INSTITUTO TECNOLÓGICO de León

CONFIGURACIÓN DE LOS SERVICIOS DE DHCP Y DDNS SOBRE UN SERVIDOR LINUX, PERMITIENDO LA INTERCONEXIÓN ENTRE LINUX Y WINDOWS USANDO SAMBA.

MONOGRAFÍA

Qué para obtener el Titulo de:
INGENIERO EN SISTEMAS COMPUTACIONALES.

PRESENTA:
SOTELO FIGUEROA MARCO AURELIO.

Con Asesoría de:
ING. JUAN PABLO MURILLO RUIZ

León, Guanajuato. Octubre del 2006

Índice.

Índice de Figuras.

Índice de Tablas.

A Dios por la vida y las oportunidades que me ofrece día tras día.

A mis padres y hermanos que con esmero me han proporcionado las herramientas para llegar hasta donde he llegado el día de hoy y me han guiado a ser la persona que hoy en día soy.

Al Instituto Tecnológico de León por la oportunidad de superación y a todos lo catedráticos que han marcado mi carrera profesional.

A todos mis compañeros que siempre me han apoyado cuando más lo necesito.

León, Guanajuato. Octubre del 2006.

Marco Aurelio Sotelo Figueroa.

Introducción.

Los grandes avances tecnológicos de hoy en día y el gran avance que han dado los Sistemas Operativos modernos nos permiten tener varias opciones al momento de querer implementar un servidor en nuestra empresa, además de estas opciones disponemos también de dispositivos especializados que pueden suplir dichas necesidades.

Entre todo lo que debemos de empezar a considerar antes de elegir un servidor están los costos actuales y futuros que estos nos puede acarrear, el costo de las licencias del Sistema Operativo y de los programas que debemos de adquirir para poder añadir funciones extra al servidor. También debemos de considerar si el servidor puede servir para todo lo que necesitamos o si se debería de adquirir un equipo extra para suplir dichas funciones.

En el presente trabajo se introducirá al uso y configuración de los servicios de Linux para que poder hacer uso de DHCP el cual es un método de asignación dinámica de direcciones de red, DDNS para poder hacer una asignación de un nombre genérico a una computadora sin necesidad de tener una IP fija y de Samba que permite la interconexión de información entre los sistemas Linux – Unix.

Se mostrara los orígenes del Sistema Operativo Linux, las características y funciones que lo hacen fuerte, entre lo que se encuentran: la interfaz gráfica que usa, el tipo de sistema de archivos que puede manejar y sobre los cuales puede ser instalado como los son NTFS, FAT32, FAT16, ext3, etc., los protocolos de comunicación que están implementados en el Kernel del Sistema Operativo como los son TCP/IP, ApleeTalk, etc., las distintas arquitecturas que soporta como Intel, AMD, PowerPC, etc.

Se muestra lo que es DDNS y como trabaja, empezando con la definición de DNS y las partes que están incluidas en él como son las zonas, los dominios, la forma en la que el DNS trabaja y convierte los nombres a direcciones IP y viceversa, lo que son los “Resolvers”, el “Caching” y otros conceptos básicos para la aplicación de un DNS de forma dinámica.

Se explican los orígenes del protocolo DHCP que son el RARP y BOOTP y la forma en la que estos trabajaban. Se ven las características y conceptos básicos que conservó DHCP de estos protocolos, y la forma en la que los clientes de DHCP pueden obtener una dirección IP y otros parámetros a través de un servidor DHCP.

Se muestran los orígenes de Samba y la forma en que trabaja de tal manera que una máquina con Linux instalado pueda comunicarse con otra que tenga Windows instalado y puedan compartir recursos entre dichas máquinas. Se enseñará la forma en que Microsoft tiene implementado el protocolo de NetBIOS y la forma en la que esté ultimo usa los nombres en una red como identificadores únicos para dispositivos.

Finalmente se dan los pasos necesarios para que a una máquina con Linux instalado se le pueda agregar el cliente de DDNS, montar un servidor de DHCP y Samba, de tal manera que puedan interactuar los tres servicios en una red con sistemas operativos heterogéneos.

Capítulo 1: Antecedentes.

Linux.

Historia.

Linux es un Unix libre, es decir, un Sistema Operativo (pero a diferencia de estos y otros sistemas operativos propietarios, Linux ha sido desarrollado por miles de usuarios de computadoras a través del mundo). Linux fue creado inicialmente como un pasatiempo por un estudiante joven, Linus Torvalds, de la universidad de Helsinki en Finlandia, con asistencia por un grupo de “hackers” a través de Internet. Linus tenía un interés en Minix, un sistema pequeño o abreviado del Unix (desarrollado por Andy Tanenbaum); y decidido a desarrollar un sistema que excedió los estándares de Minix. Quería llevar a cabo un sistema operativo que aprovechase la arquitectura de 32 bits para multitarea y eliminar las barreras del direccionamiento de memoria.

Linux tiene todas las prestaciones que se pueden esperar de un Unix moderno y completamente desarrollado: multitarea real, memoria virtual, bibliotecas compartidas, carga de sistemas a demanda, compartimiento, manejo de debido de la memoria y soporte de redes TCP/IP.

Linux corre principalmente en computadoras basados en procesadores 386/486/586, usando las facilidades de proceso de la familia de procesadores 386 para implementar las funciones nombradas.

La parte central de Linux (conocida como núcleo o “Kernel”) se distribuye a través de la Licencia Pública General GNU, lo que básicamente significa que puede ser copiado libremente, cambiado y distribuido, pero no es posible imponer restricciones adicionales a los productos obtenidos y, adicionalmente, se debe dejar el código fuente disponible, de la misma forma que está disponible el código de Linux. Aún cuando Linux tenga registro de Copyright, y no sea estrictamente de dominio público. La licencia tiene por objeto asegurar que Linux siga siendo gratuito y a la vez estándar.

Por su naturaleza Linux se distribuye libremente y puede ser obtenido y utilizado sin restricciones por cualquier persona, organización o empresa que así lo desee, sin necesidad de que tenga que firmar ningún documento ni inscribirse como usuario. Por todo ello, es muy difícil establecer quiénes son los principales usuarios de Linux. No obstante se sabe que actualmente Linux está siendo utilizado ampliamente en soportar servicios en Internet, lo utilizan universidades alrededor del todo el mundo para sus redes y sus clases, lo utilizan empresas productoras de equipamiento industrial para vender como software de apoyo a su maquinaría, lo utilizan cadenas de supermercados, estaciones de servicio y

muchas instituciones del gobierno y militares de varios países. Obviamente, también es utilizado por miles de usuarios en sus computadores personales. El apoyo más grande, sin duda, ha sido Internet ya que a través de ella se ha podido demostrar que se puede crear un sistema operativo para todos los usuarios sin la necesidad de fines lucrativos.

Básicamente podemos decir que hoy Linux es un sistema muy completo. El proyecto de Linus Torvalds aún no ha terminado, y se piensa que nunca se terminará debido a la continua evolución de la informática

Linux implementa la mayor parte de las características que se encuentran en otras implementaciones de Unix, más algunas otras que no son habituales. Algunas de las características que encontramos en Linux son:

- Multitarea: la palabra multitarea describe la habilidad de ejecutar varios programas al mismo tiempo, Linux utiliza la llamada multitarea preventiva, la cual asegura que todos los programas que se están utilizando en un momento dado serán ejecutados, siendo el sistema operativo el encargado de ceder tiempo de microprocesador a cada programa.
- Multiusuario: muchos usuarios pueden usar la misma máquina al mismo tiempo, esto lo hace a través de terminales y proporciona la posibilidad de que varios usuarios puedan trabajar con la misma versión de un

programa al mismo tiempo y que los cambios se actualicen inmediatamente para todos los usuarios.

- Multiplataforma: las plataformas en las que en un principio se puede utilizar Linux son 386, 486. Pentium, Pentium Pro, Pentium II, Amiga y Atari, también existen versiones para su utilización en otras plataformas, como Alpha, ARM, MIPS, PowerPC y SPARC.
- Multiprocesador: Soporte para sistemas con mas de un procesador esta disponible para Intel y SPARC.
- El núcleo de Linux ha sido desarrollado para utilizar las características del modo protegido de los microprocesadores 80386 y 80486. En concreto, hace uso de la gestión de memoria avanzada del modo protegido y otras características avanzadas.
- Protección de la memoria entre procesos, de manera que uno de ellos no pueda colgar el sistema.
- Carga de ejecutables por demanda: Linux sólo lee del disco aquellas partes de un programa que están siendo usadas actualmente.
- Política de copia en escritura para compartir páginas entre ejecutables: esto significa que varios procesos pueden usar la misma zona de memoria para ejecutarse. Cuando alguno intenta escribir en esa memoria, la página (4Kb de memoria) se copia a otro lugar. Esta política de copia en escritura tiene dos beneficios: aumenta la velocidad y reduce el uso de memoria.

- Memoria virtual usando paginación (sin intercambio de procesos completos) a disco: a una partición o un archivo en el sistema de archivos, o ambos, con la posibilidad de añadir más áreas de intercambio sobre la marcha Un total de 16 zonas de intercambio de 128Mb de tamaño máximo pueden ser usadas en un momento dado con un límite teórico de 2Gb para intercambio. Este límite se puede aumentar fácilmente con el cambio de unas cuantas líneas en el código fuente.
- La memoria se gestiona como un recurso unificado para los programas de usuario y para el caché de disco, de tal forma que toda la memoria libre puede ser usada para caché y ésta puede a su vez ser reducida cuando se ejecuten grandes programas.
- Librerías compartidas de carga dinámica y librerías estáticas.
- Se realizan volcados de estado para posibilitar los análisis post-mortem, permitiendo el uso de depuradores sobre los programas no sólo en ejecución sino también tras abortar éstos por cualquier motivo.
- El sistema Linux es compatible con ciertos estándares de Unix a nivel de código fuente, incluyendo el IEEE POSIX.1, System V y BSD. Fue desarrollado buscando la portabilidad de las fuentes: encontrará que casi todo el software gratuito desarrollado para UNIX se compila en Linux sin problemas.
- Emulación de 387 en el núcleo, de tal forma que los programas no tengan que hacer su propia emulación matemática. Cualquier máquina que

ejecute Linux parecerá dotada de coprocesador matemático. Por supuesto, si el ordenador ya tiene una unidad de coma flotante, esta será usada en lugar de la emulación, pudiendo incluso compilar tu propio "Kernel" sin la emulación matemática y conseguir un pequeño ahorro de memoria.

- Consolas virtuales múltiples: varias sesiones de acceso a través de la consola entre las que se puede cambiar con las combinaciones adecuadas de teclas (totalmente independiente del hardware de video). Se crean dinámicamente y puedes tener hasta 64.
- Soporte para varios sistemas de archivo comunes, incluyendo minix-1, Xenix y todos los sistemas de archivo típicos de System V, y tiene un avanzado sistema de archivos propio con una capacidad de hasta 4 Tb y nombres de archivos de hasta 255 caracteres de longitud.
- Acceso transparente a particiones MS-DOS (o a particiones OS/2 FAT) mediante un sistema de archivos especial: no es necesario ningún comando especial para usar la partición MS-DOS, esta parece un sistema de archivos normal de Unix (excepto por algunas restricciones en los nombres de archivo, permisos, y esas cosas). Las particiones comprimidas de MS-DOS 6 no son accesibles. El soporte para VFAT (Windows NT, Windows 95) ha sido añadido al núcleo de desarrollo.
- Un sistema de archivos especial llamado UMSDOS que permite que Linux sea instalado en un sistema de archivos DOS.

- Implementación de todo lo necesario para trabajar en red con TCP/IP. Desde manejadores para las tarjetas de red más populares hasta SLIP/PPP, que permiten acceder a una red TCP/IP por el puerto serie. También se implementan PLIP (para comunicarse por el puerto de la impresora) y NFS (para acceso remoto a ficheros). Y también se han portado los clientes de TCP/IP, como FTP, Telnet, NNTP y SMTP.

Distribuciones.

Una distribución Linux, o distribución GNU/Linux (abreviada con frecuencia "distro") es un conjunto de aplicaciones reunidas que permiten brindar mejoras para instalar fácilmente un sistema Linux. Son variaciones de Linux que, en general, se destacan por las herramientas para configuración y sistemas de paquetes de software a instalar.

Existen numerosas distribuciones Linux. Cada una de ellas puede incluir cualquier número de software adicional (libre o no), como algunos que facilitan la instalación del sistema y una enorme variedad de aplicaciones, entre ellos, entornos gráficos, suites ofimáticas, servidores Web, servidores de correo, servidores FTP, etcétera.

La base de cada distribución incluye el núcleo de Linux, con las bibliotecas y herramientas del proyecto GNU y de muchos otros proyectos/grupos de software, como BSD.

Usualmente se utiliza la plataforma XFree86 o la Xorg para sostener interfaces gráficas (esta última es un modificación de XFree86, surgido a raíz del cambio de licencia que este proyecto sufrió en la versión 4.4 y que lo hacía incompatible con la GPL).

Ubuntu.

El 8 de julio de 2004, Mark Shuttleworth y la empresa Canonical Ltda anunciaron la creación de la distribución Ubuntu. Ésta tuvo una financiación inicial de 10 millones de dólares. El proyecto nació por iniciativa de algunos programadores de los proyectos Debian, Gnome y Arch que se encontraban decepcionados con la manera de operar del proyecto Debian, la distribución Linux sin ánimo de lucro más popular del mundo.

De acuerdo con sus fundadores, Debian era un proyecto demasiado burocrático donde no existían responsabilidades definidas y donde cualquier propuesta interesante se ahogaba en un mar de discusiones. Asimismo, Debian no ponía

énfasis en estabilizar el desarrollo de sus versiones de prueba y sólo proporcionaba auditorias de seguridad a su versión estable, la cual era utilizada sólo por una minoría debido a la poca o nula vigencia que poseía en términos de la tecnología Linux actual.

Tras formar un grupo multidisciplinario, los programadores decidieron buscar el apoyo económico de Mark Shuttleworth, un emprendedor sudafricano que tras fundar la compañía Thawte en la cochera de su domicilio, logró venderla cuatro años después a la empresa VeriSign por 575 millones de dólares estadounidenses.

Shuttleworth vio con simpatía el proyecto y decidió convertirlo en una iniciativa autosostenible, combinando su experiencia en la creación de nuevas empresas con el talento y la experiencia de los programadores de la plataforma Linux. De esta forma nació la empresa Canonical, la cual se encarga de sostener económicamente el proyecto mediante la comercialización de servicios y soporte técnico a otras empresas. Mientras los programadores armaban el sistema, Shuttleworth aprovechó la ocasión para aplicar una pequeña campaña de mercadotecnia para despertar interés en "la distribución-sin-nombre" (en inglés: the no-name-distro).

Tras varios meses de trabajo y un breve período de pruebas, la primera versión de Ubuntu “Warty Warthog” fue lanzada el 20 de octubre de 2004.

Algunas características que Ubuntu incorpora:

- Esta basada en la distribución Debian.
- Disponible en 3 arquitecturas: Intel x86, AMD64, PowerPC.
- Ubuntu usa como base el escritorio Gnome, pero existe XUbuntu que se basa en el escritorio XFce, y KUbuntu que usa como base el escritorio KDE.
- El sistema incluye funciones avanzadas de seguridad y entre sus políticas se encuentra el no activar procesos latentes por omisión al momento de instalarse. Por eso mismo, no hay un Firewall predeterminado, ya que no existen servicios que puedan atentar a la seguridad del sistema.
- Para labores/tareas administrativas incluye una herramienta llamada “sudo” (similar al MacOS X), con la que se evita el uso del usuario “root”.
- Mejorar la accesibilidad y la internacionalización, de modo que el software esté disponible para tanta gente como sea posible. En la versión 5.04, el UTF-8 es la codificación de caracteres por defecto.
- No sólo tiene como lazo a Debian el uso del mismo formato de paquetes “deb”, Ubuntu tiene uniones muy fuertes con esa comunidad, contribuyendo cualquier cambio directamente e inmediatamente, más que anunciándolos. Esto sucede en los tiempos de lanzamiento. Muchos de

los desarrolladores de Ubuntu son también responsables de los paquetes importantes dentro de la distribución de Debian.

Ubuntu divide todo el software en cuatro secciones, llamadas los "componentes", para reflejar diferencias en licencias y la prioridad con la que se atienden los problemas que informen los usuarios. Por defecto, se instala una selección de paquetes que cubre las necesidades básicas de la mayoría de los usuarios de computadoras. Los paquetes de Ubuntu generalmente se basan en los paquetes de la rama estable “Sid” de Debian, las cuatro secciones son:

- main: esta sección contiene solamente los paquetes que cumplen los requisitos de la licencia de Ubuntu, y para los que hay soporte disponible por parte de su equipo. Éste está pensado para que incluya todo lo necesario para la mayoría de los sistemas Linux de uso general. Los paquetes de esta sección poseen ayuda técnica garantizada y mejoras de seguridad oportunas.
- restricted: contiene el software que está soportado por los desarrolladores de Ubuntu debido a su importancia, pero que no está disponible bajo ningún tipo de licencia libre para incluir en main. En este lugar se incluyen los paquetes tales como los controladores propietarios de algunas tarjetas gráficas, como por ejemplo, los de nVIDIA. El nivel de la ayuda es más limitado que para la sección “main”, puesto que los desarrolladores pueden no tener acceso al código fuente.

