

Servidores de Nombre - DNS

En las redes **TCP/IP** cada interfaz de red es identificada a través de una dirección **IP** única de 32 bits. A cada interfaz de red se le puede asociar un nombre ó **hostname**. Los nombres son asignados a los dispositivos puesto que son más fáciles de manejar que los números **IP**, es decir, permiten a las personas un uso más amigable de la red.

En la mayoría de los casos se pueden utilizar indiferentemente los nombres o las direcciones. Por ejemplo al ejecutar el comando **telnet**, se podría utilizar una dirección **IP**:

```
> telnet 150.185.131.1
```

ó utilizar el nombre asociado a esta dirección:

```
> telnet merlin.cecalc.ula.ve
```

Cuando un comando es introducido con una dirección **IP** ó con un nombre, la conexión siempre se realiza utilizando la dirección **IP**. Por esta razón el sistema debe convertir el nombre a la dirección **IP**, de forma transparente al usuario, antes de realizar la conexión.

Existen dos técnicas para traducir nombres a direcciones: a) mediante la utilización de una tabla llamada **host table**; y b) mediante la utilización de un sistema de base de datos distribuida llamada **Domain Name Service (DNS)**.

La tabla **host table**, es un archivo de texto que asocia direcciones **IP** a nombres. En la mayoría de los sistemas **UNIX** esta tabla se encuentra en el archivo **/etc/hosts**. Esta técnica era utilizada en los comienzos de lo que es hoy **Internet**. Para ese tiempo existía una comunidad pequeña de unos cientos de máquinas, lo que permitía que se mantuviese toda la información necesaria de cada máquina en un solo archivo.

Sin embargo, con la utilización de los protocolos **TCP/IP** la población de la red aumento considerablemente y con ello surgieron los siguiente problemas:

- **Carencia de escalabilidad.** La tabla que mantenía la equivalencia de nombres a direcciones **IP** creció de tal forma que se convirtió en una forma ineficiente para resolver el problema.
- **Carencia de un proceso automático de actualización.** Las máquinas registradas recientemente, pueden ser referenciadas por su nombre sólo cuando el sitio recibe la actualización de la tabla. Sin embargo no hay forma de garantizar que la **host table** sea distribuida a los sitios. Antes de adaptar **DNS**, el **Network Information Center (NIC)** era el organismo encargado de mantener la información de todas las máquinas registradas. El **NIC** no sabía cual sitio tenía la versión actualizada de la tabla y cual no. Esta falta de consistencia en la **host table** es la mayor debilidad de esta técnica.

Las debilidades inherentes a la **host table** son solventadas con la utilización del **DNS**. El sistema de dominio de nombres (**Domain Name System**) es una base de datos distribuida, la cual forma un sistema jerárquico para traducir de nombres (**hostnames**) a direcciones **IP**. En el **DNS** no existe una base de datos central con toda la información de los **hosts** de **Internet**. Por el contrario la información es distribuida entre cientos de **servidores de nombres (name servers)**. Esto permite controlar por segmentos toda la base de datos en general, logrando que la información de cada uno de estos segmentos este disponible a través de toda la red utilizando un esquema **cliente - servidor**.

Los programas llamados **name servers** (servidores de nombres) forma la parte del servidor en el mecanismo **cliente-servidor** de **DNS**. Los **name servers** contienen la información de un segmento de la base de datos y la ponen a disposición de los clientes.

Los clientes son llamados **resolvers**, los cuales no son más que rutinas de librería que crean preguntas y las envían a través de la red a los servidores.

La estructura de **DNS** se asemeja a la estructura jerárquica de los sistemas de archivos de **UNIX**, la cual se representa con un árbol invertido. El tope de esta jerarquía se representa por un punto “.” y cada nodo del árbol, en general, representa una partición de la base de datos. Cada una de estas particiones es llamada **domain** (dominio), los cuales pueden a su vez ser divididos en **subdomains** (subdominios). En la Fig. 2-1, se puede apreciar parte de la estructura de **DNS**.

