funcionamiento por parte del administrador de correo.

3.2.1. Tipos de registros

En muchos típicos escenarios de red únicamente se utilizan dos tipos de registros: los de tipo A y los de tipo PTR. Establecen la correspondencia entre un nombre de equipo (en inglés *hostname)* y una dirección IP y entre una

dirección IP y un nombre de equipo respectivamente. Estos registros pueden ser utilizados también en escenarios de correo electrónico. Existe un tipo de registro que se utiliza exclusivamente en entornos de correo electrónico: el registro Mail Exchanger o MX.

El registro MX responde a la pregunta de cómo sabe un servidor SMTP a qué servidor de correo en un dominio dado ha de entregar el correo. Como no podía ser de otra forma, el sufijo correspondiente al nombre del dominio se utiliza para realizar las búsquedas directas *DNS* (*DNS lookups*). La primera búsqueda se realiza para el valor del registro MX en el DNS. Esta entrada permite al operador del DNS especificar uno o más servidores que pueden recibir correo en un dominio. El típico ejemplo lo constituye el dominio ejemplo.es. La entrada MX del DNS podría ser correo.ejemplo.es.

Si no existe una entrada MX en el DNS, se intenta buscar un registro de tipo A en el dominio destino. Si la búsqueda resulta exitosa, el correo se envía. En caso contrario, esto es, si ni el registro MX ni el registro A devuelven un resultado satisfactorio, el mensaje es devuelto al remitente.

Existen dos grandes motivos para especificar registros MX en el DNS:

— No es deseable estar forzado a hacer corresponder un registro A con el servidor de correo de una organización. Imaginemos una organización que cuenta con su dirección web http://www.ejemplo.es. La organización podría desear que los visitantes accedieran a ella a través de una URL más corta, por ejemplo http://ejemplo.es. Sin embargo, puede que no quiera que su servicio web se ejecute en el servidor de correo o viceversa. En este caso, la definición de un registro MX solucionaría el problema.

— La razón más importante es que la búsqueda directa en el DNS no devuelve únicamente una lista de nombres de equipos sino que también contiene un valor que expresa la prioridad del equipo en la lista. La lista muestra los datos en modo ascendente, de modo que el equipo con el número de prioridad más bajo será contactado antes. Si dos equipos cuentan con el mismo número de prioridad, el sistema de contacto es aleatorio.

Un establecimiento de un mismo valor de prioridad en los registros MX puede servir como método poco sofisticado de balanceo de carga. También se consigue haciendo corresponder un registro A con diferentes direcciones IP. El uso de registros MX permite la configuración de servidores de correo de respaldo simplemente asignando diferentes valores a su prioridad. Esto no puede hacerse con registros de tipo A. La siguiente tabla mostraría un ejemplo:

PRIORIDAD	NOMBRE DE EQUIPO
10	mx1.ejemplo.es
10	mx2.ejemplo.es
20	mx3.ejemplo.es
30	mx4.ejemplo.es

En el caso de establecer esta configuración en el sistema DNS del dominio ejemplo.es, los servidores SMTP tratarán de enviar correo en primer lugar a los equipos mx1.ejemplo.es y mx2.ejemplo.es. En el caso de que ambos equipos fallasen, se intentaría enviar el correo al sistema mx3.ejemplo.es y finalmente, en caso de fallo, a mx4.ejemplo.es. En el caso de que éste último también fallara, el correo es guardado y se intentaría su entrega pasados unos instantes.

3.3. Tareas de administración

Las tareas principales en la administración de un servidor de correo electrónico podrían ser:

— Gestión de usuarios.

— Gestión del tamaño de los buzones de correo.

— Gestión de listas de correo.

— Gestión del servicio de correo web (*webmail*). Una solución de correo web es una aplicación que se ejecuta en un servidor, es accesible a través de un navegador y permite funcionalidades similares a un cliente de correo convencional.

— Gestión del servidor de noticias. Es un sistema encargado de servir los contenidos de los grupos de noticias a los que se suscriben los usuarios.

— Establecimiento del número máximo de conexiones permitidas.

— Gestión de la seguridad del sistema de correo.

  • Prevención del *relay* de correo.

  • Prevención del correo basura.

  • Prevención de la existencia de virus en el sistema de correo.

3.4. Tipos de servidores de correo electrónico

3.4.1. Sendmail

Es el programa que permite la emisión de correo para otros programas y scripts en los sistemas UNIX y Linux. Muchos de los agentes de usuario de correo (MUAs) como Pine o Mail y algunos programas que proporcionan correo web (webmail) como por ejemplo Squirrelmail, utilizan la interfaz de Sendmail para la emisión de mensajes de correo.

Cuenta con una arquitectura monolítica, es muy robusto y puede procesar mensajes de correo en la práctica total de redes. La parte negativa está en su casi mítica complejidad de configuración y administración, sobre todo en la parte de seguridad.

3.4.2. Postfix

Es un MTA modular, escrito en lenguaje C, de software libre y liberado en 1998 con el nombre de VMailer. Es muy ligero, apenas sobrecarga los sistemas puesto que únicamente se cargan aquellos módulos que se precisan en un momento dado. Funciona en prácticamente todas las versiones de sistemas operativos basados en UNIX y Linux.