- universe: esta sección contiene una amplia gama del software, que puede o no tener una licencia restrictiva, pero que no recibe apoyo por parte del equipo de Ubuntu. Esto permite que los usuarios instalen toda clase de programas en el sistema, pero los guarda en un lugar aparte de los paquetes soportados por las secciones "main" y "restricted".
- multiverse: contiene todos los paquetes sin soporte debido a que no cumplen los requisitos del Software Libre.

Capítulo 2: DDNS.

DNS (Domain Name System)

Introducción.

A mediados de los 70, ARPANET era una comunidad pequeña y amistosa formada por unos cientos de máquinas. Un solo archivo, “hosts.txt”, contenía toda la información que se necesitaba saber sobre esas máquinas: contenía un mapeo de nombre a dirección para cada máquina en ARPANET.

El archivo “/etc/hosts” se obtenía entonces del archivo “hosts.txt” (el cual tenía información adicional que no era necesaria).

Sin embargo, cuando ARPANET se cambió a los protocolos TCP/IP, la población de la red explotó, y con ello se presentaron los siguientes problemas:
La carga y el tráfico de red para la máquina que contenía las tablas que hacían posible el mapeo de nombres a direcciones IP se volvió inmanejable.
Colisiones de nombres. El NIC (organismo encargado de administrar la creación de nombres) no podía garantizar que alguien asignara el mismo nombre a máquinas distintas (Esto se debe a que con el método del “hosts.txt” se tenía un dominio plano, es decir, sin jerarquías).

Consistencia: Mantener la consistencia del archivo a lo largo de una red en crecimiento se hacía cada vez más difícil (Por ejemplo, cuando el archivo HOSTS.TXT llegaba a una máquina muy lejana ya era obsoleto).

Este método se volvía ineficiente a medida que aumentaban el número de máquinas. Es aquí donde entra el DNS (Domain Name System, Sistema de Dominios de Nombres). DNS es una base de datos distribuida. DNS permite un control local sobre los segmentos de la base de datos general, logrando que cada segmento esté disponible a lo largo de toda la red utilizando un esquema cliente servidor. La robustez y un desempeño adecuado se lograban gracias a la duplicidad de servidores y el “caching” (almacenamiento temporal).

Los programas llamados servidores de nombres (“name servers”) comprenden la mitad del mecanismo cliente - servidor de DNS. Los servidores de nombres contienen información acerca de un segmento de la base de datos y la ponen a disposición de los clientes, llamados “resolvers”.

Los “resolvers” son solo biblioteca de rutinas que crean preguntas y las envían a lo largo de la red hacia el servidor de nombres.

La estructura de una base de datos DNS se asemeja mucho a un árbol invertido (Es decir, un árbol cuyo tronco está hacia arriba y no abajo, como es común). Cada hoja o nodo de ese árbol es un dominio, comenzando todos los dominios desde el dominio principal o raíz (el cual se denota con un punto). Cada uno de esos dominios a su vez puede subdividirse en más subdominios. Existen muchos dominios (com, edu, gov) y debajo de ellos hay aun más subdominios.

Por ejemplo, una máquina llamada bach dentro del dominio ing.ula.ve tiene el nombre unívoco (llamado oficialmente dominio completamente calificado) “bach.ing.ula.ve” (FQDN, Full Qualified Domain Name) debido a que incluye tanto el nombre de la máquina como al dominio al cual pertenece.

La dirección completa de la máquina se lee de izquierda a derecha (desde lo más específico, el nombre de la maquina pasando por cada uno de los dominios a los cuales pertenecen).

Cada máquina en la red pertenece a un dominio, cuyo servidor de nombres contiene la información acerca de la máquina. Esta información puede incluir direcciones IP, información acerca de enrutamiento de correo, etc. (una máquina también puede tener uno o más alias de dominio, lo cual quiere decir que existen 2 referencias hacia la máquina, una de ellas es un apuntador de un dominio (alias) a su nombre canónico (u oficial).

El DNS con su estructura (aparentemente complicada) permite eliminar los problemas que presentaba la existencia de un archivo de datos planos:

Elimina el problema de nombres repetidos (a cada organización se le asigna un dominio único, por lo que pueden existir dos máquinas con el mismo nombre mientras estén en dominios separados).

Elimina el problema de carga y tráfico de red en una sola máquina ya que la información está distribuida y está disponible de manera redundante.

Hay consistencia, ya que la actualización de la información se hace de manera automática, sin intervención del administrador de la red (al menos no de la manera tradicional).

Diferencias entre el DNS y el sistema de Archivos en Linux (LFS).

La estructura de la base de datos DNS, como podemos ver en la Figura 2-1, es muy similar a la estructura del sistema de ficheros Linux. La base datos completa (o el sistema de ficheros) lo podemos ver como un árbol invertido, con el nodo raíz en lo alto del árbol. Cada nodo en el árbol tiene una etiqueta de texto, la cual identifica al nodo relativo a su padre. Es similar a una ruta relativa en el sistema de ficheros. Existe una etiqueta reservada para el nodo raíz que es la etiqueta “null” o “”.El sistema de ficheros Linux, la raíz se representa con una barra (“/”).

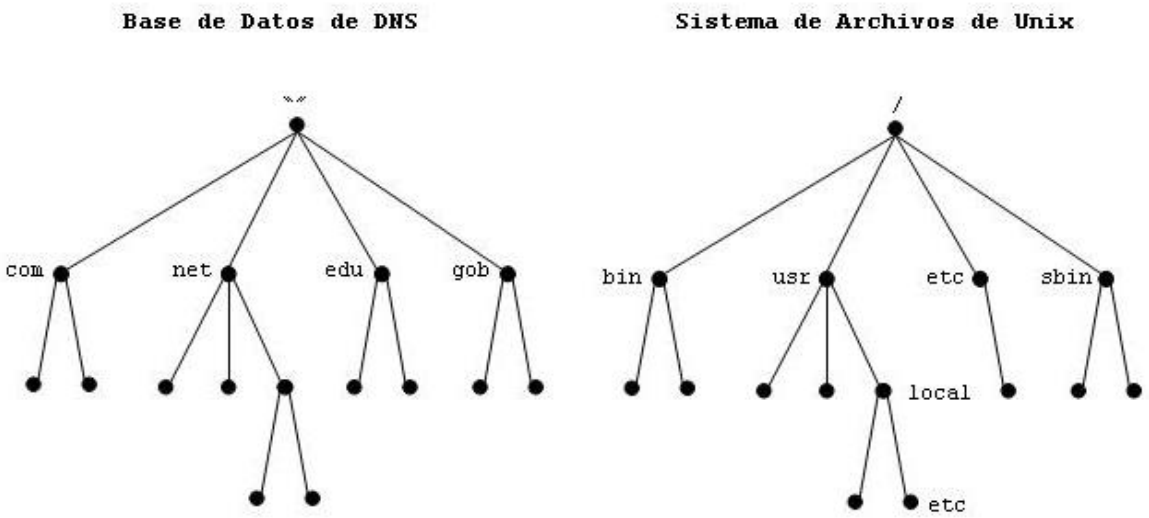


Figura 2-1: Base de datos DNS y el Sistema de ficheros Linux

Cada nodo es también la raíz de un nuevo subárbol dentro del árbol completo. Cada uno de estos subárboles representa una partición dentro de la base de datos. Es equivalente a un directorio en el sistema de ficheros Linux. Cada dominio o directorio puede ser a su vez subdividido en particiones adicionales, llamados subdominios en DNS o subdirectorios en el sistema de ficheros Linux. Los subdominios (igual que los subdirectorios), se ven como hijos dentro del árbol que cuelgan de los dominios padre.

Cada dominio tiene un nombre único (como cada directorio). El nombre de dominio de un dominio identifica la posición en la base de datos, parecido a una ruta absoluta específica su posición en el sistema de archivos. En DNS, el nombre de dominio es la secuencia de etiquetas desde el nodo raíz del dominio

hasta el nodo raíz del árbol separado por etiquetas “.”. En el sistema de ficheros Linux la ruta de un directorio absoluto es una lista de nombres relativos desde la raíz hasta el directorio especificado (se representa en dirección inversa al DNS), utilizando una barra como separador de nombres.

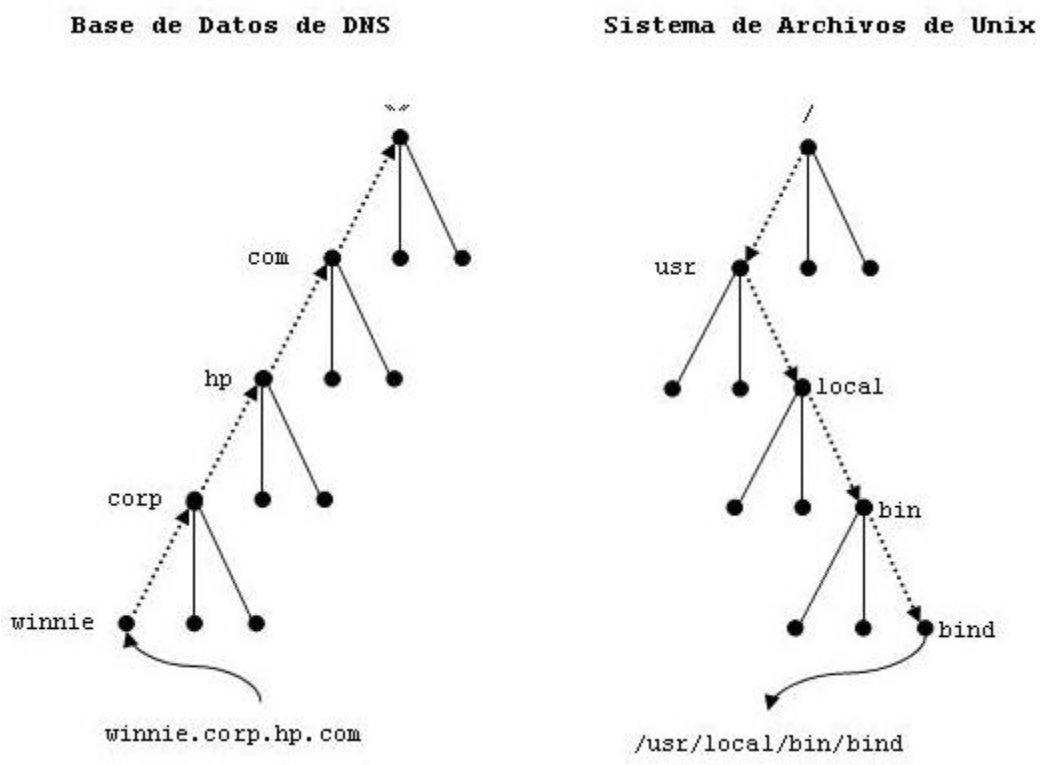


Figura 2-2: nombres en DNS y Sistema de ficheros Linux

En DNS, cada dominio puede ser administrado por una organización diferente. Cada organización puede entonces romper su dominio en un número de subdominios y delegar la responsabilidad para estos dominios a otras organizaciones. Por ejemplo, InterNIC gestiona el dominio “edu” (educacional),

pero delega la responsabilidad de gestionar el subdominio “berkeley.edu” a U.C Berkeley (ver Figura 2-3). Esto es parecido a montar de forma remota un sistema de ficheros. Algunos directorios en un sistema de ficheros pueden ser sistemas de ficheros en otra máquina (en la Figura 2-3 muestra como el sistema de ficheros que aparece en “/usr/nfs/winken” en realidad es un sistema de ficheros montado remotamente).

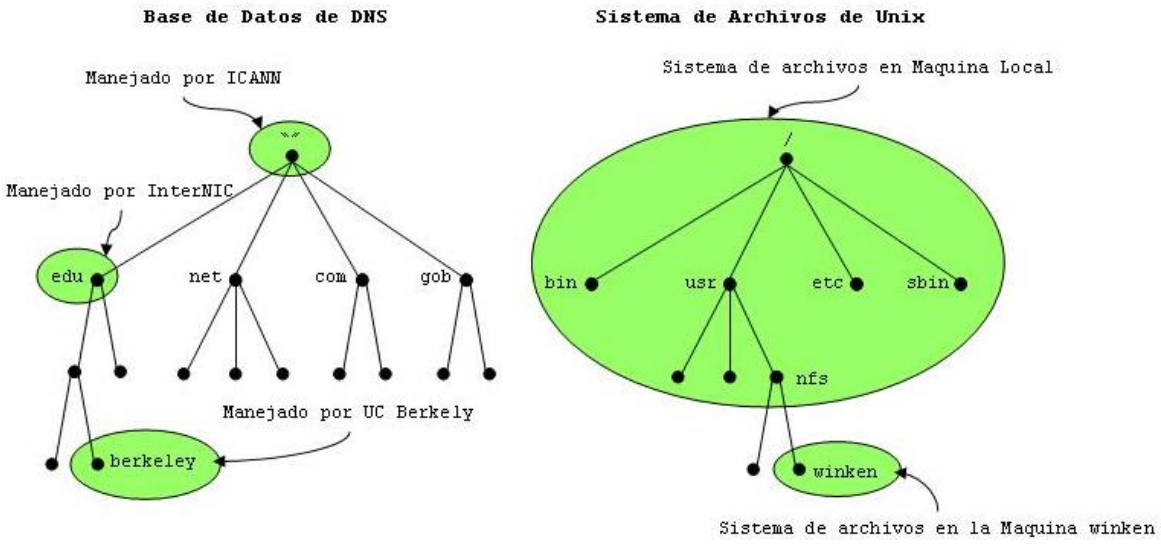


Figura 2-3: Gestión remota de subdominios y sistemas de ficheros

Los nombres de dominio son utilizados como índices a la base datos DNS. Podríamos pensar en los datos del DNS como adjuntados a un nombre de dominio. En un sistema de ficheros, los directorios contienen ficheros y subdirectorios. Igualmente, los dominios pueden contener tantas máquinas como subdominios. Un dominio contiene las máquinas y subdominios cuyos nombre de

dominio están incluidos en el dominio (“www.yahoo.com” es una máquina dentro del dominio yahoo.com y “subdominio.yahoo.com” puede ser un subdominio dentro del dominio “yahoo.com”).

Cada máquina en una red tiene un nombre de dominio, el cual apunta a la información sobre la máquina (ver Figura 2-4). Esta información puede incluir la dirección IP, información sobre enrutamiento de e-mail, etc. Las máquinas pueden también tener uno o más alias a nombres de dominio, son simplemente apuntadores desde un nombre de dominio (el alias) a otro (el nombre de dominio oficial o nombre canónico). En la Figura 2-4 “mailhub.nv” es un alias para el nombre canónico “rincon.ba.ca”.

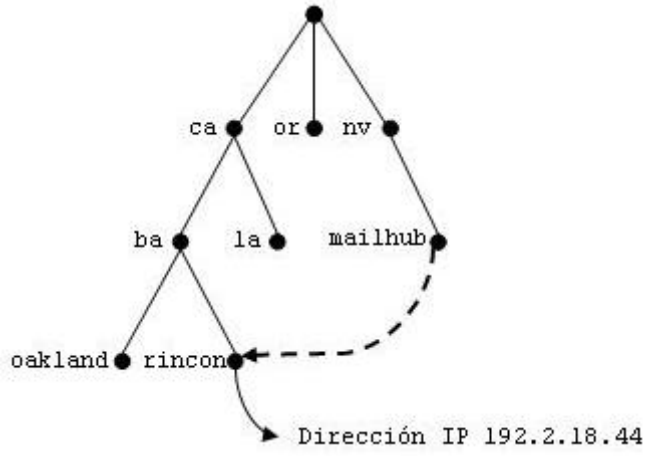


Figura 2-4: Alias apuntando a un nombre canónico

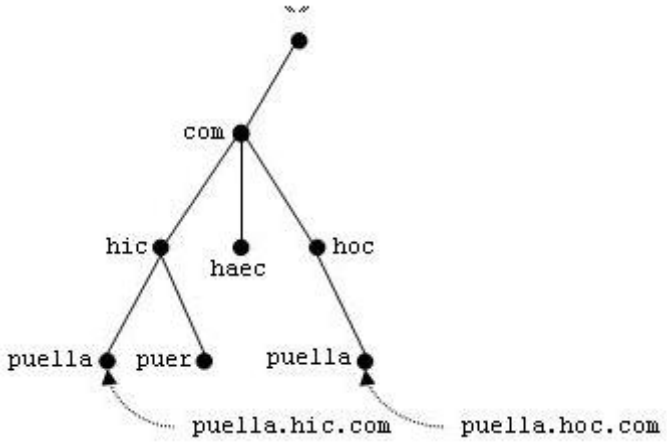


Figura 2-5: Resolviendo el problema de la colisión de nombres

¿Porqué una estructura tan complicada? Para resolver los problemas que tenía “hosts.txt”. Por ejemplo, haciendo una estructura jerárquica de nombres de dominio, elimina el problema de la colisión de nombres. Cada dominio tiene un nombre único de dominio, de manera que la organización que gestiona el dominio es libre poner los nombres que quiera a las maquinas y subdominios dentro de su dominio. Cualquier nombre que se escoja para una máquina o subdominio no tendrá conflictos con los nombres de dominio de otra organización debido a que éstos finalizan con un nombre de dominio único (ver Figura 2-5).