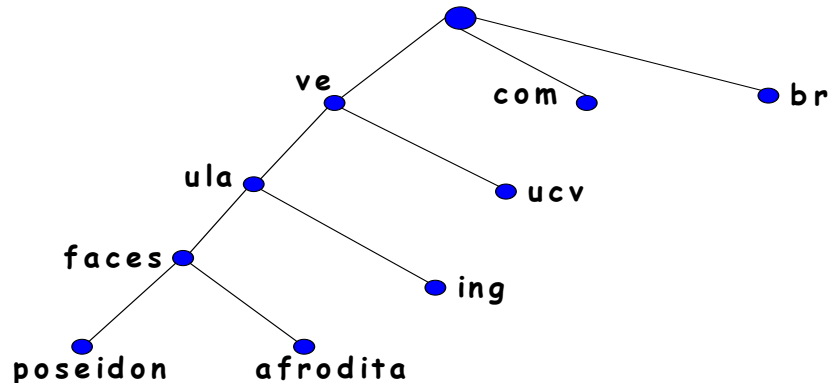


Fig. 2-1 Estructura del **DNS**

Dominios

Cada unidad de datos en el **DNS** está indicada por un nombre. Estos nombres son esencialmente rutas en el árbol invertido, llamado espacio de dominio de nombres. Cada nodo en el árbol es etiquetado con un nombre simple, el cual puede tener hasta 63 caracteres de longitud. El dominio raíz tiene una etiqueta de tamaño cero la cual es reservada. Un nombre completo de cualquier nodo en el árbol es la secuencia de etiquetas separadas por “.” las cuales se encuentran en el camino del nodo hasta la raíz.

Por conveniencia cuando el dominio raíz aparece por si mismo, este es escrito con un punto “.”. De esta forma cuando se escribe un nombre de dominio que termina en punto este es interpretado como absoluto. Por el contrario cuando se escribe un nombre que no termina en punto, este es interpretado como relativo a otro dominio diferente al

dominio raíz. Un nombre de dominio absoluto es también referido como nombre de dominio completamente calificado (**fully-qualified domain name**), algunas veces abreviado **FQDN**. En la Fig. 2-1, se muestran dos nodos debajo del dominio **faces: afroditia y poseidon** los cuales corresponden a dos máquinas que pertenecen al dominio **faces.ula.ve.** ; el punto al final del dominio indica que es **FQDN**. Así, los nombres **FQDN** son **afroditia.faces.ula.ve.** y **poseidon.faces.ula.ve.** .

Directamente bajo el dominio raíz está los dominios de nivel superior (**top-level domains**). Existen básicamente dos **tipos** de dominios de nivel superior: **geográficos** y **organizacionales**. Los dominios geográficos son identificados por dos letras y son asignados a cada país del mundo. Por ejemplo, y como se muestra en la Fig. 2-1 Venezuela es el dominio **ve** y Brasil es **br**.

Los dominios organizacionales están basados como su nombre lo indican en el **tipo** de organización (comercial, militar, etc.) a la cual pertenece el sistema. Estos dominios son los siguientes:

- **com.** Organizaciones comerciales como **sun.com, sybase.com.**
- **edu.** Instituciones educativas como **mit.edu, berkeley.edu**
- **gov.** Agencias gubernamentales como **nasa.gov**
- **mil.** Organizaciones militares como **navy.mil**
- **net .** Organizaciones relacionadas con la red como **freshmeat.net**
- **org.** Organizaciones que no entran en ninguna de las categorías anteriores
como son las organizaciones sin fines de lucro, ejemplo: **0**

Recientemente se han realizado algunas propuestas para incrementar el número de dominios de nivel superior. Los dominios propuestos son llamados dominios de nivel superior genéricos (**generic top level domains**) o **gTLDs**.

La proposición más conocida fue hecha por el **IAHC (International Ad Hoc Committee)**. El **IAHC** propuso los siguientes nuevos **gTLDs**:

- **film.** Negocios o filmaciones.
- **store.** Negocios que ofrecen bienes.
- **web.** Organizaciones que hace énfasis en el WEB.
- **arts.** Organizaciones culturales y de entretenimiento.
- **rec.** Organizaciones recreacionales y de entretenimiento.
- **info.** Entidades que proveen servicios de información.
- **nom.** Individuos u organizaciones que desean definir una nomenclatura personal.

