# DNS (Domain Name System).


*Práctica desarrollada para la III era Escuela Latinoamericana de Redes*

*Hecho por José Vicente Núñez Zuleta*

*Email: jose@ing.ula.ve*

## Introducción y copyright

DNS, versión 1.0.

Hice esta práctica para la práctica de DNS en la IIIera Escuela Latinoamericana de Redes (Eslared III). Ahora la pongo a disposición de los administradores de redes en internet con la esperanza de que encuentren en este documento una fuente de información útil en la resolución de sus problemas.

Esta guía siempre estará bajo construcción y puede contener errores.

Esta guia viene sin garantías de ningún tipo, y se prohibe su uso para fines comerciales.



Si tiene dudas, adiciones, comentarios acerca de esta guia puede contactarme en la siguiente dirección:

Jose@ing.ula.ve

La última versión de este documento puede ser conseguida en:
http://www.ing.ula.ve/~jose

## Agradecimientos

Quisiera agradecer a las siguientes personas por sus correciones y sugerencias en la elaboración de este documento:

- Daniel Pradilla (daniel@ing.ula.ve)
- Nestor Angulo Reina (nestor@ing.ula.ve)
- Ermanno Pietrosemoli (ermanno@ingula.ve)
- Edmundo Vitale (evitale@ing.ula.ve)
- Mónica Venegas (monicav@ing.ula.ve)
- Alpha Pernia (alpha@ing.ula.ve)
- Gustavo Camargo (tavo@ula.ve)
- Leonardo Contreras (leoc@ula.ve)
- Carlos Barrios (barrios@ula.ve)
- Pauly Quintero (pauly@ula.ve)
- Carlos Navas (navac@ula.ve)
- Vanessa Hamar (vanessa@ula.ve)
- Natalia Pietrosemoli (natalia@ula.ve)

## Historia del DNS:

A mediados de los 70, la *ARPANET* era una comunidad pequeña y amistosa de unas 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 la *ARPANET*.

El archivo */etc/hosts* se obtenía entonces del archivo ***HOSTS.TXT*** (el cual tenía información adicional que no era necesaria).

Sin embargo, cuando la 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 tornaba *ineficiente* a medida que aumentaban el número de máquinas.

Es allí donde entra **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 lograba 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 esta 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 root (el cual se denota con un punto "."). Cada uno de esos dominios a su vez pueden 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 (llamdo oficilamente *de 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 del host, pasando por cada uno de los "dominos" 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 aliases 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).

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 dominos separados).
- Elimina el problema de carga y tráfico de red en una sola máquina ya que la información esta distribuida. (Y esta disponible de manera redundante).
- Finalmente 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).

En esta práctica trabajaremos con la implementación de UNIX de DNS: *BIND* (*Berkeley Internet Name Domain*), la cual es la implementación más popular de DNS hasta la fecha .

## Como trabaja DNS:

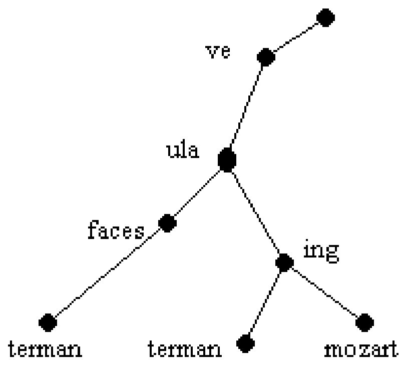
Antes de comenzar a profundizar en como trabaja DNS es necesario conocer los siguientes conceptos:

- Dominios (La estructura básica de un dominio DNS) y sus tipos.
- Delegación de Dominios (Cómo descentralizar DNS, cómo crear subdominios).
- Registros de Recursos (RR, o dónde guarda la información DNS).
- Servidores de Nombres, tipos de servidores de nombres.
- Resolver (Las bibliotecas que buscan la información por el lado del cliente)

## Dominios:

En realidad un dominio es sólo un índice dentro de la base de datos de DNS. Un dominio puede ser una máquina o puede ser un nodo del cual pueden partir otros dominios (o ambas cosas a la vez).

Por ejemplo, el siguiente árbol invertido me muestra un dominio típico de DNS:



Cada unidad de datos en DNS está indizada por nombre (recuerde que es una base de datos). Estos nombres son esencialmente rutas en un gran árbol invertido, llamado el espacio de dominio de nombres. El árbol de DNS puede ramificarse de diversas maneras en cada punto de intersección, llamado nodo. Cada nodo puede tener hasta 63 caracteres de longitud. El dominio raíz tiene una etiqueta (de tamaño cero), la cual es reservada. El nombre completo de un nombre de dominio es una secuencia de etiquetas en la ruta desde ese nodo hasta la raíz. Cuando el dominio raíz aparece por si mismo es denominado "." por conveniención. De manera que cuando alguien escribe una dirección terminada con un punto ésta es interpretada como una dirección absoluta.

En nuestro ejemplo vemos que hay dos nodos debajo del nodo ing en el árbol invertido: terman y mozart. Terman y Mozart son dos máquinas las cuales pertenecen al dominio ing.ula.ve. (Note el punto al final del dominio). Es decir, sus *nombres completamente calificados* son terman.ing.ula.ve. y mozart.ing.ula.ve. Vemos también como es posible con DNS tener dos máquinas con el mismo nombre pero en dominios diferentes (terman.ing.ula.ve y terman.faces.ula.ve).

Hemos visto qué es un dominio pero no hemos dicho nada acerca de los "dominios superiores*" (top domains*).

En un principio, la ARPANET era pequeña y solamente se crearon algunos dominios básicos (lo cual no es cierto hoy en día, ya que de 230 países por lo menos 180 tienen su propio dominio, todo esto aparte de las nuevas proposiciones).

Los dominios en Internet están divididos de dos maneras:

### ❖ Dominios genéricos:

Son conocidos como dominios internacionales u organizacionales. Son organizados conceptualmente y son los dominios básicos de Internet.