Como trabaja el DNS.

El sistema de nombres de dominio es básicamente una base de datos con información de las máquinas.

Espacio de Nombres de Dominio.

La base de datos distribuida de DNS está indexada por nombres de dominio. Cada nombre de dominio es esencialmente una dirección en un gran árbol invertido, llamado espacio de nombre de dominio. En la Figura 2-6 podemos ver una aproximación a la estructura del espacio de nombres de dominio.

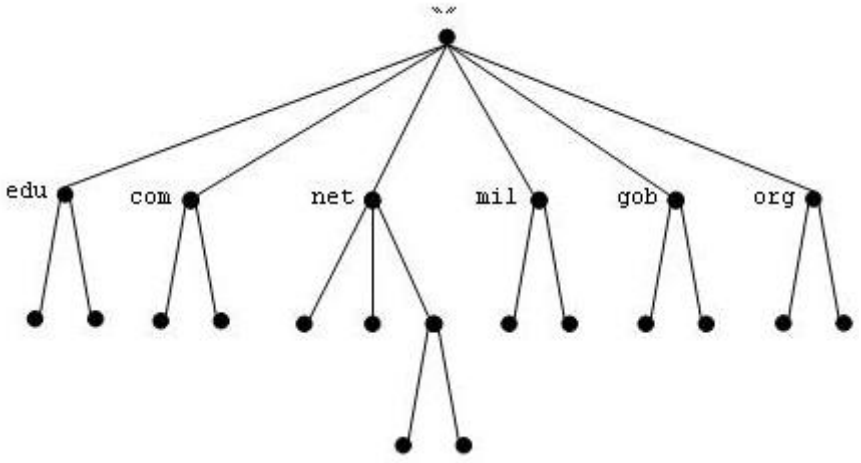


Figura 2-6: Árbol invertido del sistema DNS

Nombres de Dominio.

Cada nodo en el árbol tiene una etiqueta de texto (sin puntos). La etiqueta “null” está reservada para la raíz.

Cuando aparece un punto al final del nombre de dominio estamos especificando un nombre de dominio absoluto. También podemos hablar de nombre absoluto como Fully Qualified Domain Name (FQDN). Los nombres de dominio que no acaban en un punto “.” son interpretados como nombres relativos a otro dominio.

Dominios.

Un dominio es simplemente un subárbol en el espacio de nombres de dominio. El nombre de dominio de un dominio es el mismo que el nombre de dominio situado en el nodo superior en el dominio. Por ejemplo, el nodo del dominio “purdue.edu” es un nodo llamado “purdue.edu” (ver Figura 2-7)

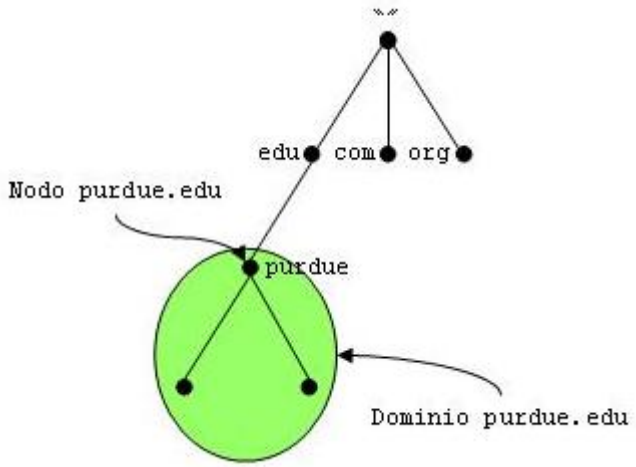


Figura 2-7: Dominio purdue.com

Cualquier nombre de dominio en el subárbol es considerado parte del dominio. Debido a esto un nombre de dominio puede tener múltiples subárboles, un nombre de dominio puede estar también en múltiples dominios. Por ejemplo el nombre de dominio “pa.ca.us” es parte del dominio “ca.us” y forma parte también del dominio “us” (ver Figura 2-8).

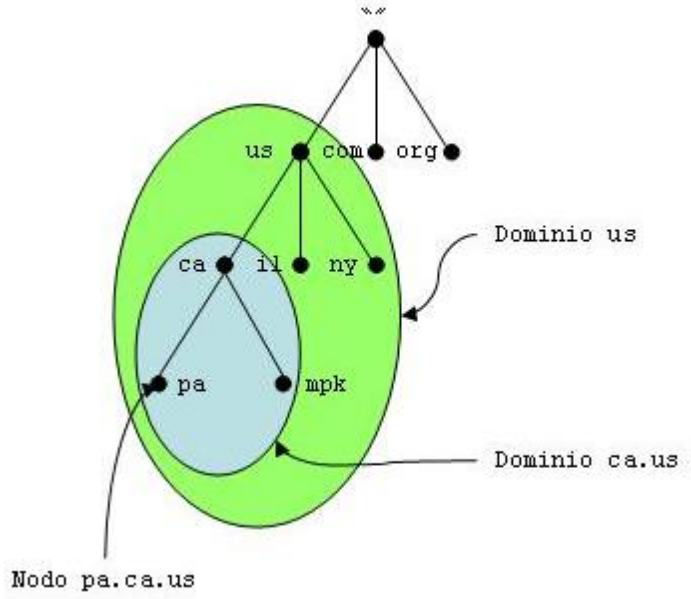


Figura 2-8: Un nodo en múltiples dominios

Por lo tanto, un dominio es justamente un subárbol en el espacio de nombres de dominio.

Las máquinas son representaciones de nombres de dominio. Los nombres de dominio de máquinas apuntan a información sobre máquinas individuales. Y un dominio contiene todas las máquinas cuyos nombres de dominio están dentro del dominio.

Los nombres de dominio situados en las hojas del árbol representando el espacio de nombres de dominio, representan máquinas individuales y apuntan a direcciones de red, información hardware, información de rutado de mail, etc.

Una forma simple de saber cuando un dominio es un subdominio de otro dominio es comparar sus nombres de dominio. Un nombre de dominio de un subdominio finaliza con el nombre de dominio del dominio padre. Por ejemplo, “la.tyrell.com” debe ser un subdominio de “tyrell.com” porque “la.tyrell.com” finaliza con “tyrell.com”.

Los subdominios de otros dominios son normalmente referenciados por su nivel dentro del árbol DNS (dentro del espacio de nombres de dominio). Podemos diferenciar entre:

- “Top Level domain” (TLD) o dominio de primer nivel: Es un dominio hijo de la raíz del árbol DNS.
- Un dominio de segundo nivel: Es un dominio hijo de un dominio de primer nivel o TLD.

El Espacio de Nombre de Dominio en Internet

Los dominios de primer nivel o “Top Level Domains” dividen el espacio de nombres de dominio de Internet de forma organizativa en varios dominios entre los cuales destacan: “com”, “edu”, “gov”, “mil”, “net”, “org”, “int”.

Actualmente estos dominios originales son llamados “generic top-level domains” o gTLDs.

Con la internacionalización de Internet se decidió crear dominios geográficos. Fueron reservados nuevos TLDs que correspondían a países individuales. Estos nombres de dominio siguen el estándar ISO 3166 que establece dos letras para cada dominio TLD de cada país.

Delegación de dominios

Uno de los principales objetivos del sistema de nombres de dominio es descentralizar la administración, esto se consigue gracias a la delegación de dominios. La delegación de dominios funciona cuando un administrador divide en subdominios el dominio y cada uno de estos subdominios es delegado a otra organización. Esto significa que una organización se convierte en la responsable de mantener todos los datos de ese subdominio y puede cambiar libremente los datos o incluso dividir el subdominio en nuevos subdominios y delegarlos a su vez.

El dominio padre contiene solo apuntadores a la localización de los subdominios de manera que puede realizar peticiones sobre estos. Por ejemplo, el dominio “stanford.edu” está delegado a las personas en Stanford que se encargan de gestionar la red de la universidad (ver Figura 2-9).

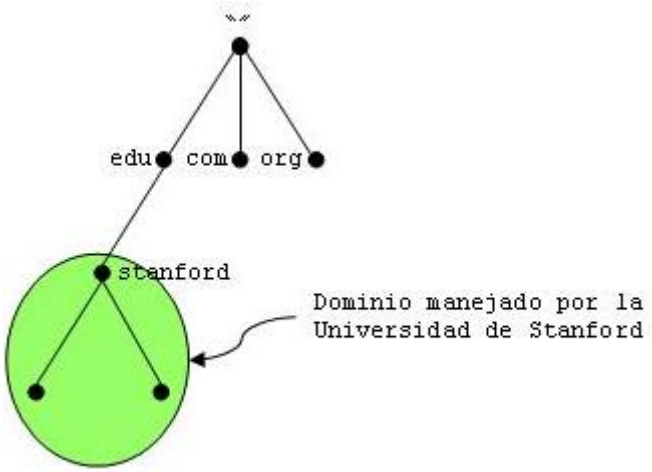


Figura 2-9: Dominio stanford.edu delegado a la universidad de Stanford

No todas las organizaciones delegan su dominio completo igual que no todos los administradores delegan todo su trabajo. Un dominio puede tener diferentes subdominios y contener máquinas que no pertenezcan a ninguno de estos subdominios.

Un dominio podría contener más información de la que el servidor de nombres necesita. Un domino podría contener datos delegados a otros servidores de nombres. Ya que las zonas están unidas por delegación, éstas nunca deben contener datos que han sido delegados.

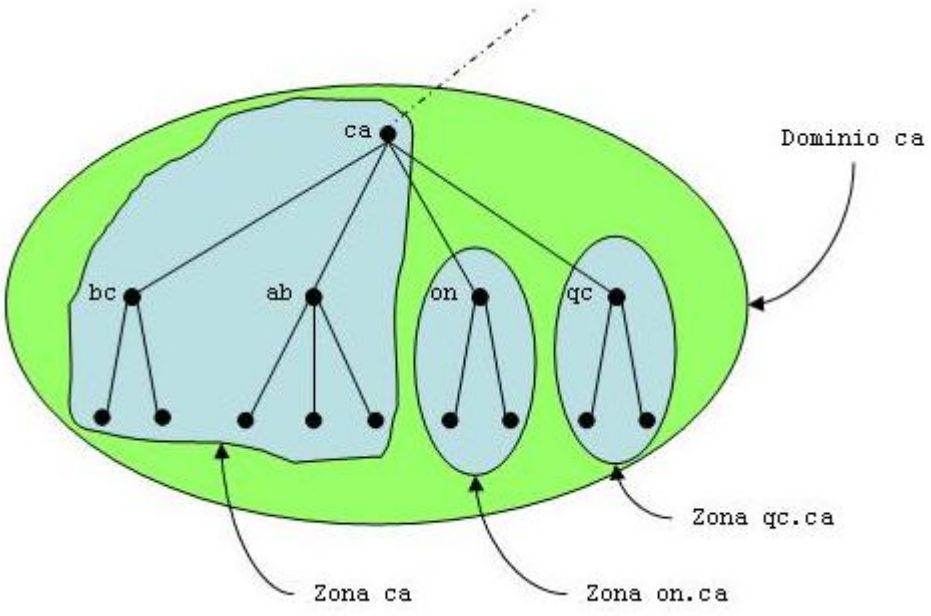


Figura 2-10: Zona y Dominio.

Tipos de servidores de nombres

La especificación DNS define dos tipos de servidores de nombres: maestro primario y maestro secundario.

Un servidor de nombres maestro primario para una zona, lee los datos de la zona desde un fichero en la máquina.
Un servidor de nombres maestro secundario para una zona, recoge los datos desde un servidor de nombres que es autoritativo para la zona, este servidor autoritativo es llamado “Servidor de Nombres Maestro”. Normalmente, el servidor maestro se corresponde con el servidor de nombres maestro primario para la zona, pero no es obligatorio, ya que un maestro secundario puede cargar la zona de otro servidor maestro secundario.

Cuando un maestro secundario se inicia, contacta su servidor de nombres maestro y si es necesario vuelca la información de la zona. Esto se denomina transferencia de zona.

Tanto el maestro primario como el maestro secundario para una zona son autoritativos para la zona y un mismo servidor de nombres puede ser maestro primario para unas zonas y maestro secundario para otras.

Ficheros de datos

Los ficheros desde donde el servidor de nombres maestro primario carga su zona “es” llamado ficheros de datos de zona, zona o simplemente archivos de datos.

Un servidor de nombres maestro secundario también puede cargar su zona de ficheros de datos. Normalmente los servidores maestros secundarios se configuran para que hagan respaldo de la zona que transfieren desde el maestro primario en ficheros de datos. Si el maestro secundario es reiniciado, cuando arranca primero leerá su archivo de respaldo y entonces comprueba si el archivo está actualizado o tiene que volver a hacer una transferencia desde el servidor primario.

La Base de Datos DNS

Cada unidad de información del DNS se llama Registro de Recurso (RR). Cada registro tiene un tipo asociado que describe el dato que contiene, y una clase que especifica el tipo de red al que se aplica. Esto último se adapta a diferentes esquemas de dirección, como direcciones IP (la clase IN), direcciones Hesiod (utilizadas por el sistema Kerberos del MIT) y algunas más. El RR típico es el

registro A, que asocia un nombre completamente cualificado con una dirección IP.

Un nodo puede ser conocido por más de un nombre. Por ejemplo, podemos tener un servidor que proporciona tanto servicio FTP como WWW, y tendrá dos nombres: “ftp.máquinas.org” y “www.máquinas.org”. Sin embargo, uno de estos nombres debe ser identificado como oficial o canónico. La diferencia es que el canónico es el único registro A que debe existir apuntando a esa dirección IP, mientras que el resto de los nombres deben ser alias (registros CNAME), que apuntan al nombre canónico.

Los datos asociados con nombres de dominios (los Resource Records o RRs) deben ser guardados de alguna manera en los servidores de nombres. La forma utilizada por Bind es guardarlos en archivos de texto simple.

Algunos de estos registros son:

- NS (Name Server): especifican qué máquinas son servidores de nombres
- MX (Mail Exchangers): especifican qué máquinas intercambian correos
- PTR (Pointer): permiten la conversión de una dirección IP a nombre
- A (Address): permiten la conversión de un nombre a dirección IP
- CNAME (Canonical Name): se utilizan para hacer un alias

La forma como se cargan los datos también depende mucho del servidor. Si el servidor es un servidor primario guarda su información en archivos de bases de datos. Los archivos de datos contienen registros fuentes que describen la zona y manejan la delegación de subdominios.

En el caso de un servidor secundario estos tienen sólo los archivos de configuración básicos y toman sus datos del servidor primario (es decir hacen una transferencia de zona).

Resolvers

Los “resolvers” son los clientes que acceden a los servidores de nombres. Los programas ejecutándose en una máquina que necesitan información del espacio de nombres de dominio utilizan “resolvers” para obtener dicha información. Un resolver gestiona:

- Peticiones a un servidor de nombres
- Interpretación de las respuestas (pueden ser RRs o errores)
- Devuelve la información a los programas que la solicitaron

Resolución

Los servidores de nombres deben recuperar información del espacio de nombres de dominio, tanto para los dominios que son autoritativos como para los que no. Este proceso se llama resolución de nombres o simplemente resolución.

Debido a que el espacio de nombres está estructurado en un árbol invertido, un servidor de nombres sólo necesita:

- Nombres de dominio
- Direcciones de los servidores de nombres raíz (root name servers)

Un servidor de nombres puede lanzar una petición a un servidor de nombres raíz para cualquier nombre el espacio de nombres de dominio, y el servidor de nombres raíz devolverá el servidor de nombres destino (encargado de gestionar la zona para el dominio solicitado) al que se debe enviar la petición.

Root Name Servers (RNS)

El servidor de nombres raíz conoce donde están los servidores de nombres autoritativos para cada uno de los TLDs (Top Level Domains). Dada una petición sobre cualquier nombre de dominio, el servidor de nombres raíz puede como

mínimo proporcionar los nombres y direcciones de los servidores de nombre que son autoritativos para el TLD en el que está ese dominio. Y los TLD pueden proporcionar la lista de servidores de nombres que son autoritativos para el dominio de segundo nivel en el que está el nombre de dominio solicitado.