Como MTA puro que es, Postfix no proporciona servicio alguno que permita a los usuarios recibir su correo vía protocolos POP o IMAP. Esta tarea ha de ser realizada por programas adicionales. Un ejemplo lo constituiría Courier IMAP.

3.4.3. Microsoft Exchange Server

Un sistema de correo Exchange Server se organiza jerárquicamente. La parte más alta se denomina organización y comprenderá el conjunto de sistemas –clientes y servidores– que ejecutan y utilizan el sistema de correo.

La administración puede simplificarse agrupando diversos sistemas clientes y servidores en unidades administrativas dentro de la organización. Estas unidades reciben el nombre de sitios (*sites*).

Un servidor Exchange consta de estructuras de datos y unidades funcionales que persiguen los siguientes objetivos:

— Almacenamiento centralizado e información administrativa tanto para la organización como para cada sitio definido en ella.

— Control del acceso a la información administrativa.

— Gestión de la transferencia y encaminamiento de los mensajes tanto dentro de la propia organización como hacia el exterior, habitualmente a través de Internet.

— Monitorización y control de los procesos y de los servidores de Exchange.

— Gestión de la replicación y distribución entre los distintos servidores de la organización de la información relativa tanto a carpetas públicas como a directorios.

Para lograr los objetivos citados, un servidor *Exchange* cuenta con diversos componentes. Los más relevantes podrían ser:

— **Cliente de Exchange**. Es el método principal para la recepción y envío de correos. Se encuentra habitualmente instalado en las estaciones de trabajo de la red.

— **La aplicación Exchange Administrator**. Controla la mayoría de las funciones administrativas. Gracias a ella el administrador podrá definir sitios, administrar usuarios, configurar conectores, etc.

— **El servicio de directorio**. Almacena toda la información administrativa necesaria para la configuración de la Organización.

— **Almacén de la información**. Constituye el punto principal de almacenamiento para los buzones de usuario y las carpetas públicas.

— **Agente de transferencia de correo (MTA)**. Encamina los mensajes a otras MTA de Exchange o a pasarelas para sistemas de correo externo.

— **Servicio de gestión de claves**. Conocido como Key Management, proporciona seguridad avanzada mediante el uso de firma electrónica y cifrado de mensajes.

— **Servicio de mantenimiento de Exchange**. Conocido como System Attendant (SA), se encarga de monitorizar el rendimiento y comportamiento de servicios de Exchange, así como de reconstruir tablas y de controlar la seguridad avanzada.

4. Protocolos de correo electrónico

En los entornos UNIX, las aplicaciones tradicionales de correo electrónico no utilizaban protocolos de red. Les bastaba con acudir, a través del sistema operativo, a los buzones locales para comprobar la existencia de mensajes. Habitualmente, se encuentran en ubicaciones como */var/mail* o */var/spool/mail.* La evolución de las redes, el aumento del uso de Internet y el incremento del volumen de comunicaciones han provocado que no sea suficiente con la existencia de buzones locales.

El protocolo más importante en el correo que se envía a través de Internet es el *Simple Mail Transfer Protocol* (*SMTP*). Su propósito es el enviar mensajes entre dos sistemas. Puede tratarse de dos servidores o de un servidor y otro sistema que actúe como cliente y cuente con un software de correo electrónico ejecutándose (Microsoft Outlook, Eudora, Thunderbird, etc.). Para leer mensajes no se utiliza el protocolo SMTP. En este territorio actúan los protocolos *Post Office Protocol* (POP) e *Internet Message Access Protocol* (*IMAP*). También es posible leer correo a través de un servicio de correo web (*webmail*)

4.1. *Simple Mail Transfer Protocol* (SMTP)

Es un protocolo orientado a línea de texto que se ejecuta sobre el protocolo TCP, lo que le convierte en muy ligero y permite iniciar sesiones únicamente utilizando un simple cliente Telnet.

Un cliente SMTP comienza una sesión conectándose al puerto 25 de un servidor SMTP. Después de la fase de intercambio de saludos, si el servidor acepta la conexión desde el cliente podrá comenzar la primera transacción de correos.

Una transacción de correos consta de tres partes: el remitente, el receptor o receptores y el contenido del mensaje. Cuando el servidor acepta el primer mensaje, se podrá continuar con el envío transacciones adicionales o finalizar la conexión SMTP.

El protocolo SMTP no requiere autentificación lo que permitió desde mediados de la década de los años 90 que los generadores de correo basura (spammers) aprovecharan esta debilidad e inundaran los buzones de correo de

prácticamente todos con correo no deseado, gracias a que podían utilizar los sistemas de correo de otros y enviar su correo basura incluso a listas de destinatarios existentes en los servidores que utilizaban para sus fines. No solo eso sino que también algunos gusanos encontraron esa vía como aprovechable para infectar muchísimos sistemas.