- Los dominios iniciales creados por el *InterNIC* son los siguientes:
- Com (Organizaciones comerciales como *sun.com*, *playboy.com*)
- Edu (Universidades, Instituciones Educativas como *Stanford.edu*)
- Gov (Agencias gubernamentales como la NASA en *nasa.gov*)
- Mil (Organismos militares *navy.mil*)
- Net (Organizaciones relacionadas con la red como *nsf.net)*
- Org (Organizaciones no comerciales como *underground.org*)

Sin embargo recientemente se ha propuesto la creción de los siguientes dominios (Los cuales vuelven obsoletos a .com, .net y .org)

- Firm (Comercios o firmas comerciales)
- Store (Negocios que ofrecen bienes para comprar)
- Web (Entidades que enfatizan actividades en el web)
- Arts (Entidades que enfatizan actividades recreacionales y de entretenimiento)
- Info (Entidades que proveen servicios de información)
- Nom (Aquellos que desean una nomenclatura personal o individual)

- Estos dominios estarán activos posiblemente al final de 1997.

### ❖ Dominios geográficos:

Son conocidos como geográficos o *ISO3166*. Son mantenidos por cada país y territorio en el mundo. Estos dominios son organizados por localidad y son útiles para organizaciones y negocios que desean operar en otros países o que quieren proteger su compañía. Si desea saber más puede consultar el *RFC920* (*Request For Comments* 920) el cual será obsoleto dentro de poco.

### ¿Cómo puedo hacer para registrar un dominio en Internet? :

Puede enviar un correo electrónico a *hostmaster@internic.net* solicitando una forma de registro para registrarse con la *InterNIC*.

Puede visitar la página de *Internet International AD HOC COMMITTE* (Los cuales trabajan para satisfacer los requerimientos de ampliación para *DNS)* en *http://www.iahc.org*.

O finalmente puede visitar *Netnames USA Domain Name Registry* (Para registrar los nuevos dominios y nuevos dominios geográficos) en *http://www.netnamesusa.com*

También puede consultar los siguientes documentos para entender un poco más sobre la organización de dominios:

- RFC1034: *DOMAIN NAMES - CONCEPTS AND FACILITIES*
- RFC 1101: *DNS Encoding of Network Names and Other Types*

## Delegación de Dominios:

Una de las metas de DNS es la descentralización de la administración. Esto se logra utilizando delegación de dominios. Delegación significa repartir tareas y responsabilidades de un *subdominio* a un grupo de personas u organización distinta a la que controla el dominio principal.

Lo que ocurre en realidad es la asignación de autoridad de sus subdominios a diferentes servidores de nombres (Estas son las máquinas que corren una implementación de DNS, por lo general *BIND)*. En sus archivos de datos, en vez de contener información sobre un subdominio, se incluye un puntero a servidores de nombres que son autoridad para ese subdominio. De esa manera, si el servidor de nombres de un dominio superior (como ula.ve) es interrogado acerca de una máquina perteneciente a un subdominio (digamos terman.ing.ula.ve) devolverá la dirección (o direcciones) de servidores que pueden darle esa información (mozart.ing.ula.ve, por ejemplo) en vez de tratar de responder el mismo.

Veámoslo con un ejemplo: Los administradores del dominio *ula.ve* quieren delegar la responsabilidad del dominio *ing.ula.ve* a los administradores de la zona. Entonces, los administradores de *ing.ula.ve* pueden cambiar datos dentro de su dominio sin afectar la configuración del dominio *ula.ve* (Cambios de datos implica agregar nuevas máquinas o nuevos subdominios al dominio actual, inclusive se pueden volver a delegar dominios). Otro ejemplo: fíjese lo que ocurre con el dominio *ve (*Dominio geográfico correspondiente a Venezuela*)*. Este dominio es administrado por *REACCIUN* quien delega a instituciones como la ULA la administración del dominio ula.ve .

## Servidores de Nombres (*Name Servers*):

Las máquinas que guardan información acerca de un espacio de dominio son llamadas servidores de nombres.

Los servidores, generalmente tienen información completa acerca de una parte del espacio de dominio de nombres, llamado *zona*. El servidor de nombres entonces se dice que tiene autoridad para esa zona (Los servidores de nombres pueden ser autoridades para más de una zona).

La diferencia entre una *zona* y un *dominio* es sutil. Una zona contiene los nombres de dominio y datos que representan a un dominio. Un dominio es un nombre que agrupa a otras máquinas o dominios inferiores.

Las especificaciones de DNS definen 2 tipos de servidores: *maestros primarios y maestros secundarios*. Un servidor primario maestro obtiene los datos de las zonas sobre la cual tiene autoridad desde archivos que están en la misma máquina que el servidor de nombres, mientras que un maestro secundario realiza una transferencia de zona (y sigue siendo autoridad para la zona).

## Registro de Recursos (RR):

Los datos asociados con nombres de dominios deben ser guardados de alguna manera en los servidores de nombres. La forma utilizada por Bind es guardarlos en archivos de texto simple con etiquetas llamadas registros de recursos (*Resource Records* o RR). Cada clase de registro pertenece a un tipo de red o software. Actualmente hay clases para Internet (y cualquier red basada en *TCP/IP*), redes basadas en el Protocolo *Caosnet* (Red vieja, de importancia histórica) y redes que usan el protocolo *hesiod*. En esta práctica (y seguramente en la vida real) solamente trabajaremos con registros tipo *IN (Internet).*

Aparte de estos registros existen otros registros que guardan información útil sobre las máquinas pertenecientes a un dominio (De ellas hablaremos más en la parte práctica). Algunos de éstos:

- 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 maestro* guarda su información en archivos de bases de datos (*database files*).

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 (para la red *Loopback* y las pistas de los *Root Name Servers*) y toman sus datos del servidor primario (Es decir hacen una *transferencia de zona*).

## Resolvers:

Es un conjunto de bibliotecas de las aplicaciones clientes (Es decir aquellas que solicitan información acerca de un espacio de dominios de nombres).