Cada servidor de nombres consultado proporciona al servidor de nombres solicitante la información sobre el servidor de nombres más cercano al que hay que contactar para obtener la respuesta (excepto el último servidor de nombres que será el que proporcione la respuesta)

Los servidores de nombre raíz son muy importantes para que la resolución tenga éxito. Por este motivo el sistema DNS proporciona mecanismos como “caching” para ayudar a descargar de trabajo a estos servidores raíz. Pero siempre que no se pueda recurrir al “caching” serán los servidores raíz los encargados de iniciar la búsqueda de los servidores de nombre que son capaces de contestar a la petición realizada.

Si todos los servidores de nombres raíz dejasen de funcionar durante un periodo de tiempo, todo el sistema de resolución de nombres en Internet fallaría. Para protegerse de este problema, existen al menos 13 servidores de nombres raíz.

La Figura 2-11 muestra un ejemplo de como se lleva a cabo la resolución de un nombre de dominio “girigiri.gbrmpa.gov.au”.

El servidor de nombres local hace una petición a un servidor de nombres raíz para la dirección de girigiri.gbrmpa.gov.au y este le referencia a los servidores de nombre del dominio geográfico “au”. El servidor de nombres local hace la misma pregunta a uno de los servidores de nombres de “au” y este le referencia a los servidores de nombres del dominio “gov.au”. El servidor de nombres de “gov.au” referencia a su vez a los servidores de nombres del dominio “gbrmpa.gov.au”. Finalmente el servidor local hace la solicitud a uno de los servidores de nombres de “gbrmpa.gov.au” y éste le devuelve la respuesta.

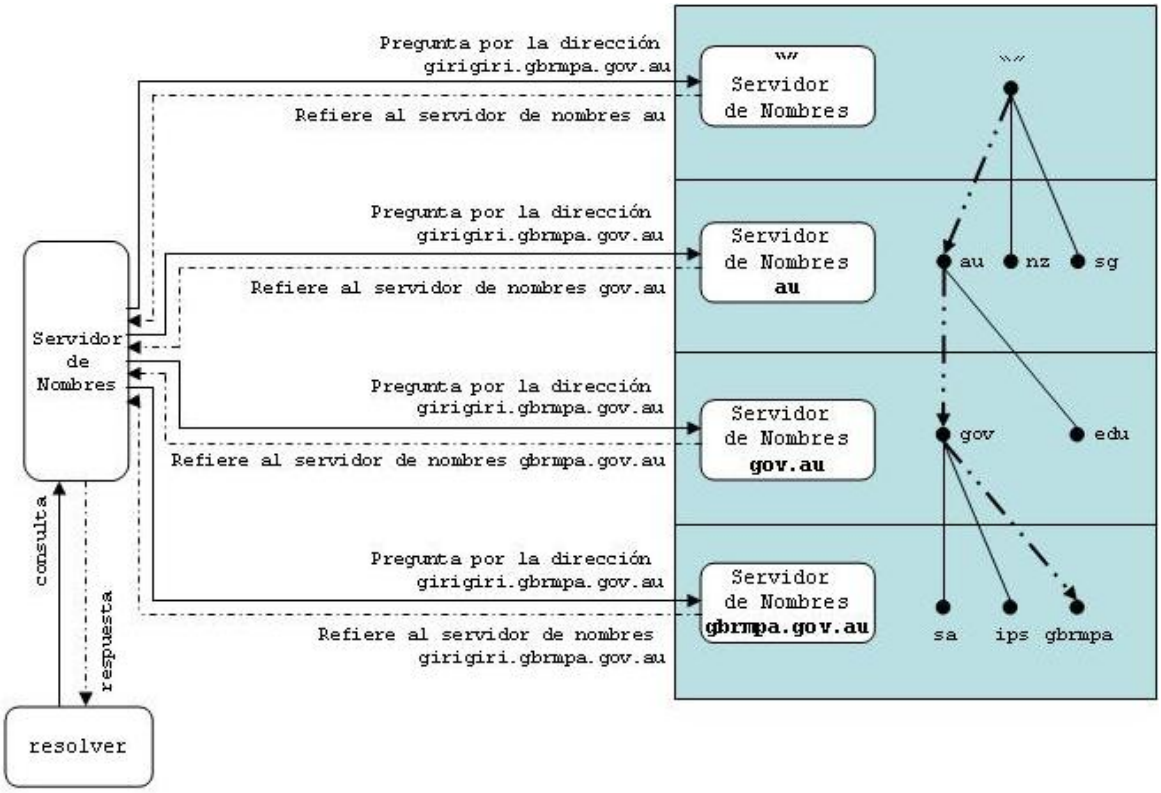


Figura 2-11: Proceso de resolución de un nombre de dominio

Recursividad

¿Por qué el servidor de nombres local no puede simplemente devolver al resolver la referencia al siguiente servidor de nombres?

Esto se debe a que el “resolver” no debía tener la suficiente inteligencia como para seguir las referencias a otros servidores de nombres y por lo tanto hace una petición recursiva a su servidor de nombres local.

Las peticiones pueden ser recursivas o iterativas (o no recursivas), una petición recursiva pone la mayor parte del peso de la resolución en un único servidor de nombres mientras que una petición iterativa hace referencia al proceso de resolución utilizado por un servidor de nombres que recibe peticiones de forma iterativa.

En el proceso recursivo un resolver envía una petición recursiva al servidor de nombres para resolver un nombre de dominio. El servidor de nombres está obligado a responder con los datos solicitados o bien con un error. El servidor de nombres no podrá referenciar al solicitante para que realice la petición a otro servidor de nombres.

Si el servidor de nombres no es autoritativo para los datos solicitados, entonces tendrá que hacer la petición a otros servidores de nombre para encontrar la respuesta. Podrá enviar peticiones recursivas a estos servidores, de manera que entonces estos servidores de nombres tendrán de encontrar la respuesta y retornarla al servidor de nombres solicitante. O bien podrá enviar peticiones iterativas (es el caso del ejemplo anterior donde el servidor de nombre local envía peticiones iterativas, y el resolver envía una petición recursiva) y entonces ser referenciado a otros servidores de nombres hasta llegar al servidor que tiene la información solicitada.

Las implementaciones existentes normalmente hacen la segunda opción en que se hacen peticiones iterativas desde el servidor de nombres local hacía los servidores de nombres intermedios.

Un servidor de nombres que recibe una petición recursiva para la que no puede contestar directamente lo que hará será hacer una petición a los servidores de nombres más cercanos. Estos servidores serán los servidores de nombres autoritativos para la zona más cercana al nombre de dominio que se ha solicitado.

Iteración

En el proceso de resolución por iteración un servidor de nombres simplemente retorna al solicitante la mejor respuesta que conoce sobre el dominio solicitado. El servidor de nombres consulta sus datos locales (incluyendo el caché) y busca la información solicitada. Si no la encuentra devuelve al solicitante la mejor información para que este pueda continuar el proceso de resolución. Normalmente esta información será nombres de dominio y direcciones de los servidores de nombres que conoce más cercanos a la información solicitada (ver Figura 2-12).

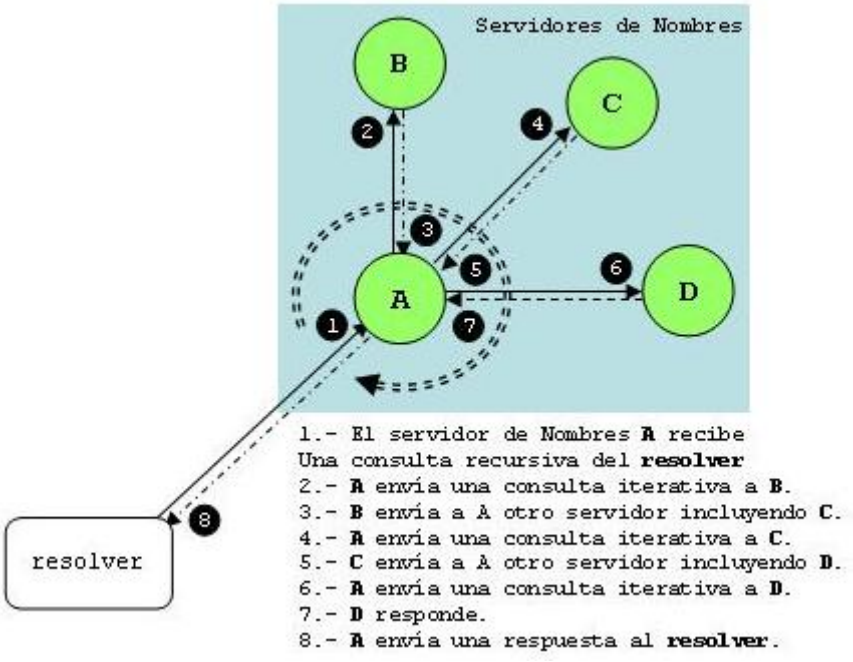


Figura 2-12: Resolución iterativa.

Mapeo de direcciones a nombres (Resolución Inversa)

Una de las mayores funcionalidades carentes en el proceso de resolución explicado es como las direcciones pueden ser mapeadas con nombres. El mapeo dirección-nombre es utilizado para producir información que pueda ser fácil de interpretar por humanos. También se utiliza en algunos procesos de autenticación.

Los datos (incluyendo las direcciones) en el espacio de nombres de dominio están indexados por nombre. Dado un nombre, encontrar una dirección es relativamente fácil.

Debido a que es fácil que dado un dominio encontremos su dirección, ¿porqué no crear parte del espacio de nombres de dominio que utilice direcciones como etiquetas? En el espacio de nombres de dominio de Internet, esta porción del espacio de nombres es el dominio “in-addr.arpa”.

Los nodos en el dominio “in-addr.arpa” tiene etiquetas con los números en la representación de octetos de las direcciones IP.

Los octetos hacen referencia al método común de expresar las direcciones IP como 4 números dentro del rango 0-255, separados por puntos. Ej. 10.10.0.254.

Esto provoca que el dominio “in-addr.arpa” sea gigantesco (como se puede ver en la Figura 3-11) ya que cubre todo el espacio de direcciones IP.

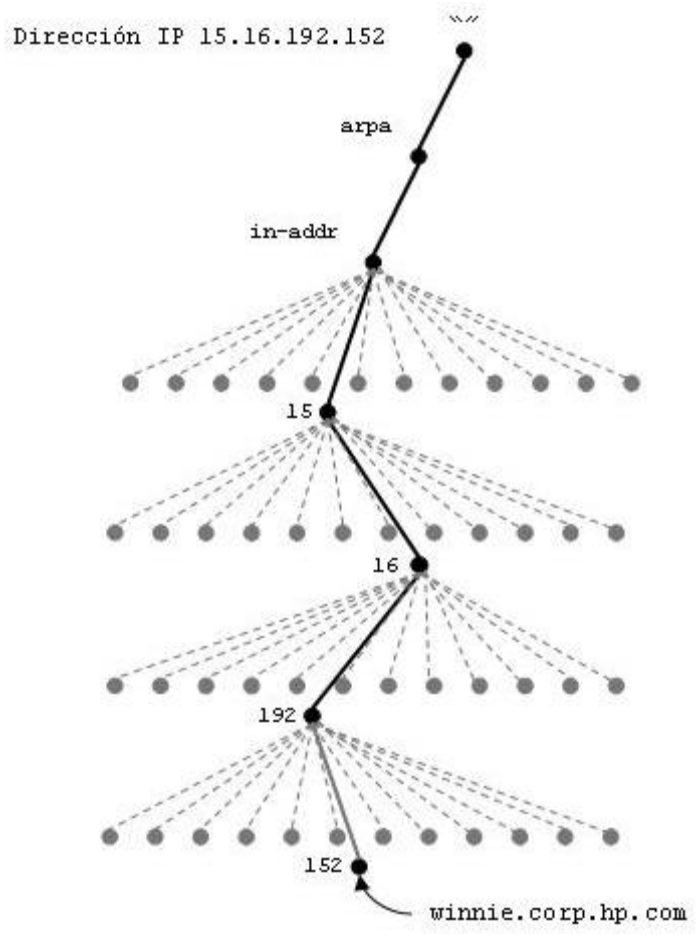


Figura 2-13: dominio in-addr.arpa

Debemos recalcar que cuando se lee un nombre de dominio en “in-addr.arpa”, la dirección IP aparece invertida debido a que el nombre se lee desde las hojas hasta la raíz.

Por ejemplo, “winnie.corp.hp.com” con IP 15.16.192.152, tiene el subdominio “in-addr.arpa”: 152.192.16.15.in-addr.arpa. El cual mapea con el nombre de dominio “winnie.corp.hp.com”.

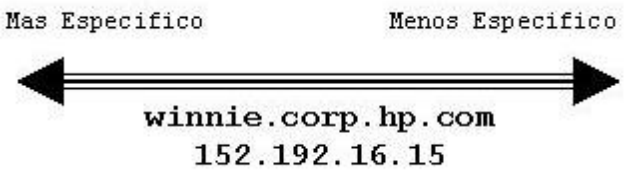


Figura 2-14: Representación jerárquica de nombres y direcciones.

Haciendo que el primer octeto de la dirección IP aparezca más arriba en la estructura de árbol invertido del espacio de nombres de dominio (ver Figura 2-14), proporciona a los administradores la capacidad necesaria para delegar la autoridad para los dominios “in-addr.arpa”.

Caching

Es una característica del DNS que acelera el proceso de resolución de nombres.

Un servidor de nombres que está procesando una petición recursiva podría necesitar enviar unas cuantas peticiones para encontrar la respuesta a su solicitud. No obstante, este servidor descubre mucha información sobre el

espacio de nombres de dominio. Cada vez que hace referencia a una nueva lista de servidores de dominio, el servidor aprende que estos servidores de dominio son autoritativos para ciertas zonas, y aprende las direcciones de estos servidores. Al final de proceso de resolución, cuando finaliza la búsqueda y encuentra los datos que buscaba, este servidor puede almacenar toda esta información aprendida para futuras referencias.

Los servidores de nombres almacenan toda esta información para ayudar a acelerar peticiones sucesivas. La próxima vez que un “resolver” haga una petición al servidor de nombres para datos sobre un dominio sobre el que el servidor de nombres tiene información, el proceso de resolución será más corto.

El servidor de nombres podría almacenar en su caché las respuestas de manera que cuando un resolver haga una petición puede contestar directamente. Incluso si el servidor de nombres no conoce la respuesta pero conoce la identidad de los servidores autoritativos para la zona del nombre de dominio solicitado, podrá hacer la petición directamente a ellos sin necesidad de pasar por servidores raíz.

Ejemplo: Nuestro servidor de nombres tiene almacenado la dirección de “eecs.berkeley.edu”. En el proceso de resolución, el servidor de nombres almacena información (nombres y direcciones) de los servidores de nombres de “eecs.berkeley.edu” y “berkeley.edu”.

Ahora si un “resolver” realiza una petición para la dirección “baobab.cs.berkeley.edu”, el servidor de nombres puede saltar la petición al servidor de nombres raíz. Reconociendo que “berkeley.edu” es el dominio más cercano a “baobab.cs.berkeley.edu”, el servidor de nombres comenzaría la búsqueda a través del servidor “berkeley.edu” (ver Figura 2-16).

Por otro lado, si el servidor de nombres ha descubierto que no hay dirección para “eecs.berkeley.edu”, la próxima vez que reciba una petición para dicha dirección, podrá simplemente responder utilizando la información de su caché.

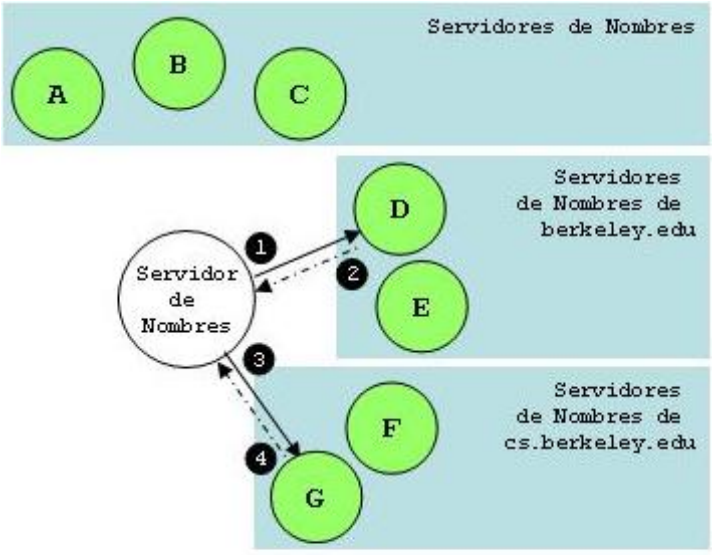


Figura 2-15: Resolviendo con Caching

Entonces el proceso de “caching” permite evitar consultas innecesarias a los servidores de nombres raíz.

Time to Live (TTL)

Los servidores de nombres no pueden guardar los datos en caché indefinidamente (ya que si estos datos cambian no serán correctos y se devolverán datos no válidos).