Delegación

La delegación de dominios permite alcanzar uno de los objetivos primordiales del **DNS**; la **administración descentralizada**. Una organización que administra un dominio lo puede dividir en subdominios y delegarlos a otras organizaciones. De esta forma cada organización que maneja un subdominio se encarga de mantener toda la data correspondiente a ese subdominio. Ellos pueden cambiar libremente la data y incluso dividir su subdominio en más subdominios. De esta manera el **dominio padre** en lugar de contener información acerca del subdominio que esta delegando, contiene sólo apuntadores a la fuente de la data del subdominio, es decir, a los servidores de nombres. De esta manera si un servidor de nombres de un dominio es interrogado acerca de una máquina que pertenece a uno de sus subdominios, este devolverá la dirección del servidor que le puede contestar.

Como ejemplo de la delegación, considere el dominio **ula.ve**. En este dominio se realizan varias delegaciones tal como **ing.ula.ve**. De esta manera los administradores de la facultad de ingeniería están encargados de agregar, eliminar y modificar los datos del nuevo dominio independientemente de los datos que se tienen en el dominio padre, **ula.ve**.

Servidores de Nombres (Name Servers)

Los programas que guardan la información acerca del espacio de dominio de nombres son llamados **servidores de nombres (name servers)**. Los servidores generalmente mantienen la información completa acerca de una parte del espacio de dominio de nombres llamado zona. Se dice entonces que el servidor de nombres tiene autoridad para esa zona.

El término zona es algunas veces usado indiferentemente con la palabra dominio, pero debe hacerse una distinción entre estos términos. Una zona se refiere a la información que contiene el archivo de datos, mientras el término dominio se utiliza en un contexto más general, es decir un dominio es parte de la jerarquía de dominios identificada por un nombre de dominio.

Las especificaciones de **DNS** definen dos tipos de servidores de nombres: **maestros primarios** y **maestros secundarios**. Un servidor **primario maestro** obtiene los datos de la zona sobre la cual tiene autoridad desde archivos que están en la misma máquina del servidor de nombres, mientras que un **maestro secundario** obtiene los datos de la zona de otro servidor autorizado para la zona; esto es llamado transferencia de zona.

Archivos de Datos

Los datos asociados con cada dominio de nombres esta contenida en los llamados registro de recursos (**resource records**) o simplemente **RR**. Los **RR** describen todos los **hosts** en la zona y marca toda delegación de subdominios.

Los archivos que los servidores de nombres primarios utilizan son llamados archivos de datos (**data files**). Estos archivos de datos contienen registro de recursos que describen la zona.

Resolvers

Resolvers son los clientes que acceden a los servidores de nombres. Los programas que necesitan la información de un dominio de espacio de nombres utilizan el **resolver**. Estos a su vez realizan las siguientes tareas:

- Interrogan al servidor de nombres.
- Interpretan las respuestas (las cuales pueden ser **RR** o errores).
- Devuelven la información al programa que la solicita.

Resolución de nombres

Los servidores de nombres tienden a buscar datos en el espacio de dominio de nombres. Tienen que comportarse de esa manera dada la inteligencia limitada de los **resolvers**. No sólo pueden dar datos acerca de la zona (puede ser más de una) sobre la que tienen autoridad, sino que pueden buscar a través del espacio del dominio de nombres para encontrar datos sobre los cuales no poseen autoridad. A este proceso se le conoce como **resolución**.

Debido a que el espacio de nombres es estructurado, un servidor de nombres solo necesita una pieza de información para encontrar el camino a cualquier punto en el árbol, los nombres y direcciones de los servidores de nombre raíz.

Servidores Raíz Primarios (Root Name Servers, RNS)

Los **RNS** saben cuáles servidores de nombres tienen autoridad para los dominios superiores. Si se les hace una pregunta acerca de un subdominio, los servidores raíz maestros pueden al menos proveer los nombres y direcciones de los servidores de nombres con autoridad para el segundo nivel de dominios a los cuales un dominio pertenece. Cada servidor interrogado da, al que pregunta, información de cómo “estar más cerca” de la respuesta que está buscando o provee él mismo una respuesta. Lo que hacen los **RNS** es proveer punteros desde los dominios superiores a los servidores de nombres de los dominios inferiores. Por ejemplo para conseguir el servidor de nombres del dominio **ve**, se debe interrogar a los servidores raíz.