Un *resolver* tiene como tareas:

- Interrogar al servidor de nombres
- Interpretar respuestas (Que pueden ser registros RR o errores)
- Devolver información al programa que la solicita.

## ¿Cómo trabaja?:

Los servidores de nombres tienden a buscar datos de un espacio de dominio de nombre. Tienen que comportarse de esa manera, dada la poca inteligencia del *resolver*. No sólo pueden dar datos acerca de zonas de la que tienen autoridad sino que pueden buscar a lo largo de un espacio de dominio para encontrar datos sobre los que no tienen autoridad (A esto se le conoce como *resolución*).

La resolución comienza siempre desde los servidores de dominios superiores hasta llegar al servidor que tiene la información autorizada de un dominio en particular (Es decir comienza desde la parte superior del árbol invertido hasta llegar a la rama buscada).

En el proceso de resolución de nombres podemos identificar algunos elementos:

### Servidores de 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.

Como puede haberse dado cuenta 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:

Primero 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, entonces el servidor de nombres busca en su cache y si no está allí entonces comienza el proceso de resolución que se puede comportar de estas dos formas:

- **Recursiva**

Un servidor de nombres envía una respuesta recursiva *cuando es el servidor y no el cliente el que pregunta a otros servidores de nombres por la información del dominio solicitada*. Esto ocurre cuando el servidor de nombres sabe que el resolver no tiene la inteligencia de manejar una referencia a otro servidor de nombres (Es decir, el resolver hace explícitamente una pregunta recursiva). 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 (de 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 entonces devuelve la respuesta (servidor) más cercano al dominio buscado. Si el servidor falla, no lo vuelve a reintentar.

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

## Mapeo de direcciones a nombres:

Hemos visto como convertir nombres a direcciones, pero ahora queremos saber como se convierte una dirección IP  a nombres. El mapeo de direcciones a nombres es útil para salidas que sean fáciles de leer por los seres humanos,  fáciles de interpretar en bitacoras 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 tablas de máquinas (hosts tables) la conversión es fácil. Requiere de una búsqueda secuencial a lo largo de la tabla de máquinas usando una dirección. En DNS, sin embargo, el espacio de dominio de nombres esta indizado 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 especifico 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 procesando una búsqueda recursiva 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 acortado un poco, ya que el servidor primero revisará en su caché 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).

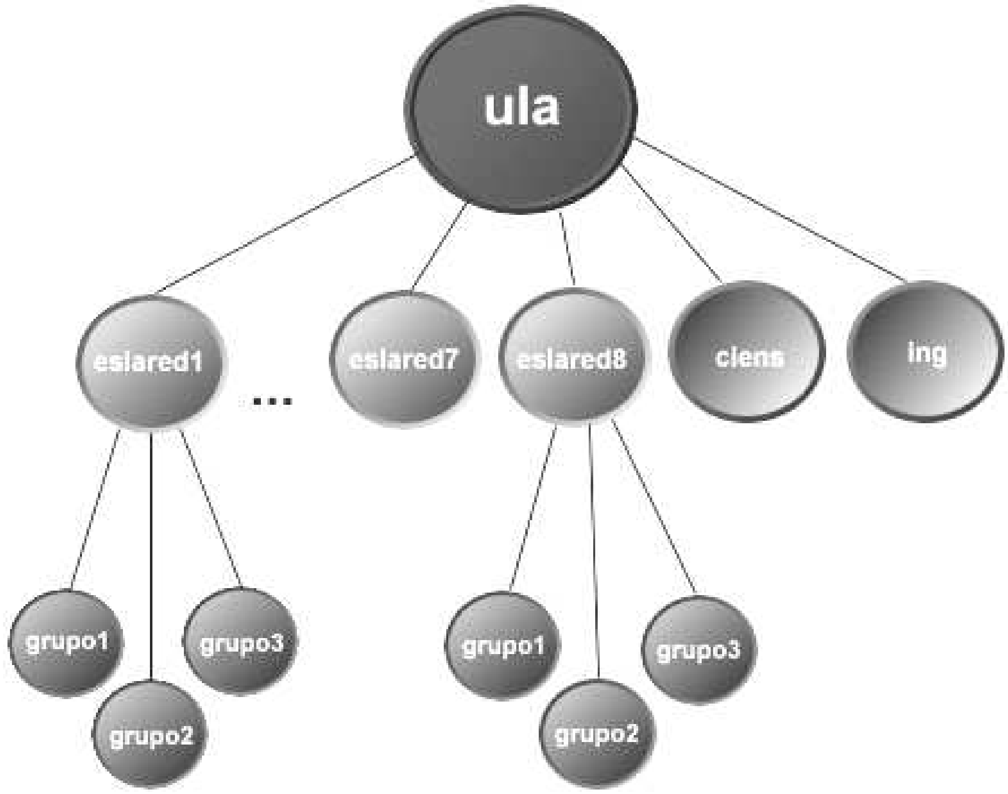
## Tiempo de vida (Time To Live, TTL):

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 (De ser así, las modificaciones hechas en un servidor maestro nunca se propagarían por la red). De esta manera, 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 transfiriendo una zona. Una vez que el tiempo de vida expira, el servidor busca de nuevo los datos del dominio del cual es servidor secundario.

## DESARROLLO PRÁCTICO

La siguiente figura muestra un conjunto de dominios, parte del dominio principal **ula**.




Los dominios *ing.ula* y *ciens.ula* ya fueron creados por los administradores de RedULA, pero se necesita que usted y sus compañeros construyan varios dominios adicionales dentro del dominio *ula.ve* llamados eslared1, eslared2, eslared3, y así sucesivamente (el instructor le indicará cómo):
Cada uno de esos dominios a su vez contendrán a los dominios *grupo1,grupo2 y grupo3.*
En esta práctica su máquina será tanto cliente como servidor de un dominio dado.