El administrador de la zona que contiene los datos en caché tendrá que imponer un “Time To Live” o TTL.

El TTL es la cantidad de tiempo que cualquier servidor de nombres mantendrá los datos sobre nuestro dominio en la caché. Después de que este tiempo expira, el servidor de nombres debe descartar los datos en caché y recoger nuevos datos de los servidores de nombre autoritativos.

La decisión de que TTL escoger es una decisión donde hay que balancear los factores de desempeño y consistencia.

Escoger un TTL pequeño ayudará a asegurar que los datos sobre nuestro dominio son consistentes a través de la red, debido que los servidores de nombre remotos tendrán interrupciones más rápidamente y entonces estarán forzados a realizar nuevas peticiones al servidor de nombres autoritativo. Por otro lado, esto incrementa la carga en nuestro servidor de nombres ya que recibiremos más peticiones.

Escoger un TTL grande hará que los servidores que tiene la información de nuestro dominio en caché tengan un tiempo de respuesta a peticiones menor, pero la información inconsistente se mantendrá más tiempo cuando realicemos cambiamos en la zona de nuestro servidor de nombres.

DDNS (Dynamic Domain Name System)

Muchas de las grandes empresas están utilizando la asignación dinámica de direcciones IP, así también como numerosas empresas proveedoras de servicios de Internet utilizan DHCP para asignarles a sus clientes una dirección IP. Por eso se hizo necesario que el DNS soportará adición y eliminación de registros, debido a esto nació el DDNS (DNS Dinámico).

Así pues el DDNS es el encargado de resolver el nombre de una máquina a su respectiva dirección IP, dicha dirección no será la misma, es decir, cambiará de forma dinámica cada vez que el cliente se conecte a Internet o se conecte a una red donde se obtenga la dirección IP a través de DHCP.

Funcionamiento del DDNS.

Primero se necesita tener una máquina corriendo con el Servidor de DDNS, así una serie de clientes autorizados podrán actualizar los registros agregándolos o eliminándolos de la zona donde este ubicado el servido. Un cliente puede encontrar el servidor en una zona recibiendo los registros NS del DNS. Si un servidor recibe un mensaje de actualización y este no es el servidor primario de la zona, entonces se reenvía la actualización “upstream” al servidor principal de la zona, este proceso se conoce como “update forwarding”. Si este servidor es un servidor esclavo de una zona entonces esté reenvía la actualización “upstream” a otro servidor. Solo el servidor principal de la zona recibe los datos de la zona, todos los demás servidores recibirán estos datos provenientes del servidor principal de una manera directa o indirecta (por medio de servidores esclavos). Una vez que el servidor principal ha procesado la actualización dinámica y ha modificado los datos de la zona, los servidores esclavos obtienen una copia por medio de la transferencia de zonas.

Cuando una actualización dinámica hace un cambio permanente en una zona, se necesita llevar un registro en disco de dicha actualización. Pero reescribir los archivos de las zonas cada ves que se agregan, modifican o eliminan registros de dichas zonas puede ser muy pesado para el servidor y consumir muchos recursos. Así pues el escribir estos archivos toma tiempo, y un servidor de nombres puede recibir varias actualizaciones dinámicas cada segundo. Así cada ves que se recibe una actualización dinámica algunos servidores de DDNS lo que hacen es mantener un registro de las actualizaciones en un archivo de sucesos, aunque los cambios sufren efecto de manera inmediata en la copia de la zona que mantiene el servidor en la memoria. Los datos son escritos a disco en un intervalo de tiempo ya definido (usualmente este es de cada hora). Cuando se escriben a disco los datos los servidores de DDNS pueden optar por dos opciones como son: eliminar los archivos de sucesos ya que no se ocupan o dejarlos para las transferencias de zona.

Prerrequisitos para la Instalación de DDNS.

- Instalar el programa BIND.
- Modificar el DHCP de manera que pueda registrar en el servidor de DDNS los cambios hechos o instalar el BIND en el cliente y que este mande las actualizaciones al servidor de DDNS.

Capítulo 3: DHCP.

Introducción.

DHCP es un protocolo que trabaja bajo el modelo Cliente/Servidor y valida las máquinas (clientes DHCP) en una red IP para obtener sus configuraciones desde un servidor (servidor DHCP), esto reduce el trabajo necesario para administrar una red IP. La opción más significativa que recibe el cliente es una dirección IP, aunque se pueden enviar más opciones.

El cliente DHCP puede ser cualquier sistema operativo (Windows 9x/2000/XP, Linux, Unix, MacOS, etc.). La parte cliente solicita los valores de la configuración de Red y esta debe de disponer de un servidor DHCP que administre la asignación de los valores de configuración IP y conteste peticiones de subredes.

Antes de que fuera desarrollado DHCP hubo una serie de protocolos que empezaron con la asignación dinámica como RARP y BOOTP.

RARP.

RARP es un protocolo que trabaja sobre la capa de Enlace de Datos. Provee de un mecanismo que permite a las máquinas determinar su dirección IP conociendo solo su dirección MAC.

RARP es usado cuando se prende una computadora sin disco duro (terminal), como esta no tiene una configuración de una dirección IP almacenada usa RARP para conocer su dirección IP. El servidor RAR contiene una base de datos que simplifica el mapa de direcciones IP correspondientes a cada dirección MAC.

Cuando un cliente RARP quiere encontrar su dirección IP envía una trama de “broadcast“ (con destino a la dirección MAC FF:FF:FF:FF:FF:FF) que contiene su dirección MAC, el servidor RARP recibe el mensaje y este busca en su tabla RARP y si lo encuentra entonces crea un paquete que contiene la dirección IP del Cliente, en caso de que no encuentre la dirección MAC que el cliente envió el paquete es eliminado.

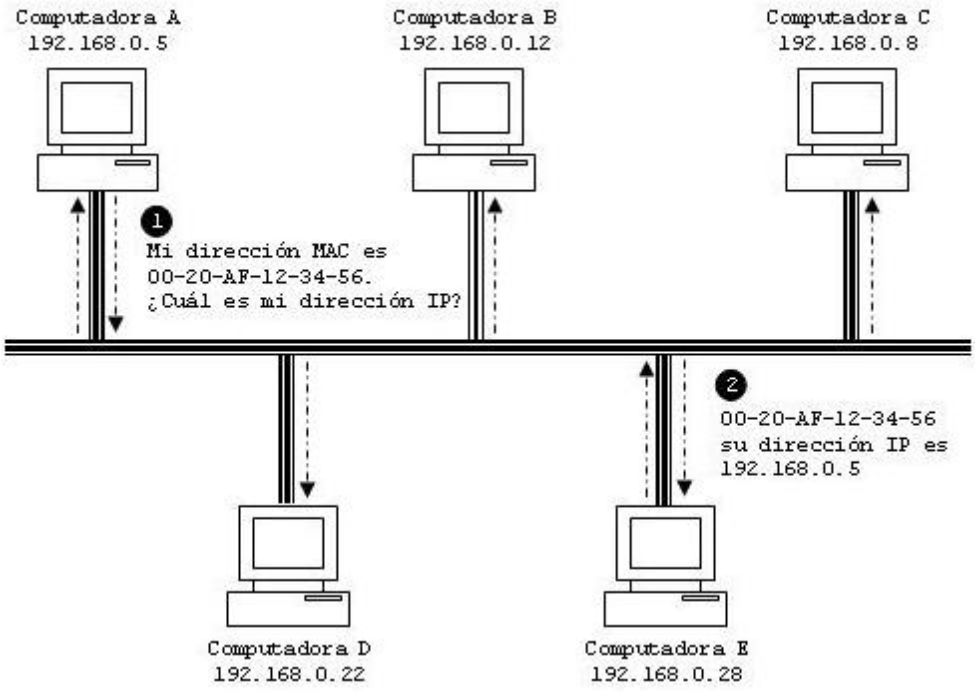


Figura 3-1: Ejemplo de RARP.

RARP puede determinar y configurar la dirección IP de una terminal, pero no puede determinar y entregar otro tipo de información (como mascara de subred, puerta de enlace, etc.). Otra contra de RARP es que el servidor RARP solo puede dar servicio a solo una subred. Para solucionar todo esto se creo el protocolo BOOTP.

BOOTP.

BOOTP, como RARP, es un protocolo utilizado en computadoras sin disco duro para obtener una dirección IP. También puede proveer otros parámetros de configuración que pueden suplir al archivo de inicio. BOOTP es un protocolo basada en IP que usa UDP para permitir la comunicación entre el cliente BOOTP y el servidor BOOTP. Lo mas importante es que BOOTP permite a las máquinas ser configuradas dinámicamente usando el protocolo TCP/IP. En lugar de configurar manualmente cada máquina en una red basada en TCP/IP, BOOTP entrega la información sin necesidad de la intervención del usuario.

BOOTP es un proceso cliente/servidor donde el cliente BOOTP durante la fase de inicio pide la información de la configuración de un servidor BOOTP. El Servidor BOOTP recibe la petición del cliente BOOTP, busca la dirección MAC en su base de datos de configuración de BOOTP y reenvía la información de la configuración IP. El cliente recibe la configuración TCP/IP. El cliente BOOTP también lee el archivo de inicio del servidor BOOTP que provee de una ruta. Su mayor implementación de un mecanismo por el cual el servidor BOOTP provee de las opciones de un Sistema Operativo a los clientes BOOTP. En esas opciones se puede incluir un servidor DNS, WINS o un servidor de nombres de NetBIOS (NBNS), servidores de tiempo, etc.

Alguna de la información más importante suministrada por BOOTP incluye:

- Dirección IP.
- Mascara de Subred.
- Dirección IP de la puerta de enlace predeterminada para la subred del cliente.
- Dirección IP del servidor primario y secundario de DNS.
- Dirección IP del servidor primario y secundario de WINS o NBNS.

Alguna información adicional suministrada por un servidor BOOTP puede incluir:

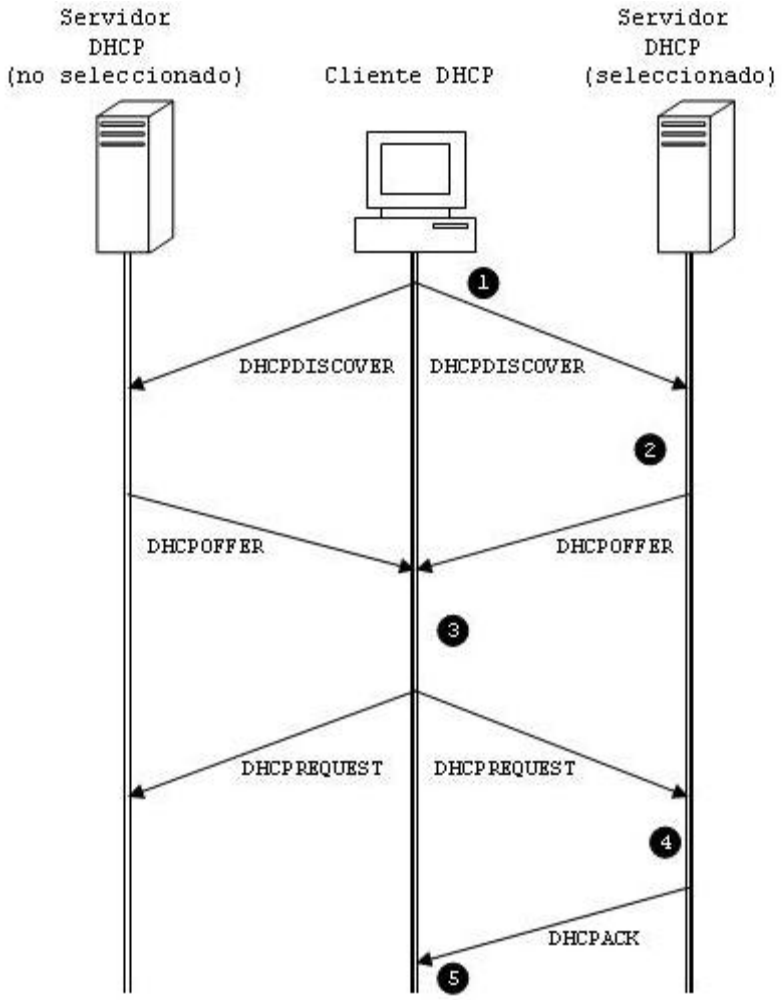
- Dirección IP del Servidor de Inicio.
- El nombre del archivo de inicio que va a ser usado.
- El nombre de Dominio del Cliente.
- Dirección IP del Servidor de Tiempo.
- Time Offset (en segundos) del CMT (Coordinated Universal Time)

Características de DHCP.

DHCP, definido bajo los estándares RFC2131 y RFC2132, fue desarrollado para resolver varias deficiencias de su antecesor BOOTP y agrego varias características como:

- DHCP permite a los Administradores controlar la configuración de los parámetros en la red.
- Los clientes que usan DHCP pueden ser configurados dinámicamente. Esto permite agregar y cambiar redes sin necesidad de cambiar cada máquina o estación de trabajo.
- Para tolerancia a los fallos, múltiples servidores DHCP pueden servir a una o más subredes.
- Servidores DHCP por medio de agentes de relevo de BOOTP pueden servir a una o más subredes.
- DHCP provee de una base de datos dinámica para direcciones IP.
- Los clientes pueden continuar usar direcciones IP asignadas por DHCP hasta que el cliente reinicia. Los clientes BOOTP deben de obtener una dirección IP de un servido BOOTP cuando se esta iniciando.

Funcionamiento de DHCP.



1.- El Cliente envía un DHCPDISCOVER por el puerto 67 (Servidor BOOTP).
2.- Los Servidores escuchan el puerto 67 para determinar la configuración y enviar un DHCPOFFER por el puerto 68 (Cliente BOOTP).
3.- El Cliente recibe el DHCPOFFER seleccionando la configuración y envía un DHCPREQUEST por el puerto 67 (Servidor BOOTP).
4.- El Servidor actualiza la base de datos de DHCP y envía un DHCPACK por el puerto 68 (Cliente BOOTP).
5.- El Cliente configura TCP/IP con la configuración.

Figura 3-2: Conversación DHCP.

El cliente envía una difusión DHCPDISCOVER a todos los nodos. Un cliente está prefijado por DHCP. Dicho cliente envía una petición a un servidor solicitando una configuración IP (habitualmente, durante el proceso de carga). Alternativamente, puede sugerir la dirección IP que desea usar, como cuando solicita una aplicación a un alquiler. El cliente intenta localizar un servidor DHCP enviando una difusión (255.255.255.255) llamada DHCPDISCOVER a su segmento local.

El servidor envía una unidifusión DHCPOFFER al cliente. Cuando el servidor recibe la difusión, determina si puede servir la petición desde si propia base de datos. En caso de no poder hacerlo, intenta reenviar la petición a otro servidor o servidores DHCP, dependiendo de su configuración. Si es capaz de atenderla, el servidor ofrece la información de configuración IP al cliente en forma de una unidifusión DHCPOFFER. La DHCPOFFER es una propuesta de configuración que puede incluir la dirección IP, la dirección del servidor DNS y el tiempo de alquiler.

El cliente envía una difusión DHCPREQUEST a todos los nodos. Si el cliente considera que la oferta es adecuada enviará otra difusión, una DHCPREQUEST, solicitando específicamente esos parámetros IP. ¿Por qué el cliente difunde la petición en lugar de unidifundirla al servidor? Se utiliza una difusión porque el primer mensaje, la señal DHCPDISCOVER, podría llegar a más de un servidor

DHCP. En caso de que varios servidores realicen una oferta, la DHCPREQUEST difundida permite que todo el mundo sepa que dicha oferta fue aceptada, que habitualmente suele ser la primera que se recibió.

El servidor envía una unidifusión DHCPACK al cliente. El servidor que recibe la DHCPREQUEST oficializa la configuración enviando una unidifusión de confirmación, la DHCPACK. Aunque también se puede dar que el servidor no envíe la DHCPACK debido a que haya podido alquilar esa información a otro cliente en el intervalo entre ambas señales. La recepción de la DHCPACK permite al cliente que use inmediatamente la dirección asignada. Si el cliente detecta que la dirección ya está en uso en el segmento local, envía una señal DHCPDECLINE y el proceso se inicia de nuevo. Si el cliente recibe una DHCPACK desde el servidor después de recibir la DHCPREQUEST, reinicia el proceso.

El cliente libera la dirección IP. Si el cliente ya no necesita su dirección IP, envía una DHCPRELEASE al servidor.

Prerrequisitos para la Instalación del DHCP.

- Análisis de la Red para ver las direcciones que se van a asignar dinámicamente.
- Información de la Puerta de Enlace predeterminada e información adicional que se quiera enviar a los clientes DHCP.
- Software de servidor de DHCP.

Capítulo 4: Samba.

Introducción.