Los **RNS**, así como los **NS** normales, son muy importantes en la resolución de un nombre dentro de un dominio particular. Debido a que son tan importantes, **DNS** provee mecanismos para asegurar siempre el servicio utilizando redundancia (servidores secundarios) o aliviando la carga de los servidores primarios y **root** (usando **caching**). Sin embargo, en ausencia de mecanismos como el caching, la resolución debe empezar en los servidores de raíz maestros.

Métodos de búsqueda

En el momento que un cliente desea obtener la dirección **IP** de una máquina, interroga al servidor de nombres de su dominio. Luego el servidor de nombres verifica sus tablas de máquinas a ver si allí consigue el nombre por el cual le están preguntando. Si es así, entonces retorna la dirección **IP** asociada con ese nombre. Si la información pertenece a otro dominio, el servidor de nombres busca en su **cache** y si no está allí entonces comienza el proceso de resolución que se puede comportar de las dos formas siguientes:

- **Recurrente.** Un servidor de nombres envía una respuesta recurrente *cuando es el servidor y no el cliente el que pregunta a otros servidores de nombres por la información solicitada del dominio*. Esto ocurre cuando el servidor de nombres sabe que **el resolver** no tiene la inteligencia para manejar una referencia a otro servidor de nombres (Es decir, **el resolver** hace explícitamente una pregunta recurrente). A medida que un servidor de nombres pregunta (obtenga respuesta o no) va guardando los nombres encontrados en su **cache** para evitarse búsquedas innecesarias.
- **Iterativa.** El servidor de nombres *da la mejor respuesta*, que ya sabe, a quien preguntó (es decir, da una referencia al servidor de nombres más cercano a la información de dominio interrogado). Primero consulta sus datos locales, si no está allí busca entonces en su **cache** y si aún no encuentra nada devuelve la respuesta al servidor más cercano al dominio buscado. Si el servidor falla, no lo vuelve a reintentar.

Las bibliotecas del **resolver** hacen búsquedas recurrentes e iterativas, mientras que entre servidores de nombres solo se hacen búsquedas iterativas.

Equivalencia de direcciones a nombres

Hemos visto como convertir nombres a direcciones, pero ahora queremos saber como se convierte una dirección **IP** a nombres. La equivalencia de direcciones a nombres (**mapping**) es útil para salidas que sean fáciles de leer por los seres humanos, fáciles de interpretar en bitácoras del sistema (**log files**) y como una forma de autenticación (por ejemplo, el archivo **.rhosts** y **host.equiv** bajo UNIX, además de ser usados por ciertos servidores **FTP**).

Cuando se usan las tablas de **hosts** de las máquinas (**hosts tables**), la conversión es fácil. Requiere de una búsqueda secuencial a lo largo de la tabla usando una dirección. En **DNS**, sin embargo, el espacio de dominio de nombres esta indexado por nombres y no por números (como es el caso de una dirección **IP**). **DNS** soluciona esto valiéndose de un dominio el cual usa números como nombres, el dominio **in-addr.arpa**.

Los nodos en el dominio **in-addr.arpa** son nombrados después de los números en una representación de octetos separados por puntos (Recuerde que una dirección **IP** tiene la forma: **octeto. octeto. octeto. octeto**, para un total de 32 bits, con cada octeto en el rango de 0 a 255).

En este dominio la dirección **IP** se lee desde lo más específico a lo más general; por ejemplo, **berlioz.ing.ula.ve** (150.185.146.42) se leería 42.146.185.150.**in-addr.arpa**, lo cual retorna en una búsqueda a **berlioz.ing.ula.ve**.

La razón por la que se escribe así es porque una dirección **IP** también es jerárquica (en nuestro ejemplo 42 corresponde al número de la máquina y 146.185.150 corresponde a la dirección de red).

Caching

Podría parecer que el proceso de buscar un nombre es sumamente lento, sin embargo no es así. Una de las razones de la rapidez es el uso de **caching**.

Este mecanismo trabaja de la siguiente manera: Un servidor de nombres que está ejecutando una búsqueda recurrente podría enviar unas cuantas preguntas para encontrar una respuesta acerca de un dominio. Sin embargo, el servidor descubre información acerca del nombre del dominio a medida que explora. Cada vez que es referido a otro servidor, aprende que esos servidores son autoridades de las zonas interrogadas y aprende esas direcciones. Si encuentra el dato buscado, lo guarda para usarlo en una futura referencia. La próxima vez que un **resolver** haga una pregunta acerca de un nombre de un dominio que el servidor conozca, el proceso es acertado, ya que el servidor primero revisará en su **cache** para dar la respuesta.