## Configuración de los clientes:

Configure a su máquina para que sea cliente de sí misma y de las demás máquinas dentro del laboratorio. (Solamente coloque unos cuantos servidores, *no pierda mucho tiempo con esta parte, recuerde que debe configurar también a su servidor).*
Por ejemplo mega.eslared1.ula.ve es cliente de si misma y de mega.eslared2.ula.ve, y mega.grupo1.eslared1.ula.ve es cliente de si misma y es cliente de mega.grupo2.eslared1.ula.ve.

## Configuración de los servidores:

Se desea que cada uno de estos dominios y subdominios tengan referencias a:
Un intercambiador de correos (*Mail Exchanger*) llamado *mailserver* (DNS juega un papel vital en el envío y recepción de correos usando los registros *MX*. Se darán más detalles en la práctica).

- Un enrutador llamado *router*
- Un servidor *NIS* llamado *nisserver*
- Un servidor *DNS* llamado *dnsserver*

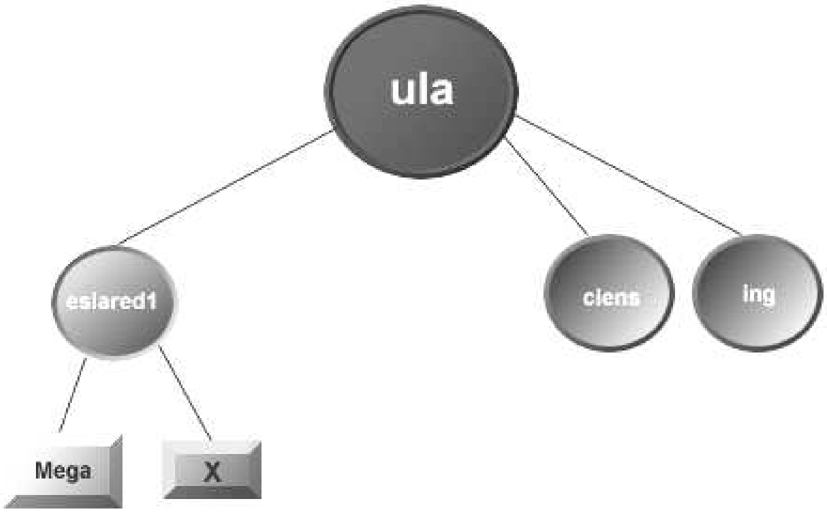
Para tener redundancia (medida de protección a prueba de fallos), se exige que cada servidor primario sea a su vez servidor secundario de otros dominios (Por ejemplo, grupo1 debe ser servidor secundario de grupo2 y grupo3).

Pruebe agregando máquinas a los dominios que la haya asignado el instructor (por ejemplo, una máquina llamada ***angel*** dentro del dominio *grupo1.eslared1.ula.ve* tendría el nombre completamente calificado de ***angel****.grupo1.eslared1.ula.ve*).

También tendrá que hacer delegación de dominios para los administradores de los subdominios grupo1, grupo2 y grupo3 (Puede utilizar direcciones IP ficticias).

## ¿Cómo configurar BIND en una máquina con Linux?

Estos archivos se proveen como guía para la parte práctica propuesta, y en ellos se muestra la configuración del servidor de nombres del dominio *eslared1.ula.ve*.




Supongamos que tenemos una máquina dentro del dominio *eslared1.ula.ve* y queremos que esta use como servidor de nombres a una máquina llamada *mega.eslared.ula.ve* (Supongamos que la máquina se llama "*X*").

### Archivos de configuración básicos en el cliente:

La configuración en el cliente depende bastante del sistema operativo. Hoy en día es común tener estaciones de trabajo Unix, PC's con Windows y Linux (Entre otras plataformas).
Linux (Y otras variantes de UNIX) utilizan las bibliotecas del resolver en el lado del cliente.

Pasos de configuración bajo Linux:

1. Verifique que en el */etc/hosts* solamente aparezcan líneas similares a esta:

```
127.0.0.1   loopback
150.185.176.1  mega.eslared1.ula.ve mega
```

2. Edite /etc/resolv.conf

```
# Defino el dominio por omisión el cual se agregará a nombres que
no
# terminen en punto
domain eslared1.ula.ve
# Defino otros dominios de búsqueda
search eslared1.ula.ve ing.ula.ve ciens.ula.ve faces.ula.ve
# Defino servidores de nombres
# Busco en mega1.eslared1.ula.ve
nameserver 150.185.176.1
# Luego en mozart.ing.ula.ve
nameserver 150.185.146.1
```

```
# Luego en arha.ciens.ula.ve
nameserver 150.185.128.2
```

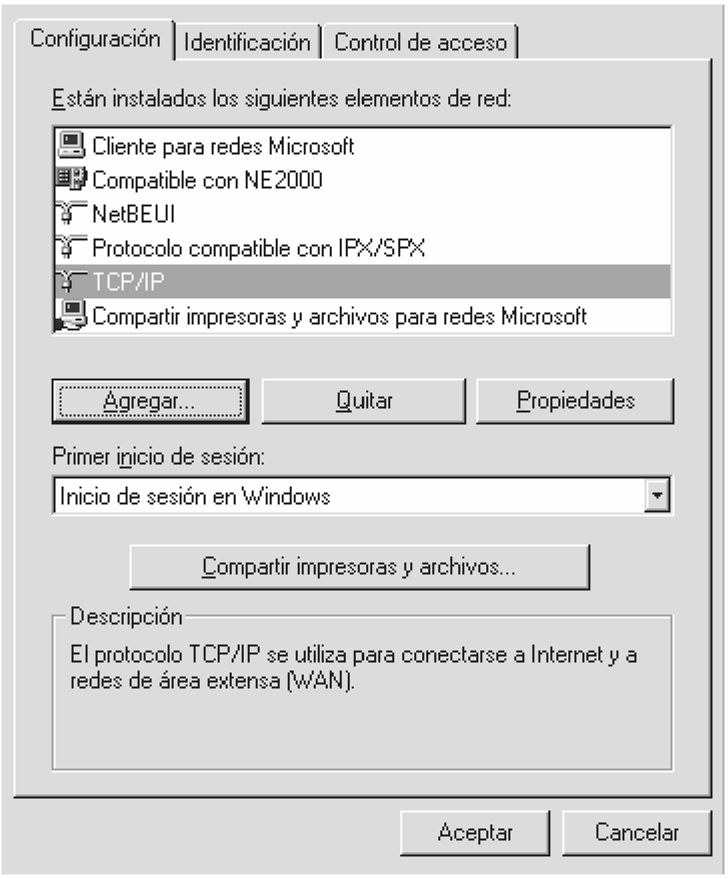
3. Edite el archivo */etc/host.conf* (se define allí el orden de búsqueda de información)

```
# Primero busque en el /etc/hosts y si falla use a Bind
order hosts, bind
multi on
```