Samba es una suite de aplicaciones Unix que habla el protocolo SMB (Server Message Block). Muchos sistemas operativos, incluidos Windows y OS/2, usan SMB para operaciones de red cliente-servidor. Mediante el soporte de este protocolo, Samba permite a los servidores Unix entrar en acción, comunicando con el mismo protocolo de red que los productos de Microsoft Windows. De este modo, una máquina Unix con Samba puede enmascararse como servidor en tu red Microsoft y ofrecer los siguientes servicios:

- Compartir uno o más sistemas de archivos.
- Compartir impresoras, instaladas tanto en el servidor como en los clientes.
- Ayudar a los clientes, con visualizador de Clientes de Red.
- Autentificar clientes contra un dominio Windows.
- Proporcionar o asistir con un servidor de resolución de nombres WINS.

La suite Samba esta compuesta por un de demonios que proporcionan recursos compartidos a clientes SMB sobre la red. Estos demonios son:
smbd. Un demonio que permite compartir archivos e impresoras sobre una red SMB y proporciona autentificación y autorización de acceso para clientes SMB.
nmbd. Un demonio que busca a través del Windows Internet Name Service (WINS), y ayuda mediante un visualizador.

Microsoft también ha contribuido materialmente poniendo a disposición su definición de SMB y del Internet-savvy Common Internet File System (CIFS), como Public Request for Comments (RFC), y otros documentos estándar. El protocolo CIFS es el nuevo nombre de las futuras versiones del protocolo SMB que serán usadas en los productos Windows.

Sin embargo, existen algunas razones específicas por las cuales se podría desear instalar un servidor Samba en una red:

- No se necesita pagar por un servidor Windows NT para obtener las funcionalidades que este proporciona.
- Se puede proporcionar un área común para datos o directorios de usuarios en orden a realizar una transición desde un servidor NT hacia un Unix, o viceversa.
- Se puede compartir impresoras a entre clientes Windows y Unix.
- Se puede acceder a ficheros NT desde un servidor Unix.

Redes SMB/CIFS.

Ahora que ya tienes una breve visión de Samba, tomémonos algún tiempo para familiarizarnos con el entorno que ha adoptado Samba: una red SMB/CIFS. Trabajar con redes SMB es significativamente diferente a trabajar con redes Unix TCP/IP, debido a que hay bastantes conceptos nuevos que aprender y mucha información a cubrir. Primero, discutiremos los conceptos básicos existentes tras una red SMB, seguido de algunas implementaciones de Microsoft a SMB, y finalmente te mostraremos dónde puede encajar un servidor Samba y dónde no.

Introducción a NetBIOS.

En 1984, IBM diseñó un simple "application programming interface" (API) para conectar en red sus computadoras, llamado Network Basic Input/Output System (NetBIOS). El API NetBIOS proporcionaba un diseño rudimentario para que una aplicación se conectara y compartiese datos con otras computadoras.

Es útil pensar en el API NetBIOS como en extensiones de red para llamadas de la API BIOS estándar. Con BIOS, cada llamada de bajo nivel está confinada al hardware de la máquina local y no necesita ayuda para viajar a su destino. NetBIOS, sin embargo, originalmente tenía que intercambiar instrucciones con

computadoras de redes IBM PC o Token Ring. Exigió por consiguiente un protocolo de transporte de bajo nivel para llevar las peticiones de una computadora a la siguiente.

A finales de 1985, IBM lanzó dicho protocolo, el cual unión con el API NetBIOS para convertirse en NetBIOS Extended User Interface (NetBEUI). NetBEUI fue diseñado para redes de área local (LANs), y permitía a cada máquina usar un nombre (de hasta 15 caracteres) que no estuviera siendo usado en la red.

El protocolo NetBEUI se volvió muy popular en las aplicaciones de red, incluyendo a las que corrían bajo Windows para Grupos. Más tarde, emergieron también implementaciones de NetBIOS sobre protocolos IPX de Novell, los cuales competían con NetBEUI. Sin embargo, los protocolos de red escogidos por la comunidad de Internet fueron TCP/IP y UDP/IP, y las implementaciones de las APIs NetBIOS sobre dichos protocolos pronto se convirtieron en una necesidad.

TCP/IP usa números para representar direcciones de computadoras, tales como 192.168.220.100, mientras que NetBIOS usa sólo nombres. Este fue el mayor problema a solucionar a la hora de hacer relacionarse a los dos protocolos. En 1987, El Internet Engineering Task Force (IETF) publicó una serie de documentos de estandarización, titulados RFC 1001 y 1002, que perfilaban

cómo NetBIOS podría trabajar sobre una red TCP/UDP. Este juego de documentos todavía gobiernan a cada una de las implementaciones que existen hoy en día, incluyendo aquellas proporcionadas por Microsoft para sus sistemas operativos, así como a la suite Samba.

Desde entonces, la norma que estos documentos gobiernan se ha conocido como NetBIOS sobre TCP/IP, o NBT para abreviar. El estándar NBT (RFC 1001/1002) actualmente establece un trío de servicios sobre una red:

- Un Servicio de Nombres
- Dos Servicios de Comunicación:
- Datagramas.
- Sesiones.

El servicio de nombres resuelve el problema nombre-a-dirección comentado antes; permite a cada computadora declarar un nombre específico en la red que pueda ser convertido a una dirección IP de máquina, como hacen hoy en día los DNS en Internet. Los servicios de datagramas y sesiones son ambos protocolos secundarios de comunicación, usados para transmitir datos desde y hacia máquinas NetBIOS a través de la red.

Obteniendo un Nombre.

Para un ser humano, tener un nombre es sencillo. Sin embargo, para una máquina sobre una red NetBIOS, esto puede ser algo más complicado.

En el mundo NetBIOS, cuando cada máquina se vuelve activa, quiere reclamar un nombre para sí; esto se denomina registro de nombre. Sin embargo, dos máquinas en el mismo grupo de trabajo podrían solicitar el mismo nombre; esto causaría problemas de confusión para cualquier máquina que quiera comunicar con una de esas dos. Hay dos aproximaciones diferentes para asegurarnos de que esto no ocurra:

- Usar un Servidor de Nombres NetBIOS (NBNS) para controlar el registro de nombres NetBIOS de las máquinas.
- Permitir a cada máquina de la red defender su nombre en el caso de que otra máquina intente usarlo.

La Figura 4-1 ilustra un registro de nombre (negado), con y sin Servidor de Nombres NetBIOS.

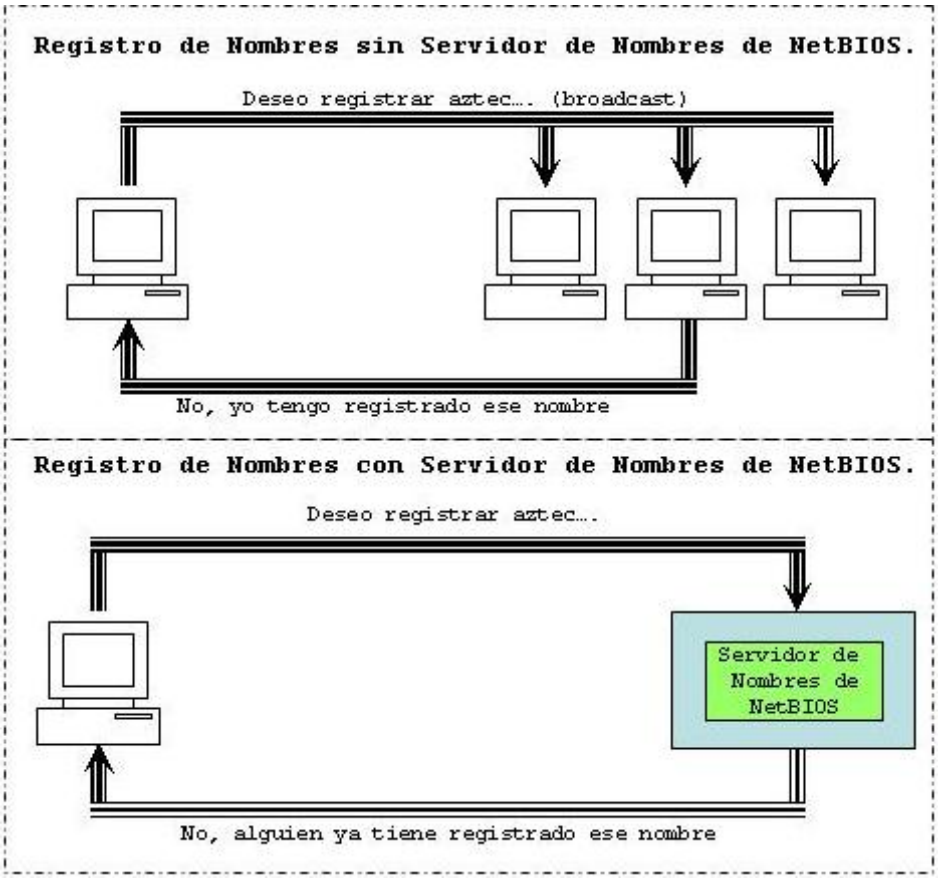


Figura 4-1: Registro de Nombre NBNS contra no-NBNS

Debe haber una forma de resolver un nombre NetBIOS hacia una dirección IP específica como ya mencionamos antes; esto es conocido como resolución de nombre. Hay dos formas diferentes también aquí con NBT:

- Haber reportado cada máquina su dirección IP cuando escucha una petición “broadcast” para su nombre NetBIOS.
- Usar el NBNS para resolver nombres NetBIOS a direcciones IP.

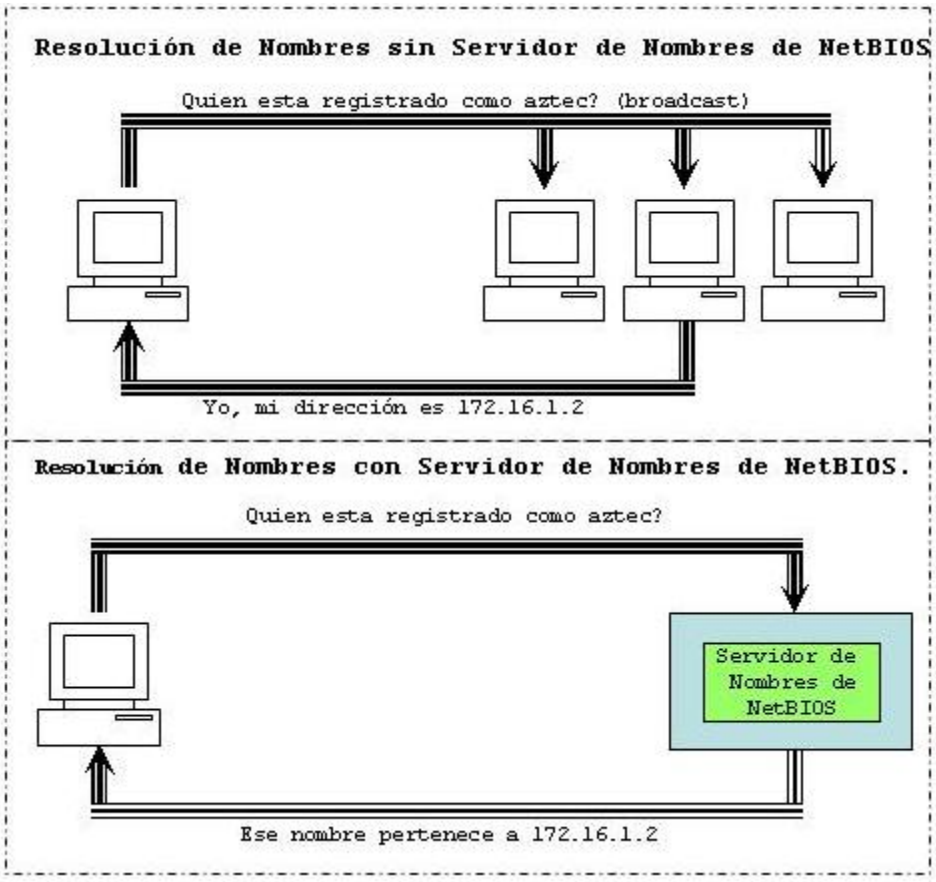


Figura 4-2: Resolución de nombre con-NBNS versus sin-NBNS

Como se puede ver en la Figura 4-2, tener un NBNS en la red puede ayudar enormemente. Para ver exactamente por qué, veamos el método sin-NBNS.

Aquí, cuando una máquina cliente arranca, manda un mensaje “broadcast” declarando que desearía registrar un nombre NetBIOS específico para ella. Si nadie objeta nada ante el uso de ese nombre tras múltiples intentos de registro, obtiene el nombre. En la otra parte, si otra máquina en la red está actualmente usando ese nombre, enviará un mensaje de respuesta al cliente solicitante

indicando que ese nombre ya está siendo usado. Esto es conocido como defender el nombre del equipo. Este tipo de sistema es útil cuando un cliente ha caído inesperadamente de la red -otro puede tomar su nombre-, pero se incurre en un importante aumento del tráfico de la red para algo tan simple como el registro de nombre.

Con un NBNS, ocurre lo mismo, pero con la diferencia de que la comunicación se está confinada a la máquina solicitante y al servidor de nombres NBNS. No ocurre “broadcasting” cuando la máquina desea registrar el nombre; el mensaje de registro es simplemente enviado desde el cliente hacia el servidor NBNS, y este NBNS responde si el nombre está o no libre. Esto es conocido como comunicación punto-a-punto, y es de gran ayuda en redes con más de una subred. Esto se debe a que los routers suelen estar preconfigurados para bloquear paquetes entrantes que son mensajes de difusión (broadcast) para todas las máquinas de la red.

Los mismos principios se aplican a la resolución de nombres. Sin un NBNS, la resolución de nombres NetBIOS podría realizarse mediante un mecanismo broadcast. Todos los paquetes se enviarían a cada una de las computadoras de la red, con la esperanza de que alguna máquina que se vea afectada por la petición responda directamente a la máquina solicitante. En éste punto, queda claro que usar un servidor de nombres NBNS y una comunicación punto-a-punto

para este propósito carga mucho menos la red que usar broadcast para cada una de las peticiones de resolución de nombres que se produzcan.

Tipos de Nodos.

Ahora el problema es como decirles a los clientes qué estrategia deben seguir para realizar el registro de nombre y la resolución. Para esto cada máquina en una red NBT aprende una de las siguientes designaciones, dependiendo de cómo se maneje el registro y la resolución de nombre: b-node, p-node, m-node y h-node. Las conductas de cada tipo de nodo se resumen en la Tabla 1.

Papel	*Valor*
b-node	Usa registro broadcast y sólo resolución.
p-node	Usa registro punto-a-punto y sólo resolución.
m-node	Usa broadcast para registro. Si tiene éxito, notifica al servidor NBNS el resultado. Usa broadcast para resolución; usa servidor NBNS si el broadcast no tiene éxito.
h-node (hybrid)	Usa servidor NBNS para registro y resolución; usa broadcast si el servidor NBNS no responde o no está operativo.

Tabla 4-1: Tipos de Nodos NetBIOS.

En el caso de los clientes Windows, se encontraran listados normalmente como h-nodes o hybrid nodes. Incidentalmente, los h-nodes fueron inventados más tarde por Microsoft, como un tipo de nodo más tolerante a fallos de rutas, y no aparece en el RFC 1001/1002.

Nombres.

Los usos de creación de nombres NetBIOS son diferentes a los de los nombres tipo DNS. Primero, los nombres NetBIOS existen en un espacio único. En otras palabras, no existen calificadores del tipo ora.com o samba.org para definir secciones dentro de los nombres; sólo hay un nombre único para representar a cada computadora. Segundo, los nombres NetBIOS sólo pueden contener hasta 15 caracteres, no pueden comenzar con asterisco (*), y pueden consistir sólo en caracteres alfanuméricos estándar (a-z, A-Z, 0-9) y los unos pocos caracteres especiales (! @ # $ % ^ & ( ) - ' { } . ~).

Aunque puedes usar el punto (.) en un nombre NetBIOS, no es recomendable, debido a que esos nombres puede que no funcionen en las futuras versiones de NetBIOS sobre TCP/IP.

Con NetBIOS, una máquina no sólo advierte de su presencia, sino que también le dice a las otras máquinas qué tipo de servicios ofrece. Por ejemplo, en la figura 4-3 se puede ver una máquina llamada mixtec.

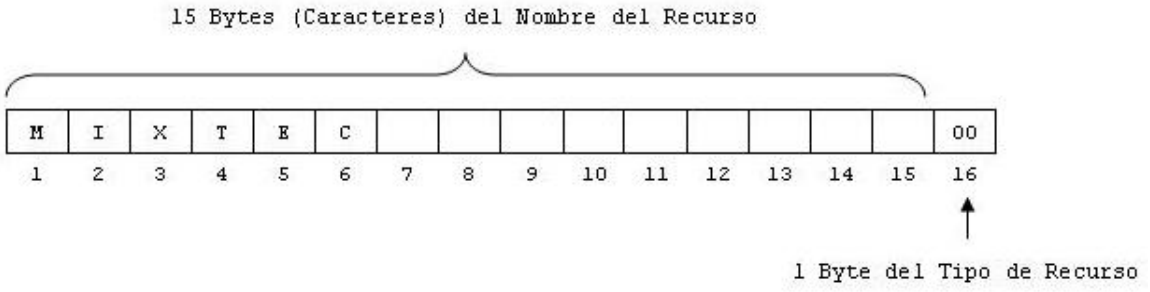


Figura 4-3: Estructura de un Nombre NetBIOS.