El **cache** solamente se guarda en memoria temporal la cual se borra cuando el servidor reactualiza su memoria (por ejemplo, cuando se apaga la máquina en la que corre el servidor).

Tiempo de vida (TTL -Time To Live)

Estos tiempos son los que le dicen a un servidor secundario cuanto tiempo debe mantener en memoria sus datos antes de buscar datos actualizados del servidor maestro.

Los servidores de nombres no mantienen los datos en **cache** por siempre. El administrador de una zona decide el tiempo de vida de los datos buscando un balance entre la veracidad de la información y la cantidad de tiempo perdido en transferir una zona. Una vez que el tiempo de vida expira, el servidor busca de nuevo los datos del dominio del cual es servidor secundario.

Configuración de BIND

En UNIX, **DNS** es implementado por el **Berkeley Internet Name Domain (BIND)**. Existen varias versiones de BIND: 8. En esta sección se explicará la configuración del cliente, de los servidores y de los archivos de datos.

Configuración de los clientes (Resolver)

Los clientes son configurados en el archivo `/etc/resolv.conf`. El cliente no es un proceso distinto y separado, es una librería de rutinas llamadas por los procesos de red. El archivo `resolv.conf` es un archivo de texto simple. Los principales comandos utilizados en este archivo de configuración son:

- **nameserver address.** La entrada **nameserver** identifica, por el número **IP**, los servidores a los cuales el cliente les va a realizar las preguntas. Los servidores de nombres son consultados en el orden que aparecen en el archivo.
- **domain name.** La entrada **domain** define el nombre del dominio por omisión. El cliente agrega el nombre de dominio por omisión a cualquier **hostname** que no contenga un punto “.”.

- **search domain ...** Esta entrada define una serie de dominios que son utilizados cuando un **hostname** no contiene un punto “.”.
- **options option ...** Esta entrada permite seleccionar algunas opciones extra para el **resolver**.

Un ejemplo típico de archivo de configuración se presenta a continuación:

```
;Archivo de configuración para el Resolver

;Se define el dominio por omisión
domain cecalcul.ula.ve

;Se define otros dominios de búsqueda
search ula.ve ciens.ula.ve faces.ula.ve ing.ula.ve

; Se definen los servidores de nombre
nameserver 150.185.138.1

nameserver 150.185.130.8

nameserver 150.185.128.2
```

Configuración De Los Servidores

El lado servidor de **DNS** es un demonio llamado **named**. Existen tres **tipos** de configuración diferentes de servidores de nombres los cuales requieren que el sistema local ejecute el software **named**.

- **Primario.** El servidor primario es la fuente autorizada de toda la información acerca de un dominio específico. Él carga la información de un archivo mantenido localmente por el administrador. Este archivo (archivo de zona) contiene la información más precisa acerca de una porción de la jerarquía de dominios sobre la cual el servidor tiene autoridad. La configuración de un servidor primario requiere un conjunto de archivos: archivos de zona para el dominio regular y para el dominio reverso, el archivo de configuración del **servidor**, el archivo de **cache** y el archivo **loopback**.
- **Secundario.** Un servidor secundario transfiere un conjunto completo de información de dominio desde el servidor primario. El archivo de zona es transferido desde el servidor primario y es guardado como un archivo local de disco (a esta operación se le llama *transferencia de zona*). Solamente se requieren el archivo de inicio, el archivo de **cache** y el archivo **loopback**. Un servidor secundario es considerado también primario ya que tiene una copia exacta de los archivos del servidor primario, lo cual lo hace autoridad.
- **Sólo Cache.** Un servidor de sólo **cache** corre el **software** del servidor, pero no tiene los archivos de base de datos del servidor. Aprende las respuestas de otros servidores de nombres, las guarda y las usa para responder preguntas futuras sobre esa misma información. Solamente requiere de un archivo de **cache** (con información acerca de los **root servers** a los cuales debe preguntar). Se dice que

este tipo de servidor no es autoritario ya que la información que obtiene es de segunda mano.