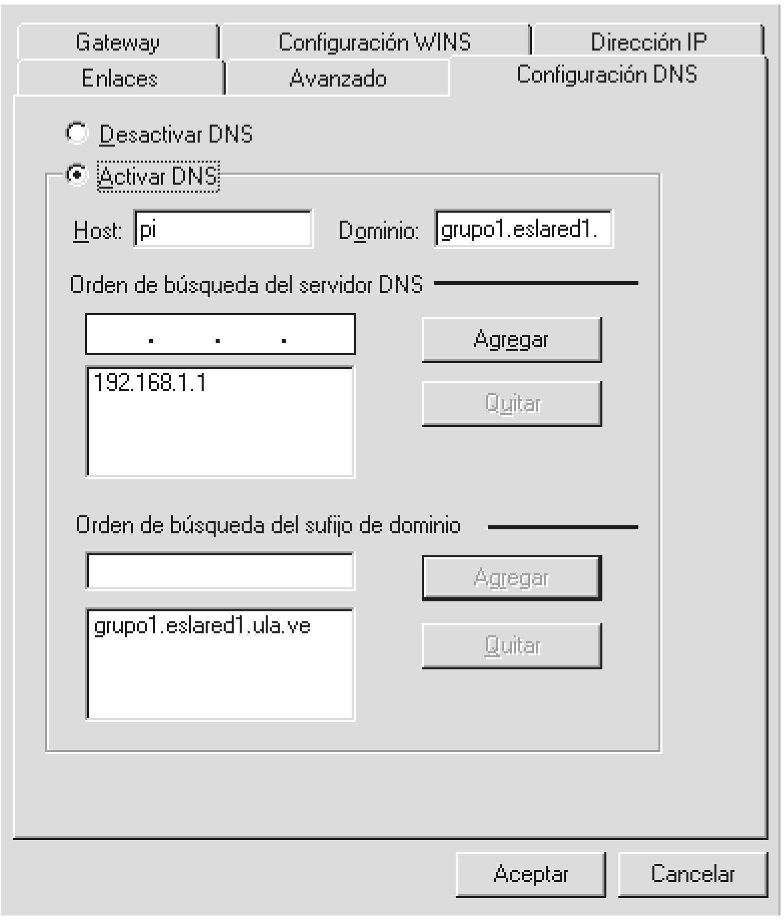
**OPCIONAL: Configuración del cliente bajo Windows95.**

1. Vaya al panel de control y haga doble click en el icono que dice redes


2. Presione seleccione el protocolo TCP/IP y presione el botón que dice propiedades:




3. Llene las casillas de manera apropiada y al final presione OK.




4. Cuando se le pregunte si desea reiniciar a Windows responda que si.

## Configuración del servidor:

Los siguientes archivos de configuración le mostrarán como puede implementar el dominio. Estos archivos pueden ser utilizados como base para los otros dominios propuestos en la práctica.

Los servidores de nombres de estos dos dominios son servidores primarios y secundarios a la vez, y se supone que están conectados a Internet (usando un enrutador).

Explicaremos la configuración de los servidores suponiendo que hablamos de una máquina llamada *mega.eslared1.ula.ve*

Bind puede ser configurado para correr de maneras diferentes. Las configuraciones más comunes para correr Bind son:

- **Solamente *resolver***: El código del resolver se encarga de preguntar a los servidores de nombre por información de dominio. La parte de configuración de resolver ya fue explicada con anterioridad por lo que no entraremos en detalle.

- **Solo cache**: Un servidor de sólo cache corre el software del servidor, pero no tiene archivos de base de datos del servidor. Aprende las respuestas de otros servidores de nombres, la guarda y la 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.

- **Primario**: El servidor de nombres primario es la fuente autoritaria de toda la información referente a un dominio. Carga la información de archivos mantenidos localmente por el administrador del dominio. Este archivo (el archivo de zona) contiene la información más precisa acerca de una pieza de la jerarquía del dominio sobre la cual el servidor tiene autoridad. Un servidor primario requiere un conjunto completo de archivos: archivos de zonas para el dominio regular (*db.eslared1.ula.ve*) y el dominio reverso (*db.eslared1.ula.ve.rev*), el archivo de inicialización (*named.boot*), el archivo de pistas que contiene a los RNS (*db.root*) y el archivo *loopback* (db.127.0.0). Ninguna otra configuración requiere esta cantidad de archivos.

- **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 del los archivos del servidor primario (Lo cual lo hace autoridad).