Una combinación de mayor concienciación sobre aspectos de seguridad, la evolución de las aplicaciones y las restricciones impuestas por los proveedores de servicios de Internet y los legisladores de algunos países al correo procedente de servidores de correo abiertos (Open Relay) ha permitido minimizar esta deficiencia. Encontramos ejemplos de estas medidas en la obligación que los servidores que formen parte de un sistema de correo electrónico se conozcan antes de proceder al envío o el la recomendación del uso de redes privadas virtuales (VPNs). También conviene citar las recomendaciones para el envío de correo electrónico publicadas por la *Internet Engineering Task Force* (*IETF*) en la RFC 5068. Por último, es interesante destacar que EE.UU. declaró ilegal en 2003 el envío de correo basura a través de una retransmisión abierta.

Existe numerosa información sobre el protocolo SMTP, mucha de ella publicada en webs y accesible vía Internet. Por supuesto, la documentación recogida en las RFCs puede convertirse en gran ayuda para un administrador. Como ejemplos estarían la RFC 821, en la que se describe el comportamiento básico del protocolo, la RFC 2821, la RFC 2554 o la RFC 2034, que amplían aspectos fundamentales tanto para la comprensión como para la implementación y administración del protocolo SMTP.

4.2. *Post Office Protocol* (POP)

El protocolo POP es el más ampliamente utilizado para la recuperación de los correos electrónicos que se encuentran almacenados en los servidores de correo. A día de hoy, la versión más extendida sigue siendo POP3.

Este protocolo se centra en asignar permisos a los usuarios sobre sus buzones, desde los que podrán descargar sus mensajes a sus máquinas locales y, en su caso, eliminarlos del servidor. Los servidores POP no fueron concebidos para almacenar los mensajes de forma permanente. Existen servicios POP, proporcionados por algunos proveedores, que incluso impiden la posibilidad de dejar en el servidor mensajes en el servidor por lo que obliga a descargarlos al equipo local. Ya se intuye que la gran desventaja del protocolo POP es que únicamente proporciona un sistema de almacenaje intermedio y que sus usuarios han de descargarse los mensajes en otra parte, por ejemplo en el disco duro de sus equipos. Esto resulta poco práctico si se desea consultar los mensajes de correo desde diferentes ubicaciones. Además, desde el punto de vista de la administración del correo, obliga al administrador a disponer un sistema de copia de seguridad para los discos de los equipos donde los usuarios descargan sus mensajes de correo.

Otra desventaja del protocolo POP es que únicamente proporciona una carpeta para cada usuario; cada usuario accede exclusivamente a su carpeta de entrada.

El protocolo POP necesita que el cliente establezca una conexión TCP con el servidor por el puerto 110. Una vez establecida la conexión, el servidor

envía al cliente una invitación seguida por un intercambio de comandos que permitirán, en primer lugar, la autentificación mediante usuario y contraseña y posteriormente la transacción. Aquí podrán mostrarse, descargarse y eliminarse los correos ubicados en el servidor.

De igual modo que para el protocolo SMTP, existe numerosa documentación técnica sobre este protocolo en las RFCs. Alguna de ellas sería la RFC 1939.

4.3. *Internet Message Access Protocol* (IMAP)

El protocolo IMAP está diseñado para permitir que los mensajes de correo se almacenen permanentemente en el servidor. La versión más implantada es IMAP4. La posibilidad de que los mensajes permanezcan en el servidor de correo elimina la necesidad de mantener un casi siempre costoso sistema de copia de seguridad para cada equipo local de usuario y soluciona la universalidad del acceso al correo para los usuarios. El protocolo IMAP cuenta con características para conexiones *Transport Layer Security* (*TSL*), haciéndolo más seguro y resistente a entornos que sean definidos como potencialmente inseguros.

Para permitir a los usuarios trabajar con sus buzones de correo aunque no se encuentren conectados a un servidor de correo, la mayoría de aplicaciones de correo que trabajan con el protocolo IMAP permiten cachear los buzones y los mensajes descargados en un disco local. Esta característica posibilita la consulta del correo en condiciones de bajo ancho de banda. En esta situación se podrían consultar las cabeceras del correo y posponer la descarga de adjuntos hasta que se pueda disponer de un ancho de banda adecuado. Igualmente, permite la eliminación de correo sin necesidad de visualizar el cuerpo del mensaje, evitando así la obligación de descargar mensajes sin interés y ahorrando recursos, como el ancho de banda y tiempo.

A diferencia del protocolo POP, IMAP soporta múltiples carpetas y almacena en el servidor información de estado de los mensajes de correo (por ejemplo si el mensaje fue enviado, leído o eliminado). Esto posibilita que un usuario que acceda a su correo desde múltiples ubicaciones, incluso con diferentes clientes de correo, tendrá una vista consistente y actualizada del estado de sus mensajes.

El protocolo IMAP también soporta búsquedas en la parte del servidor, posibilitando que no sea necesario que el programa cliente de correo tenga que descargarse los mensajes para buscar uno de ellos.

IMAP es compatible con la mayoría de estándares de formatos de mensajería, por ejemplo MIME (*Multipurpose Internet Mail Extensions)*, que permite la recepción de ficheros adjuntos.

Puede consultarse documentación técnica para el protocolo IMAP en la RFC 1730 o la RFC 2060, por ejemplo.