El archivo de configuración del servidor mantiene los parámetros de funcionamiento, apuntes a los archivos de datos del dominio y direcciones de servidores remotos. Este archivo de configuración difiere entre las dos vertientes de **BIND**: 4 (versiones 4.x.x), 8 (versiones 8.x.x) y 9 (versión 9.x.x). En primer lugar se mostrara la forma de configurar la versión BIND 4.

Configuración Para BIND 8

El archivo de configuración del servidor **DNS** en la implementación **BIND 8** se llama **named.conf**. Esta implementación es mucho más configurable y flexible que la versión 4. Esta nueva versión introduce nuevas opciones como listas de acceso, y además permite aplicar estas opciones selectivamente a cada una de las zonas. Estos cambios realizados en la configuración del servidor requieren de una nueva sintaxis en el archivo de configuración. El archivo de configuración de esta versión consiste en una serie de declaraciones algunas de las cuales contienen a su vez un bloque interno de subdeclaraciones. Las declaraciones permitidas son las siguientes:

- **acl**. Define una lista de dir. **IP** para control de acceso y otros usos.
- **include**. Incluye un archivo.
- **key**. Especifica un identificador clave, el cual puede ser usado en la declaración **server** para asociar un método de autenticación con un servidor en particular.
- **logging**. Especifica que se debe guardar el servidor en la bitácora y además, donde debe estar esta bitácora.
- **options**. Controla las opciones de configuración global del servidor. Las principales opciones son las siguientes:
 - **directory**. El directorio de trabajo.
 - **named-xfer**. Ubicación del archivo empleado en la transferencia de zona.
 - **pid-file**. Archivo con el **PID** del servidor.
 - **statistics-file**. Archivo con las estadísticas.
 - **notify**. Indica si se debe enviar una notificación a los servidores secundarios cuando ocurre un cambio en los archivos de zona.
 - **transfers-in**. Indica el número máximo de transferencias de zona simultáneas.
 - **datasize**. El tamaño máximo de la memoria para almacenar data.
 - **stacksize**. El tamaño máximo de la memoria empleada para la pila.
 - **coresize**. El tamaño de un **coredump**.

- **files.** El número de archivos abiertos concurrentemente.
- **server.** Define las características asociadas a un servidor de nombres.
- **zone.** Define una zona. En esta declaración se indica el tipo de servidor de la zona, es decir si es **master** (primario), **slave** (secundario) o **hint** (caché), además de ciertas opciones adicionales.

Además de estas declaraciones, el archivo de configuración permite comentarios al estilo de C++; es decir, se puede utilizar */** y **/* para encerrar un comentario; o se puede utilizar *//* para colocar un comentario hasta el final de la línea.

Veamos ahora un ejemplo de un archivo de configuración común:

// Archivo de configuración para un servidor primario y

// Secundario.

```
acl DNS_srvrs0 {
    150.185.176.1;
    150.185.177.1;
    localhost;
};

options {
    directory "/etc/named";
    named-xfer "/usr/sbin/named-xfer";
    pid-file "named.pid"

    statistics-file "named.stats";

    notify yes;

    transfers-in 10;

    datasize default;
    stacksize default;

    coresize 0;

    files unlimited;

};

logging {
```

```

        channel syslog_errors {
            syslog daemon;

            severity error;

        };
    };

zone "0.0.127.in-addr.arpa" in {
    type master;

    file "db.127.0.0";

};

zone "eslared1.ula.ve" in {
    type master;

    allow-transfer { DNS_srvrs0; };

    file "db.eslared1.ula.ve";

};

zone "176.185.150.in-addr.arpa" in {
    type master;

    allow-transfer { DNS_srvrs0; };

    file "db.150.185.176";

};

zone "eslared2.ula.ve" in {
    type slave;

    allow-transfer { DNS_srvrs0; };

    file "db.eslared2.ula.ve";

    masters {
        150.185.177.1;
    };

};

zone "177.185.150.in-addr.arpa" in {

```

```

        type slave;

        allow-transfer { DNS_srvrs0; };

        file "db.150.185.177";

        masters {
                150.185.177.1;
        };
};

zone "." in {
        type hint;
        file "root.cache";
};

```

En este ejemplo se muestran la configuración para un servidor que es primario para el dominio **eslared1.ula.ve** y del dominio reverso **150.185.176**, y es secundario para el dominio **eslared2.ula.ve** y para el dominio reverso **150.185.177** cuyo servidor primario es **150.185.177.1**. El directorio donde se guardan los archivos de zona es **/etc/named**.