### Pasos para configurar el servidor (Se proveen archivos solo como ejemplo):

1. Edite el archivo /etc/named.boot (sino existe, créelo)

```
;   Hecho por José Vicente Nuñez Zuleta
;   Email: jose@ing.ula.ve
;   Web http://www.ing.ula.ve/~jose
;   ESLARED 97
;
;   En este archivo se especifica que tipo de servidor y
;   la ubicación de los archivos de datos. named.boot
;   contiene una línea por cada mapa que debe ser leído.
;
;   En este caso, somos servidor primario del dominio
;   eslared1.ula.ve y secundarios de los demás subdominios
;   de ula.ve
;
;   Este archivo puede estar en el directorio /etc o puede estar
;   en otro directorio (por ejemplo invocando a
;   /usr/sbin/named -b /directorio_cualquiera/named.boot)
;
;   ** Comienzo de la configuración **
;
;   Directorio en donde van mis archivos de datos (mapas)
directory /etc/named
```

```
;
;   Soy servidor primario para los siguientes dominios:
;tipo de servidor  dominio donde es la autoridad     archivo donde
guarda la inf.
primary    eslared1.ula.ve           db.eslared1.ula.ve
primary    0.0.127.in-addr.arpa          db.127.0.0
primary    176.185.150.in-addr.arpa      db.150.185.176
;
;   Y servidor secundario de estos dominios
; (Estos mapas son traídos de manera automática de los servidores
; autoritarios de estos dominios)
;tipo de servidor  dominio    servidor   archivo
secondary  ula.ve     150.185.130.1  db.ula.ve
secondary  130.185.150.in-addr.arpa  150.185.130.1
    db.150.185.130
secondary  eslared2.ula.ve       db.eslared2.ula.ve
secondary  177.185.150.in-addr-arpa  150.185.177.1
    db.150.185.177
;
;   Archivo de cache (se especifican los "root servers", no
contiene datos de cache)
;tipo de serv  dominio sobre el que autoritario  archivo
cache      .      db.root
```

Cree un directorio llamado */etc/named* y edite allí el archivo *db.dominio* (Donde dominio es el nombre del dominio del cual será servidor primario).

```
; db.eslared1.ula.ve
;
; Hecho Por:   José Vicente Nuñez Zuleta
; Email:   jose@ing.ula.ve
; WWW:     http://www.ing.ula.ve/~jose
; ESLARED 97
;
;
; Este archivo contiene la mayoría de la información del dominio
; Direcciones e información de máquinas
;
; En los RFC el orden en que son llenados los campos es el
siguiente:
;
;   SOA Indica autoridad para los datos de este dominio
;   IN  Clase Internet (Por defecto en el mundo real)
;   NS  Lista a un servidor de Nombres para este dominio
;   A   Mapeo de direcciones a nombres
;   CNAME  Nombre canónico (Para hacer alias)
;   HINFO  Información de la máquina (RFC1340, Campo Opcional)
;   MX  Intercambiadores de correo
;
; Sin embargo el orden no es obligatorio
;
; Aparte de eso, los campos de información tienen limitaciones
; en cuanto a tamaño:
;
; etiquetas:       63 octetos o menos
; nombres:     255 octetos o menos
; TTL,Time To Live valores positivos de un numero entero de 32
bits con signo
;          en segundos.
; Para mayor información acerca de este y otros registros consulte
el
; RFC1035, Domain Names: Implementation and Specification
;
; NOTA: Si utiliza un nombre completamente calificado asegure de
terminarlo
; con un punto ("."). Por ejemplo:
;   angel.eslared1.ula.ve.    BIEN
;   angel.eslared1.ula.ve  MAL (DNS le pegara .eslared1.ula.ve. y
el nombre
;   quedara como angel.eslared1.ula.ve.eslared1.ula.ve.)
;
; La primera entrada es el SOA, Start Of Autority, comienzo de
autoridad
; Indica que este servidor de nombres es la mejor fuente de
información
; para los datos dentro de este dominio
```

```
; Cada archivo db.dominio y db.dominio.rev REQUIERE un registro
SOA
;
; El formato de la primera línea es:
; dominio IN SOA host responsable
; - @ me agrega eslared1.ula.ve como mi dominio (Lo definí en
named.boot)
; - @ es una abreviatura (pude haber colocado eslared1.ula.ve)
; - host es la máquina en donde se crearon los datos
; - responsable es la dirección de correo del responsable
;   (Se especifica la máquina en donde esta la cuenta y el
carácter @ se
;   reemplaza con un punto). Es decir:
;   root.mega.eslared1.ula.ve es en realidad
root@mega.eslared1.ula.ve

@   IN  SOA mega.eslared1.ula.ve.  root.mega.eslared1.ula.ve. (
        274971 ;   Serial de Modificación (dd/mm/aa/Numero de
Modificaciones)
        10800  ;   Actualice los datos cada 3 horas (En segundos)
        3600   ;   Si no pudo obtener los datos, reintente cada
hora
        604800 ;   Caduca la validez de los datos después de una
semana
            ;   Significa, que si hemos fallado en contactar al
servidor
            ;   "expire" segundos es mejor no utilizar los datos
        86400  ;   Tiempo de vida por defecto de los datos (1 día)
        )
; Los tiempos anteriores le dicen a los servidores secundarios
cada cuanto buscar
; por datos nuevos (Es decir, hacer una transferencia de zona)
; NOTA: Si los valores anteriores de tiempo son muy pequeños, su
servidor puede verse
; muy saturado. Trate de tener un balance entre exactitud de la
información y carga
;
    IN  NS  mega.eslared1.ula.ve.  ; Servidor de nombres para
eslared1.ula.ve
    IN  NS  giga.eslared1.ula.ve.  ; Servidor de nombres para
eslared1.ula.ve
                        ; SINTAXIS: dominio IN NS
máquina.mi.dominio
    IN  MX  5   mega.eslared1.ula.ve.  ; Intercambiador de correo
    IN  MX  10  giga.eslared1.ula.ve.  ; SINTAXIS:
(dominio|máquina) IN MX prioridad máquina.mi.dominio
                        ; Entre mas baja la prioridad, sera mas
preferido
                        ; primero se prueba a "mega" y luego a
"giga" al
                        ; enviar correo al dominio eslared1.ula.ve
; Mapeo de nombres a direcciones
localhost   IN  A   127.0.0.1  ; Defino a localhost
mega        IN  A   150.185.176.1  ; Mapeo de dirección Nombre a
dir IP
giga        IN  A   150.185.176.2  ; SINTAXIS: máquina IN A dirIP
alfa        IN  A   150.185.176.3
beta        IN  A   150.185.176.4
gamma       IN  A   150.185.176.254
;
; Alias
; SINTAXIS: alias IN CNAME nombre.mi.dominio.
www     IN  CNAME   giga    ; Servidor WWW
loghost     IN  CNAME   mega    ; Donde enviar los logs
mailhost    IN  CNAME   mega    ; Donde enviar correos
nishost     IN  CNAME   giga    ; Servidor de NIS
routerhost  IN  CNAME   gamma   ; Enrutador

; Descripcion de los equipos
www IN  TXT "Servidor World Wide Web"
nishost IN  TXT "Servidor de NIS"
mega    IN  TXT "Mega servidor"
giga    IN  HINFO   "Pentium Intel" "Linux"
beta    IN  HINFO   "Pentium Intel" "FreeBSD"

; Delegación de dominios con "GLUE RECORDS"
```

```
; Creamos varios dominios y delegamos su autoridad a otras
personas
; Para delegar el dominio, necesitamos utilizar "glue records", es
decir
; 1) Declaramos el nuevo dominio y sus servidores de nombres
; 2) Hacemos el mapeo de Nombres a direcciones IP
;
; Una vez hecha la delegación, solamente se agregan aquí los
; servidores de nombres
;
; En nuestro caso, creamos los dominios grupo1,grupo2 y grupo3
; con un solo servidor de nombres (por ahora)
grupo1  IN NS pi.grupo1.eslared1.ula.ve.
grupo2  IN NS mu.grupo2.eslared1.ula.ve.
grupo3  IN NS epsilon.grupo3.eslared1.ula.ve.
;
; Ahora especificamos sus direcciones IP
pi.grupo1.eslared1.ula.ve. 86400 IN A 150.185.176.9
mu.grupo2.eslared1.ula.ve. 86400 IN A 150.185.176.16
epsilon.grupo3.eslared1.ula.ve. 86400 IN A 150.185.176.18

;Fin del archivo
```