Algunos de los posibles atributos que un nombre puede tener se listan en la Tabla 4-2.

Nombre del Recurso	*Valor Hexadecimal*
Standard Workstation Service	00
Messenger Service (WinPopup)	03
RAS Server Service	06
Domain Master Browser Service (associated with primary domain controller)	1B
Master Browser name	1D
NetDDE Service	1F
Fileserver (including printer server)	20
RAS Client Service	21
Network Monitor Agent	BE
Network Monitor Utility	BF

Tabla 4-2: Tipos de Recursos Unicos NetBIOS.

SMB también usa el concepto de grupos. Las máquinas pertenecen a un grupo de trabajo, el cual es una partición de máquinas en la misma red. En el ambiente Windows, un grupo de trabajo y un grupo SMB son la misma cosa. Los posibles atributos de grupo que puede tener una máquina se ilustran en la Tabla 4-3.

Nombre del Recurso	*Valor Hexadecimal*
Standard Workstation Group	00
Logon Server	1C
Master Browser name	1D
Normal Group name (used in browser elections)	1E
Internet Group name (administrative)	20
<01><02>_ _MSBROWSE_ _<02>	01

Tabla 4-3: Tipos de Recursos de Grupo NetBIOS.

La entrada final, _ _ MSBROWSE _ _ , se usa para anunciar un grupo a otros visualizadores maestros. Los caracteres no impresos en el nombre se muestran como guiones bajos en una salida de NBTSTAT.

Datagramas y Sesiones.

Una responsabilidad de NBT es proporcionar servicios de conexión entre dos máquinas NetBIOS. Existen actualmente dos servicios ofrecidos por NetBIOS sobre TCP/IP: el servicio de sesiones y el servicio de datagramas.

El servicio de datagramas ofrece una conexión no estable entre una máquina y otra. Los paquetes de datos son simplemente enviados o difundidos (broadcasting) de una máquina a otra, sin considerar el orden en que estos llegan al destino, o si han llegado todos. El uso de datagramas no incrementa tanto el tráfico de la red como el uso de sesiones, aunque pueden echar abajo una red si se usan indebidamente. Los datagramas, por tanto, son empleados para enviar rápidamente sencillos bloques de datos a una o más máquinas. El servicio de comunicación de datagramas usando las primitivas simples se muestra en la Tabla 4-4.

Primitiva	Descripción
Send Datagram	Envía paquete datagrama a máquina o grupos de máquinas.
Send Broadcast Datagram	Difunde (broadcast) datagrama a cualquier máquina, esperando un datagrama de acuse de recibo.
Receive Datagram	Recibe un datagrama de una máquina.
Receive Broadcast Datagram	Espera por un datagrama de difusión.

Tabla 4-4: Primitivas de Datagramas.

El servicio de sesiones es más complejo. Las sesiones son un método de comunicación que, en teoría, ofrece la capacidad de detectar conexiones problemáticas o inoperativas entre dos aplicaciones NetBIOS. Esto lleva a pensar en una sesión NBT en términos de una llamada telefónica. Una conexión full-duplex es abierta entre una máquina que llama y una máquina que es llamada, y la conexión debe permanecer abierta durante la duración de la conversación. Cada parte implicada conoce a la otra máquina, y pueden comunicar con las primitivas que se muestran en la Tabla 4-5.

Primitiva	*Descripción.*
Call	Inicia una sesión con una máquina que está a la escucha bajo un nombre específico.
Listen	Espera una llamada de un llamante conocido o cualquier otro
Hang-up	Termina una llamada.
Send	Envía datos a la otra máquina.
Receive	Recibe datos de la otra máquina.
Session Status	Obtiene información sobre sesiones pedidas.

Tabla 4-5: Primitivas de Sesiones

Las sesiones son el troncal de la compartición de recursos en una red NBT. Son normalmente usadas para establecer conexiones estables desde máquinas clientes a unidades de disco o impresoras compartidas en un servidor. El cliente

"llama" e inicia la conversación, enviando información del tipo qué ficheros desea abrir, qué datos quiere intercambiar, etc. Estas llamadas pueden durar mucho tiempo -horas, incluso días- y todo esto ocurre dentro del contexto de una única conexión. Si se produce un error, el software de sesión (TCP) retransmitirá hasta que los datos sean recibidos correctamente, a diferencia del "envía-y-reza" del servicio de datagramas (UDP).

En realidad, mientras que las sesiones se supone están para manejar comunicaciones problemáticas, normalmente no lo hacen. Como probablemente habrás descubierto al usar redes Windows, es un serio problema el usar sesiones NBT. Si la conexión es interrumpida por la razón que sea, la información de sesión que está abierta entre dos computadoras puede fácilmente volverse inválida. Si esto ocurre, la única forma de restablecer la sesión para las dos mismas máquinas es llamar de nuevo y comenzar desde cero. Sin embargo, hay dos cosas importantes a recordar aquí:

- Las sesiones siempre ocurren entre dos máquinas NetBIOS -ni más ni menos-. Si un servicio de sesión es interrumpido, se supone que el cliente ha almacenado la suficiente información de estado como para restablecer la comunicación. Sin embargo, en la práctica, es raro el caso.
- Los datagramas pueden ser difundidos a múltiples máquinas, pero son inestables. Dicho de otro modo, no hay forma para el emisor de saber si

los datagramas que ha enviado han llegado correctamente a los destinatarios.

Prerrequisitos para la Instalación de Samba.

- Software para el servidor Samba.
- Recursos a compartir (Impresoras, Directorios, etc.).
- Nombre del grupo de trabajo.
- Nombre NetBIOS de la máquina que funcionara como servidor Samba.
- Usuarios que tendrán acceso a los recursos.

Capítulo 5: Integrando DDNS, DHCP y Samba en una Red.

Antecedentes.

Una empresa en sus oficinas centrales tiene una red con más de 20 máquinas que se encuentran configuradas de manera estática y una serie de 10 computadoras portátiles que necesitan acceso a la red y se configuran de manera dinámica. Las computadoras portátiles llegan de manera esporádica a la oficinas central ya que pertenecen a encargados de obras que se encuentran fuera de la ciudad y que solo van a la oficina a rendir cuentas, pero necesitan tener acceso a la red local para bajar la información al servidor.

Así pues es necesario que cada departamento (compras, ventas, obras, contabilidad y recursos humanos) tengan un espacio para poder guardar los documentos comunes o que deben de estar disponibles para todos los usuarios, ya que es muy molesto tener que estarse pasando los documentos a cada usuario del departamento que los necesite. Al igual que compartir los recursos como impresoras y unidades de disco.

Además de todo esto se requiere que se pueda acceder a las máquinas por su nombre canónico y no solo por su dirección IP, esto también incluye a las computadoras portátiles que pueden llegar a la oficina.

DHCP

Instalación.

Para la instalación de DHCP podemos instalarlo desde las fuentes que están ubicadas en la página del Consorcio de Software de Internet (Internet Software Consortium ISC), o podemos utilizar el repositorio de Ubuntu Dapper. La ventaja que tenemos al realizar la instalación desde el repositorio de Ubuntu Dapper es que si el programa necesita de otras librerías o programas que no estén instalados en la máquina, entonces estos serán descargados e instalados de forma automática y la instalación se podrá realizar de forma adecuada. Por otra parte si lo hacemos desde las fuentes necesitamos tener instaladas todas las librerías necesarias para poder compilar el programa y además de eso tener el compilador necesario.

Debido a las razones antes mencionadas utilizaremos el repositorio de Ubuntu Dapper. Lo primero que necesitamos es actualizar la base de datos del repositorio, ubicada en *"/etc/apt/sources.list"* y verificar que estén utilizando los repositorios mas actualizados.

Ya que hemos modificado el archivo del repositorio necesitamos actualizar la base de datos de la máquina por medio de los siguientes comandos:

```
$ sudo aptitude update
$ sudo aptitude upgrade
```

Una vez actualizado la base de datos del repositorio y actualizado el sistema procedemos a instalar el DHCP.

```
sudo apt-get install dhcp3-server
```

Ya con esto tenemos instalado el servidor de DHCP en nuestro servidor.

Configuración.

Se tiene configurada una 192.168.1.0/26, y se esta pensando asignar el primer host al router, la segunda al servidor de DHCP / DNS y Samba, las 20 subsecuentes a los equipos configurados estáticamente (192.168.1.3 a la 192.168.1.23) y el resto de la direcciones asignarlas dinámicamente (192.168.1.24 a la 192.168.1.63).

Para esto necesitamos editar el archivo *"/etc/dhcp3/dhcpd.conf"* y ahí configurar nuestra red.

```
$ sudo gedit /etc/dhcp3/dhcpd.conf
```

La configuración que contendrá este archivo es la siguiente:

```
default-lease-time 64800;
max-lease-time 64800;
option domain-name-servers 192.168.1.1;
subnet 192.168.1.0 netmask 255.255.255.192 {
    option subnet-mask 255.255.255.192;
    option broadcast-address 192.168.1.64;
    range 192.168.1.24 192.168.1.63;
    option routers 192.168.1.1;
}
```

Con esta configuración le estaremos pasando a cada equipo que se conecte de forma dinámica la siguiente configuración:

- Servidor de DNS = 192.168.1.1
- Mascara de Subred = 255.255.255.192
- Puerta de Enlace Predeterminada = 192.168.1.1

Samba.

Instalación.

La instalación de Samba se realizara también desde el repositorio de Ubuntu Dapper.

```
$ sudo aptitude install samba
$ sudo aptitude install smbfs
```

Configuración.

Lo primero que configuraremos es el archivo *"/etc/samba/smb.conf"* el cual contiene toda la información.

```
$ sudo gedit /etc/samba/smb.conf
```

El directorio que se va a compartir es el *"/home/public"* y no es necesario tener autentificación, el archivo *"/etc/samba/smb.conf"* quedara de la siguiente manera:

```
[global]
netbios name = Servidor
workgroup = EMPRESA
server string = Servidor de Archivos
security = share

[Compras]
comment = Carpeta Compartida de Compras
path = /home/compras
public = yes
writable = yes
create mask = 0777
directory mask = 0777
force user = nobody
force group = nogroup

[Ventas]
comment = Carpeta Compartida de Ventas
path = /home/ventas
public = yes
writable = yes
create mask = 0777
directory mask = 0777
force user = nobody
```

```
force group = nogroup


[Obras]
comment = Carpeta Compartida de Obras
path = /home/obras
public = yes
writable = yes
create mask = 0777
directory mask = 0777
force user = nobody
force group = nogroup


[Contabilidad]
comment = Carpeta Compartida de Contabilidad
path = /home/contabilidad
public = yes
writable = yes
create mask = 0777
directory mask = 0777
force user = nobody
force group = nogroup


[RH]
comment = Carpeta Compartida de Recursos Humanos
```

```
path = /home/rh

public = yes

writable = yes

create mask = 0777

directory mask = 0777

force user = nobody

force group = nogroup
```

DDNS con Cliente.

Servidor.

Instalación.

Lo primero que necesitamos es instalar Bind 9, esto lo haremos utilizando los repositorios de Ubuntu Dapper.

```
$ sudo aptitude install bind9
```

Esto hará que se instale la versión 9 del programa Bind.

Configuración.

Primero necesitamos establecer la zona que vamos a utilizar, esto lo hacemos editando el archivo *"/etc/bind/named.conf.local"* y agregamos la siguiente información:

```
zone "empresa.net" {
     notify no;
     type master;
     allow-update{
          127.0.0.1;
          192.168.1.0/26;
     };
     file "/etc/bind/db.empresa.net";
};
```

Con lo cual le estamos diciendo al servidor que va tener una zona llamada *"empresa.net"* le dice que es de tipo "maestro" y que la información para dicha zona se encuentra en el archivo *"/etc/bind/db.empresa.net"* ahora necesitamos crear el archivo y agregarle los datos de la zona que vamos a utilizar.

```
$TTL 3D
@ IN SOA servidor.empresa.net. (
     2006090901 ; serie, fecha de hoy + serie de hoy #
     8H ; refresco, segundos
     2H ; reintento, segundos
     4W ; expira, segundos
     1D ) ; mínimo, segundos
;
localhost A 127.0.0.1
```

```
server A 192.168.1.2
```

Donde estamos dando de alta el servidor y estamos agregando las opciones para que el servidor de DDNS pueda funcionar. Ahora nuestro servidor hace el cambio de nombres canónicos a direcciones IP, solo necesitamos reiniciar el Bind.

```
$ sudo /etc/init.d/bind9 restart
```

Ahora vamos a hacer que haga el proceso inverso de convertir direcciones IP a nombres canónicos, para esto volvemos a editar el archivo *"/etc/bind/named.conf.local"* y agregamos la siguiente información:

```
zone "1.168.192.in-addr.arpa" {
     notify no;
     type master;
     allow-update{
          127.0.0.1;
          192.168.1.0/26;
     };
     file "/etc/bind/db.192.168.1";
};
```

Y necesitamos crear el archivo "/etc/bind/db.192.168.1" para decirle las configuraciones que debe de tener el servidor.

```
$TTL 3D
@ IN SOA servidor.empresa.net. (
     2006090901; Serial, todays date + todays serial
     8H ; Refresco
     2H ; Reintento
     4W ; Expira
    1D) ; Minimo TTL
NS servidor.empresa.net.
2 PTR servidor.empresa.net.
```

Ahora necesitamos introducir al archivo *"/etc/named.conf"* el usuario al que se le va a permitir el acceso y la clave que este debe de tener.

```
key keyname {
    algorithm HMAC-MD5.SIG-ALG.REG.INT;
    secret password;
};
```

En la parte de keyname se debe de sustituir por el nombre del usuario y se debe de remplazar password por la clave TSIG generada en el cliente. Solo resta

decirle al servidor a quien le vamos a dar permiso de actualizar, esto lo hacemos editando el archivo *"/etc/bind/named.conf.local"*, agregando la opción "allow-update" y quedando como sigue:

```
zone "empresa.net" {
     notify no;
     type master;
     allow-update{
          127.0.0.1;
          192.168.1.0/26;
           key keyname;
     };
     file "/etc/bind/db.empresa.net";
};
zone "1.168.192.in-addr.arpa" {
     notify no;
     type master;
     allow-update{
          127.0.0.1;
          192.168.1.0/26;
           key keyname;
     };
     file "/etc/bind/db.192.168.1";
};
```

Cliente.

Instalación.

Ahora necesitamos instalar Bind 9 en el cliente del DDNS, esto lo haremos de la misma manera que en el servidor, utilizando los repositorios de Ubuntu Dapper.

```
$ sudo aptitude install bind9
```

Con esto instaremos la versión 9 del programa Bind.

Configuración.

Lo primero que necesitamos es crear una clave TSIG (Secret Key Transaction Signatures for DNS), esto lo haremos con el siguiente comando:

```
$sudo dnskeygen -H 128 -h -n keyname
```

Aquí *"keyname"* es el nombre del usuario que usaremos. Esto nos creara dos archivos "*Kkeyname.+157+00000.key*" y "*Kkeyname.+157+00000.private*" estos dos archivos se deben de poner en el directorio de zonas *"/etc/bin/".*

Una ves generados estos archivos encontraremos en ellos la clave que debe de llevar el servidor para poder autentificar al usuario, por ejemplo si ejecutamos el comando *"dnskeygen"* con el nombre *"usuario"* obtendremos los archivos *"Kusuario.+157+00000.key"* y *"Kusuario.+157+00000.private"*. En el archivo *"Kusuario.+157+00000.key"* se generará lo siguiente:

```
bob. IN KEY 513 3 157 hrCDCUNBtlY3sgF8NPnJrg==
```

Siendo "*hrCDCUNBtlY3sgF8NPnJrg==*" la clave que introduciremos en el servidor de DDNS para autentificarlo.

Ahora lo que resta es hacer que el cliente de DDNS pase la actualización al servidor, esto lo haremos con el siguiente comando:

```
nsupdate -k /etc/namedb/tsig:keyname.
```

Donde *"keyname"* es el usuario con el cual creamos los archivos. Ahora solo resta decir que es lo que haremos y en que dirección. Esto lo hacemos con el comando:

```
update usuario.empresa.net. 600 IN A 192.168.0.1
```

La primera parte *“update”* indica que se va a actualizar también se puede poner *“update add”* para agregar el registro al servidor, la segunda *“usuario.empresa.net.”* es el dirección sobre la cual se va a trabajar, lo siguiente *“600”* es el TTL, la siguiente *“IN A”* indica el tipo de registro sobre el que se va a trabajar en este caso el registro A y por ultimo la dirección *“192.168.0.1”* que indica el servidor de DDNS al cual se le van a mandar las actualizaciones.