Archivos de Datos

Todos los archivos de datos que utilizan los servidores de nombres poseen un mismo formato básico y utilizan el mismo tipo de registros; además, son comunes para las versiones 4 y 8. Ellos utilizan los registros de recursos estándar llamados **RRs**. La descripción de estos registros se muestra a continuación:

SOA	(Start Of Authority)	Marca el comienzo de los datos de la zona y define los parámetros que afectan la zona.
NS	(Name Server)	Identifica un servidor de nombres de dominio.
A	(Address)	Convierte un nombre de máquina a una dirección.
PTR	(Pointer)	Convierte una dirección a un nombre de máquina.
MX	(Mail Exchange)	Identifica donde deben ser entregados los correos de un nombre de dominio dado.
CNAME	(Canonical Name)	Define un alias para una máquina.
HINFO	(Host Information)	Describe el hardware y el OS de la máquina.
TXT	(Text)	Guarda texto arbitrario.

Veamos ahora ejemplos para los archivos de datos indicados en la configuración de los archivos del servidor. Todos los servidores deben tener un archivo que realice la traducción del dominio reverso.

db.127.0.0 (dominio reverso 0.0.127.in-addr.arpa):

```
;db.127.0.0
;
@ IN SOA leon.eslared1.ula.ve. root.leon.eslared1.ula.ve. (
    210399 ;serial
    10800 ;refresh every 3 hours
    3600 ;retry after 1 hour
    604800 ;expire after 7 days
    86400 ;ttl 1 day
)
IN NS leon.eslared1.ula.ve.
1 IN PTR localhost.
```

db.eslared1.ula.ve:

```
;db.eslared1.ula.ve
;
@ IN SOA leon.eslared1.ula.ve. root.leon.eslared1.ula.ve. (
    210399 ;serial
    10800 ;actualizar cada 3 horas
    3600 ;reintentar después de 1 hora
    604800 ;caducan después de 1 semana
    86400 ;tiempo de vida de los datos
    ; 1 día.
)
; Se definen los servidores para el dominio
IN NS leon.eslared1.ula.ve.
IN NS tigre.eslared2.ula.ve.
```

```

; Se definen los intercambiadores de correos
                IN    MX    5    leon.eslared1.ula.ve.

; Se definen los host en la zona
leon            IN    A    150.185.176.1
oso            IN    A    150.185.176.2
elefante       IN    A    150.185.176.3
router1        IN    A    150.185.176.254
www            IN    CNAME elefante

; Se Definen los subdominios
grupo1         IN    NS    aguila.grupo1.eslared1.ula.ve.
grupo2         IN    NS    buitre.grupo2.eslared2.ula.ve.

; Registros Glue para los servidores dentro de los dominios
aguila.grupo1.eslared1.ula.ve. IN    A    150.185.176.10
buitre.grupo2.eslared2.ula.ve. IN    A    150.185.176.30

```

En el registro **SOA** se especifican la máquina donde están los datos y el responsable (en este caso root). @ es una abreviatura para el dominio, en nuestro caso **eslared1.ula.ve**. Los tiempos (en segundos) definidos en este registro le indican a los servidores secundarios cuando actualizar los datos que ellos poseen. Observe como todos los nombres son completamente calificados (terminan con “.”).

En los registros **MX** se especifican por prioridad (número menor indica mayor prioridad) los servidores de correo para el dominio. En el ejemplo presentado se crearon dos nuevos dominio: grupo1 y grupo2 y se delega la autoridad de estos dominios a las máquinas **águila** y **buitre** respectivamente.

db.176.185.150.in-addr-arpa:

```

;db.176.185.150.in-addr.arpa.
;
@ IN    SOA    leon.eslared1.ula.ve. root.leon.eslared1.ula.ve. (
                210399    ;serial
                10800    ;actualizar cada 3 horas
                3600     ;reintentar después de 1 hora
                604800   ;caducan después de 1 semana

```

```

                                86400      ; tiempo de vida de los datos
                                ; 1 día.
                                )
IN      NS      leon.eslared1.ula.ve.
IN      NS      tigre.eslared2.ula.ve.

1       IN      PTR      leon.eslared1.ula.ve.
2       IN      PTR      oso.eslared1.ula.ve.
3       IN      PTR      elefante.eslared1.ula.ve.
10      IN      PTR      aguila.grupo1.eslared1.ula.ve.
30      IN      PTR      buitres.grupo1.eslared1.ula.ve.
254     IN      PTR      router1.eslared1.ula.ve.