Edite el archivo */etc/named/db.150.185.176* (Donde 150.185.176 es el nombre del dominio reverso definido en /etc/named.boot)

```
; db.150.185.176
;
; Hecho por:   José Vicente Nuñez Zuleta
; Email:   jose@ing.ula.ve
; Web:     http://www.ing.ula.ve/~jose
;
; ESLARED 97
;
; Este archivo se utiliza para hacer el mapeo de dirección a
nombres
;
; Debe haber un archivo de este tipo por cada red la cual
pertenezca
; al dominio eslared1.ula.ve
; por ejemplo, si el dominio eslared1.ula.ve abarcara a las
subredes:
; - 150.185.176.0
; - 150.185.177.0
; Deberían existir los archivos:
; - 176.185.150.in-addr.arpa
; - 177.185.150.in-addr.arpa
;
; @ me agrega como dominio 176.185.150.in-addr.arpa. (definido en
named.boot)
@   IN  SOA mega.eslared1.ula.ve.  root.eslared1.ula.ve. (
    214971     ;Serial (dd/mm/aa/ # De modificación)
    10800      ;Refrescar, 3 horas
    3600       ;Reintentar,1 hora
    604800     ;Expirar, 1 semana
    86400      ;Mínimo, 1 día
    )
;
; Defino el dominio reverso para mega
; como el dominio por defecto es 176.185.150.in-addr.arpa colocar
"1" es
; como colocar 1.176.185.150.in-addr.arpa.
;
    IN  NS  mega.eslared1.ula.ve.
    IN  NS  giga.eslared1.ula.ve.
1   IN  PTR mega.eslared1.ula.ve.
2   IN  PTR giga.eslared1.ula.ve.
;
; Defino el dominio reverso para las máquinas de este dominio
;
3   IN  PTR alfa.eslared1.ula.ve.
4   IN  PTR beta.eslared1.ula.ve.
254 IN  PTR gamma.eslared1.ula.ve.
9   IN  PTR pi.grupo1.eslared1.ula.ve.
16  IN  PTR mu.grupo2.eslared1.ula.ve.
18  IN  PTR epsilon.grupo3.eslared1.ula.ve.
```

Edite el archivo /etc/named/db.127.0.0

```
; db.127.0.0
;
; Hecho por:   José Vicente Nuñez Zuleta
; Email:   jose@ing.ula.ve
; WWW:     http://www.ing.ula.ve/~jose
;
; Este archivo se usa para convertir las direcciones 127.0.0.1
(loopback)
; en la direccion localhost
;
; Pregunta: ¿Porque utilizar un archivo tan bobo? ;-)
; Respuesta: Aunque el servidor DNS puede funcionar sin este
archivo
; es bueno colocarlo para evitar sorpresas. Debido a que nadie
tiene
; la responsabilidad de mapear la red 127.0.0.1 a un nombre,
entonces
; cada servidor debe asegurarse por si mismo de hacer este mapeo.
; Si no esta definido, una búsqueda a 127.0.0.1 fallaría
;
; @ me agrega 0.0.127.in-addr.arpa como dominio (definido en
named.boot)
; esto ocurre solo si el nombre o dominio no termina en .
@   IN  SOA mega.eslared1.ula.ve   root.eslared1.ula.ve. (
        214971      ;Serial (Indica cambios hechos a la conf.)
        36000       ;Refresque cada 100 horas
        3600        ;Reintente después de una hora
        3600000     ;Expire después de 1000 horas
        2419200     ;Tiempo de vida (Un mes)
        )
    IN  NS  mega.eslared1.ula.ve.
1   IN  PTR localhost. ; Nótese al punto al final, no queremos que
expanda el dominio
```