DDNS sin Cliente.

Instalación.

Lo primero que necesitamos es instalar Bind 9 en el servidor, esto lo haremos utilizando los repositorios de Ubuntu Dapper.

```
$ sudo aptitude install bind9
```

Esto hará que se instale la versión 9 del programa Bind.

Configuración.

Lo primero que necesitamos definir es la zona sobre la cual estaremos trabajando. Para esto la editamos el archivo ”/etc/bind/named.conf.local” y agregamos la siguiente información:

```
zone "empresa.net" {
     notify no;
     type master;
     allow-update{
```

```
          127.0.0.1;
          192.168.1.0/26;
     };
     file "/etc/bind/db.empresa.net";
};
```

Con lo cual le estamos diciendo al servidor que va tener una zona llamada “empresa.net” le dice que es de tipo “maestro” y que la información para dicha zona se encuentra en el archivo “/etc/bind/db.empresa.net” ahora necesitamos crear el archivo y agregarle los datos de la zona que vamos a utilizar.

```
$TTL 3D
@ IN SOA servidor.empresa.net. (
     2006090901 ; serie, fecha de hoy + serie de hoy #
     8H ; refresco, segundos
     2H ; reintento, segundos
     4W ; expira, segundos
     1D ) ; mínimo, segundos
;
localhost A 127.0.0.1
server A 192.168.1.2
```

Donde estamos dando de alta el servidor y estamos agregando las opciones para que el servidor de DDNS pueda funcionar. Ahora nuestro servidor hace el cambio de nombres canónicos a direcciones IP, solo necesitamos reiniciar el Bind.

```
$ sudo /etc/init.d/bind9 restart
```

Ahora vamos a hacer que haga el proceso inverso de convertir direcciones IP a nombres canónicos, para esto volvemos a editar el archivo ”/etc/bind/named.conf.local” y agregamos la siguiente información:

```
zone "1.168.192.in-addr.arpa" {
     notify no;
     type master;
     allow-update{
          127.0.0.1;
          192.168.1.0/26;
     };
     file "/etc/bind/db.192.168.1";
};
```

Y necesitamos crear el archivo "/etc/bind/db.192.168.1" para decirle las configuraciones que debe de tener el servidor.

```
$TTL 3D
@ IN SOA servidor.empresa.net. (
     2006090901; Serial, todays date + todays serial
     8H ; Refresco
     2H ; Reintento
     4W ; Expira
    1D) ; Minimo TTL
NS servidor.empresa.net.
2 PTR servidor.empresa.net.
```

Ahora necesitamos que los clientes de DHCP puedan registrar su nombre eso lo haremos editando el archivo “/etc/dhcp3/dhcpd.conf” de DHCP que anteriormente habíamos creado, agregando información del DDNS, la información que le agregaremos será la siguiente:

```
ddns-domainname "empresa.net";
ddns-update-style interim;
ddns-updates on;
zone empresa.net. {
     primary 127.0.0.1;
}
zone 1.168.192.in-addr.arpa. {
     primary 127.0.0.1;
}
```

Conclusiones.

Como se puede ver, Linux ha crecido de una manera tal que hoy en día podemos ver a Linux como una alternativa viable en Servidores de gran importancia dentro de las empresas ya que nos ofrece un gran desempeño y bajo costo.

Un servidor Linux puede contener todas las herramientas necesarias a un bajo costo, ya que no se tienen que pagar las licencias para hacer uso de esté. También podemos ver que el Software Libre no es tan difícil de configurar como anteriormente se creía o pensaba.

Se mostró también que un servidor DHCP bajo Linux es fácil de instalar así como de configurar y este puede estar a la altura de cualquier servidor DHCP que exista en el mercado (tanto de Microsoft como de otras grandes compañías).

También se mostró como configurar Linux de manera que los clientes que utilicen Windows puedan acceder a los recursos que el servidor Linux proporciona por medio del programa Samba y se mostró la manera de configurar un cliente DDNS bajo Linux para así poder salir a Internet con un nombre genérico de Internet (nombre de domino).

Así podemos concluir que hoy en día el Software Libre es una alternativa viable en cuanto a costos, rendimiento y funcionalidad, ya que hoy en día se dispone de las herramientas necesarias para poder trabajar con varios sistemas operativos a distintos niveles de usuarios de una manera rápida, segura y de forma totalmente transparente para el usuario final.

Bibliografía.

Beyond DHCP - Work Your TCP/IP Internetwork With Dynamic IP.

Oscar Cepeda, Bob Chambers, Julian Mosca, Matt Robbins

International Technical Support Organization, 2000.

BIND 9 Administrator Reference Manual.

Nominum, Inc., 2001.

Linux+ Certification Bible

Trevor Kay

HungryMinds, 2002.

Linux & Windows Integration with Samba

International Technology Solutions Inc., 2006.

Integrating Linux and Windows

Mike McCune

Prentice Hall, 2000.

DHCP for Windows 2000.

Neall Alcott

O'Reilly, 2001.

Learning Debian GNU Linux

Bill McCarty

O'Reilly, 1999.

Linux Network Administrator Guide, 2nd Ed

Olaf Kirch & Terry Dawson

O'Reilly, 2000

Red Hat Linux Networking and System Administration

Terry Collings & Kurt Wal

Red Hat Press, 2002

The Official Samba-3 HOWTO and Reference Guide

Jelmer R. Vernooij, John H. Terpstra, and Gerald (Jerry) Carter

Prentice Hall, 2005.

Linux Network Servers

Craig Hunt

SYBEX, 2002.

Linux Bible – Debian

Steve Hunger

Wiley, 2001.

DNS and BIND, 4th Edition

Paul Albitz, Cricket Liu

O'Reilly, 2001.

Learning Red Hat Linux, 3rd Edition

Bill McCarty

O'Reilly, 2003.

The Network Administrators' Guide

Olaf Kirch.

O'Reilly, 1996.

Ubuntu Hacks

Bill Childers, Jonathan Oxer, Kyle Rankin

OReilly, 2006.

Using Samba, 2nd Edition

David Collier-Brown, Robert Eckstein, Jay Ts

O'Reilly, 2003.

Apéndice A. Glosario de Términos Clave.

API (Application Programming Interface): es un conjunto de especificaciones de comunicación entre componentes software. Representa un método para conseguir abstracción en la programación, generalmente (aunque no necesariamente) entre los niveles o capas inferiores y los superiores del software. Uno de los principales propósitos de una API consiste en proporcionar un conjunto de funciones de uso general, de esta forma, los programadores se benefician haciendo uso de su funcionalidad, evitándose el trabajo de programar todo desde el principio.

AppleTalk: es un protocolo propietario de Apple que se utiliza para conectar ordenadores Macintosh en redes locales. Admite las tecnologías Ethernet y Token Ring.

ARPANET (Advanced Research Projects Agency Network): fue creada por encargo del Departamento de Defensa de los Estados Unidos como medio de comunicación para los diferentes organismos del país. El primer nodo se creó en la Universidad de California y fue la espina dorsal de Internet hasta 1990, tras finalizar la transición al protocolo TCP/IP en 1983.

BOOTP (Bootstrap Protocol): es un protocolo de red UDP utilizado por los clientes de red para obtener su dirección IP automáticamente. Normalmente se realiza en el proceso de arranque de los ordenadores o del sistema operativo.

BSD (Berkeley Software Distribution): es un sistema operativo derivado del sistema Unix nacido a partir de las aportaciones realizadas a ese sistema por la Universidad de California en Berkeley.

Caching: es una característica del DNS que acelera el proceso de resolución de nombres.

CIFS (Common Internet File System): es un protocolo sucesor de SMB el cual añade más características, que incluyen soporte para enlaces simbólicos, enlaces duros (hard links), y mayores tamaños de archivo.

Datagrama: es un fragmento de paquete que es enviado con la suficiente información como para que la red pueda simplemente encaminar el fragmento hacia el ordenador receptor, de manera independiente a los fragmentos restantes. Esto puede provocar una recomposición desordenada o incompleta del paquete en el ordenador destino.

DDNS (Dynamic Domain Name System): es el encargado de resolver el nombre de una máquina a su respectiva dirección IP y esta cambiará de forma dinámica cada vez que el cliente se conecte a la red.

DHCP (Dynamic Host Configuration Protocol): es un protocolo de red en el que un servidor posee una lista de direcciones IP dinámicas y las va asignando a usuarios conforme estas van estando libres, sabiendo en todo momento quien ha estado en posesión de esa IP, cuanto tiempo la ha tenido y a quien se la ha asignado después.

DNS (Domain Name System): es una base de datos distribuida y jerárquica que almacena información asociada a nombres de dominio en redes como Internet.

Ext3 (third extended filesystem): es un sistema de archivos con registro por diario por tener un "espacio apartado para el buffer de journaling", es usado principalmente en sistemas Linux.

Fat (File Allocation Table): es un sistema de ficheros desarrollado para MS-DOS, así como el sistema de archivos principal de las ediciones no empresariales de Microsoft Windows hasta Windows Me.

Firewall: es un elemento de hardware o software utilizado en una red de computadoras para controlar las comunicaciones, permitiéndolas o prohibiéndolas según las políticas de red que haya definido la organización responsable de la red.

Gnome: es un entorno de escritorio para sistemas operativos de tipo Unix bajo tecnología X Window.

GNU (GNU's Not Unix): fue lanzado en 1984 para desarrollar un completo sistema operativo tipo UNIX, bajo la filosofía del software libre: el sistema GNU. Las variantes del sistema operativo GNU que utilizan el núcleo llamado Linux se refieren como Linux aunque deberían de ser llamados sistemas GNU/Linux.

IP (Internet Protocol): es un protocolo no orientado a conexión usado tanto por el origen como por el destino para la comunicación de datos a través de una red de paquetes conmutados.

IPX (Internetwork Packet Exchange): es un protocolo de nivel de red de Netware. Se utiliza para transferir datos entre el servidor y los programas de las estaciones de trabajo. Los datos se transmiten en datagramas.

Kerberos: es un protocolo de autenticación de redes de computadoras que permite a dos computadores en una red insegura demostrar su identidad mutuamente de manera segura.

Kernel: es el software responsable de facilitar a los distintos programas acceso seguro al hardware de la computadora o en forma más básica, es el encargado de gestionar recursos, a través de servicios de llamada al sistema.También se encarga de decidir qué programa podrá hacer uso de un dispositivo de hardware y durante cuanto tiempo.

Lenguaje C: es un lenguaje de programación creado en 1969 por Ken Thompson y Dennis M. Ritchie en los Laboratorios Bell. Se trata de un lenguaje débilmente tipado de medio nivel pero con muchas características de bajo nivel. Dispone de las estructuras típicas de los lenguajes de alto nivel pero, a su vez, dispone de construcciones del lenguaje que permiten un control a muy bajo nivel.

Lenguaje Ensamblador: es un tipo de lenguaje de bajo nivel utilizado para escribir programas de computadora, y constituye la representación más directa del código máquina específico para cada arquitectura de computadoras legible por un programador.

Linux: sistema operativo tipo Unix (que implementa el estándar POSIX), que utiliza primordialmente filosofía y metodologías libres (también conocido como GNU/Linux) y que está formado mediante la combinación del núcleo Linux con las bibliotecas y herramientas del proyecto GNU y de muchos otros proyectos/grupos de software (libre o no libre).

Mac OS (Macintosh Operating System): es el nombre del primer sistema operativo de Apple para los ordenadores Macintosh. El Mac OS original fue el primer sistema operativo con una interfaz gráfica de usuario en tener éxito.

Minix: es un clon del sistema operativo Unix distribuido junto con su código fuente y desarrollado por el profesor Andrew S. Tanenbaum en 1987. Fue creado para enseñar a sus alumnos el diseño de sistemas operativos en la Vrije Universiteit de Ámsterdam ya que Unix estaba bajo restricciones de licencia de AT&T, era demasiado complicado y corría sobre máquinas complejas.

MS-DOS (MicroSoft Disk Operating System): es un sistema operativo monousuario y monotarea

NetBEUI (NetBIOS Extended User Interface): es un protocolo de nivel de red sencillo utilizado en las primeras redes de Microsoft como Lan Manager o en Windows 95. La comunicación entre equipos se consigue gracias al intercambio de sus nombres en una red de área local, pero no dispone de mecanismos para conectar equipos que estén en redes separadas.

NetBIOS: es un protocolo de red originalmente creado para redes locales de computadoras IBM PC. NetBIOS fue la API del producto llamado "PC Network", desarrollado por Sytec, empresa contratada por IBM. "PC Network" soportaba menos de 80 nodos y era bastante simple.

NFS (Network File System): es un sistema de archivos distribuido para un entorno de red de área local. Posibilita que distintos sistemas conectados a una misma red accedan a ficheros remotos como si se tratara de locales. Este protocolo está diseñado para ser independiente de la máquina, el sistema operativo y el protocolo de transporte.

NTFS (New Technology File System): es un sistema de archivos diseñado específicamente para Windows NT, con el objetivo de crear un sistema de archivos eficiente, robusto y con seguridad incorporada desde su base, está basado en el sistema de archivos HPFS de IBM/Microsoft usado en el sistema operativo OS/2, y también tiene ciertas influencias del formato de archivos HFS diseñado por Apple.

OS/2: es un sistema operativo de IBM que intentó suceder a MS-DOS como sistema operativo de las computadoras. Se desarrolló inicialmente de manera conjunta entre Microsoft e IBM, hasta que la primera decidió seguir su camino con su Windows 3.0 e IBM se ocupó de OS/2.

POSIX (Portable Operating System Interface base on Unix): son una familia de estándares de llamadas al sistema operativo definido por el IEEE y especificado formalmente en el IEEE 1003. Persiguen generalizar las interfaces de los sistemas operativos para que una misma aplicación pueda ejecutarse en distintas plataformas.

RARP (Reverse Address Resolution Protocol): es un protocolo utilizado para resolver la dirección IP de una dirección hardware dada (como una dirección Ethernet).

Resolver: es el cliente que se conecta a un servidor de nombres de dominio,

RFC (Request For Comments): son un conjunto de notas técnicas y organizativas donde se describen los estándares o recomendaciones de Internet (originalmente ARPANET), y están hechos para hacer compatibles los programas entre sí y que se pueda usar diferente software para la misma función. Definen protocolos y lenguajes, se garantiza la interoperabilidad entre sistemas si ambos cumplen el mismo RFC.

Samba: es una implementación libre del protocolo de archivos compartidos de Microsoft Windows (antiguamente llamado SMB, renombrado recientemente a CIFS) para sistemas de tipo UNIX.

SMB (Server Message Block): es un protocolo de red que permite compartir archivos e impresoras (entre otras cosas) entre nodos de una red. Es utilizado principalmente en ordenadores con Microsoft Windows.

System V: es una variante del sistema operativo Unix desarrollada por AT&T. Fue la primera versión de UNIX. En ella se estandarizó la mayoría de los elementos del sistema operativo.

TCP/IP: es un conjunto de protocolos de red que implementa la pila de protocolos en la que se basa Internet y que permiten la transmisión de datos entre redes de computadoras.

UDP (User Datagram Protocol): es un protocolo del nivel de transporte basado en el intercambio de datagramas, permite el envío de datagramas a través de la red sin que se haya establecido previamente una conexión, ya que el propio datagrama incorpora suficiente información de direccionamiento en su cabecera. Tampoco tiene confirmación, ni control de flujo, por lo que los paquetes pueden adelantarse unos a otros; y tampoco sabemos si ha llegado correctamente, ya que no hay confirmación de entrega o de recepción.

Umsdos: es un sistema de ficheros de Linux sobre FAT. Ofrece una alternativa al sistema de ficheros ext. Su objetivo principal es conseguir una más fácil coexistencia con los datos de una partición FAT, compartiéndola.

Unix: es un sistema operativo portable, multitarea y multiusuario; desarrollado en principio por un grupo de empleados de los laboratorios Bell de AT&T, entre los que figuran Ken Thompson, Dennis Ritchie y Douglas McIlroy.

Windows: es un sistema operativo gráfico para computadoras personales propiedad de la empresa Microsoft.

WINS (Windows Internet Naming Service): es un servidor de nombres de Microsoft para NetBIOS, que mantiene una tabla con la correspondencia entre direcciones IP y nombres NetBIOS de ordenadores. Esta lista permite localizar rápidamente a otro ordenador de la red.

Xenix: era un sistema operativo tipo Unix desarrollado por Microsoft. Microsoft lo llamó así debido a que no tenía licencia para utilizar el nombre Unix.

XFree86: es una implementación del sistema X Window System. Fue escrita originalmente para sistemas operativos UNIX funcionando en ordenadores compatibles IBM PC. En la actualidad está disponible para muchos otros sistemas y plataformas.