```

Los subdominios en el dominio *in-addr.arpa* no son tan comunes o útiles como los subdominios en el espacio de nombres de máquinas, puesto que la estructura de los nombres y de las direcciones **IP** son distintas. Cuando una dirección **IP** es llevada al dominio *in-addr.arpa*, los cuatro bytes que conforman la dirección son tratados como piezas distintas. Cuando se utilizan subredes, las máscaras son orientadas a bits, los subdominios en *in-addr.arpa* pueden ocurrir sólo a nivel de bytes, lo cual indica que la delegación de subdominios se realiza a nivel de bytes. Por ejemplo el dominio **176.185.150.in-addr.arpa** solamente puede delegar a un dominio que solo contenga una dirección **IP** como por ejemplo: **125.176.185.150.in-addr.arpa**.

root.cache:

```

.                3600000  IN  NS      A.ROOT-SERVERS.NET.
A.ROOT-SERVERS.NET. 3600000  IN  A        198.41.0.4
.                3600000  IN  NS      B.ROOT-SERVERS.NET.
B.ROOT-SERVERS.NET. 3600000  IN  A        128.9.0.107
.                3600000  IN  NS      C.ROOT-SERVERS.NET.
C.ROOT-SERVERS.NET. 3600000  IN  A        192.33.4.12
.                3600000  IN  NS      D.ROOT-SERVERS.NET.
D.ROOT-SERVERS.NET. 3600000  IN  A        128.8.10.90
.                3600000  IN  NS      E.ROOT-SERVERS.NET.
E.ROOT-SERVERS.NET. 3600000  IN  A        192.203.230.10
.                3600000  IN  NS      F.ROOT-SERVERS.NET.
F.ROOT-SERVERS.NET. 3600000  IN  A        192.5.5.241
.                3600000  IN  NS      G.ROOT-SERVERS.NET.
G.ROOT-SERVERS.NET. 3600000  IN  A        192.112.36.4
.                3600000  IN  NS      H.ROOT-SERVERS.NET.
H.ROOT-SERVERS.NET. 3600000  IN  A        128.63.2.53
.                3600000  IN  NS      I.ROOT-SERVERS.NET.
I.ROOT-SERVERS.NET. 3600000  IN  A        192.36.148.17

```

.	3600000	IN	NS	J.ROOT-SERVERS.NET.
J.ROOT-SERVERS.NET.	3600000	IN	A	198.41.0.10
.	3600000	IN	NS	K.ROOT-SERVERS.NET.
K.ROOT-SERVERS.NET.	3600000	IN	A	193.0.14.129
.	3600000	IN	NS	L.ROOT-SERVERS.NET.
L.ROOT-SERVERS.NET.	3600000	IN	A	198.32.64.12
.	3600000	IN	NS	M.ROOT-SERVERS.NET.
M.ROOT-SERVERS.NET.	3600000	IN	A	202.12.27.33

HOST

Es una herramienta de depuración que permite interrogar directamente un servidor de nombres y conseguir cualquier información conocida en **DNS**. Es de gran ayuda para determinar si un servidor está bien configurado y se está ejecutando correctamente. Este programa se utiliza para resolver preguntas directamente desde la línea de comandos. Por ejemplo:

```
%host tigre.eslared2.ula.ve

merlin.cecalc.ula.ve has address 150.185.138.126

%
```

En este ejemplo el usuario le pregunta al **servidor DNS** sobre la dirección de merlin.cecalc.ula.ve. **host** muestra el nombre y la dirección de la estación.

Veamos otro ejemplo utilizando la herramienta de forma interactiva:

```
%host luna.cecalc.ula.ve

Host luna.cecalc.ula.ve not found: 3(NXDOMAIN)

%
```

En este caso la respuesta fue negativa; esto quiere decir que la máquina **luna.cecalc.ula.ve** no está registrada en el **DNS**.