Finalmente edite el archivo /etc/named/db.root

```
; db.root
;
; Hecho por:   José Vicente Nuñez Zuleta
; Email:   jose@ing.ula.ve
; WWW:     http://www.ing.ula.ve
;
;
; Archivo de inicialización para mega
;
; Además de la información local, el servidor de Nombres
; necesita saber donde están los dominios raíz
;
; Esta información debe ser actualizada cada cierto tiempo
; El dominio "." se refiere al dominio raíz
;
.   99999999   IN  NS  A.ROOT-SERVERS.NET.
.   99999999   IN  NS  B.ROOT-SERVERS.NET.
.   99999999   IN  NS  C.ROOT-SERVERS.NET.
.   99999999   IN  NS  D.ROOT-SERVERS.NET.
.   99999999   IN  NS  E.ROOT-SERVERS.NET.
.   99999999   IN  NS  F.ROOT-SERVERS.NET.
.   99999999   IN  NS  G.ROOT-SERVERS.NET.
.   99999999   IN  NS  H.ROOT-SERVERS.NET.
.   99999999   IN  NS  I.ROOT-SERVERS.NET.
A.ROOT-SERVERS.NET.    99999999   IN  A   198.41.0.4
B.ROOT-SERVERS.NET.    99999999   IN  A   198.9.0.107
C.ROOT-SERVERS.NET.    99999999   IN  A   192.33.4.12
D.ROOT-SERVERS.NET.    99999999   IN  A   128.8.10.90
E.ROOT-SERVERS.NET.    99999999   IN  A   192.203.230.10
F.ROOT-SERVERS.NET.    99999999   IN  A   192.5.5.241
G.ROOT-SERVERS.NET.    99999999   IN  A   192.112.36.4
H.ROOT-SERVERS.NET.    99999999   IN  A   128.63.2.53
I.ROOT-SERVERS.NET.    99999999   IN  A   192.36.148.17

; Si no estuviera conectados a Internet, entonces el servidor
; del dominio root soy yo mismo
;
```

```
;   Servidores para el dominio root
;.          99999999   IN  NS  mega.eslared1.ula.ve.
;   Servidores para el dominio root por numero
;mega.eslared1.ula.ve.    99999999   IN  A   192.168.1.1
```

## Depuración de errores:

Utilizaremos algunos procedimientos y herramientas prácticas para depurar errores en la configuración de nuestros servidores.
Una vez terminada la preparación de los archivos de configuración, es hora de arrancar el servidor (Y de ver cualquier mensaje de error en /var/adm/messages).

Escriba:

***/usr/sbin/named <ENTER>***

Sí su configuración tiene errores, BIND se lo dirá en */var/adm/messages*: (Para abrirlo escriba "***cat /var/adm/messages***")

```
Jul  3 07:26:27 mega named[435]: starting.  named 4.9.3-BETA26 Sun Nov 26
20:58:49 CST 1995 ^Iroot@fuzzy:/tmp/bind-4.9.3-BETA26/named
Jul  3 07:26:27 mega named[435]: cache zone "" loaded (serial 0)
Jul  3 07:26:27 mega named[435]: primary zone "eslared1.ula.ve" loaded
(serial 9705122)
Jul  3 07:26:27 mega named[435]: primary zone "176.185.150.in-addr.arpa"
loaded (serial 970512)
Jul  3 07:26:27 mega named[435]: primary zone "0.0.127.in-addr.arpa" loaded
(serial 970512)
Jul  3 07:26:27 mega named[435]: secondary zone "ula.ve" loaded (serial
970631)
Jul  3 07:26:27 mega named[436]: Ready to answer queries.
```

Hay una enorme cantidad de mensajes que syslog puede emitir (Sin embargo, en la práctica usted verá solamente algunos de ellos).

Por ejemplo, cada vez que usted reinicie named verá una salida como esta:

```
mega:/var/adm# ps -aux|grep named     (Busque Identificador del proceso en
memoria)
root       436  0.0  3.4   952   504  ?  S    07:26   0:00 /usr/sbin/named
root       482  0.0  1.1   812   168  p0 R    08:07   0:00 grep named
mega:/var/adm# kill -HUP 436      #Reinicie el servidor
mega:/var/adm# tail /var/adm/messages #Vemos el contenido del archivo
mega named[436]: reloading nameserver
Jul  3 08:09:26 mega named[436]: Ready to answer queries.
```

Este mensaje no es de error, solamente nos dice que el servidor está listo para responder preguntas (De esta manera usted puede cambiar la configuración y puede obligar a que el servidor vuelva a leerla).

Un mensaje muy común de error es el siguiente (de nuevo, revise /var/adm/messages):

Jul 3 08:09:26 mega named[436]:mega has CNAME and other data (illegal)

Este mensaje significa que hay problemas con los datos de la zona

Otra forma de detectar errores es corriendo named en modo de depuración (*debug mode*). Esto puede hacer de dos maneras:

mega #: ***/usr/sbin/named -d 4 &*** (Corre a named con un nivel de depuración 4)

Si el demonio está activo puede aumentar el nivel de depuración con la señal USR1 (cada invocación de named con esta señal incrementa el nivel de depuración en 1) y puede detener la depuración con USR2. Por ejemplo:

```
mega#  kill -SIGUSR1 `cat /var/run/named.pid `  (Nivel 1)
mega#  kill -SIGUSR1 `cat /var/run/named.pid `  (Nivel 2)
```

```
mega#  kill -SIGUSR1 `cat /var/run/named.pid `  (Nivel 3)
mega#  kill -SIGUSR2 `cat /var/run/named.pid `  (Sin
depuración)
```

La siguiente seria una salida de ejemplo de la modalidad de depuración anterior (Puede ver este archivo con ***vi /var/tmp/named.run***)

```
Debug turned ON, Level 4
Version = named 4.9.3-BETA24 Aug-4-1995 kkeyte@esoc.esa.de
bootfile = /etc/named.boot
fclose(4) succeeded
named[7913]: starting.  named 4.9.3-BETA24 Aug-4-1995
kkeyte@esoc.esa.de
considering [127.0.0.1]
dqp->dq_addr 127.0.0.1 d_dfd 5
listening [127.0.0.1]
loopback address: x100007f
considering [192.168.1.1]
dqp->dq_addr 192.168.1.1 d_dfd 6
listening [192.168.1.1]
dqp->dq_addr 0.0.0.0 d_dfd 7

ns_init(/etc/named.boot)
savehash GROWING to 2
savehash GROWING to 2

 content of zones before loading
zone origin eslared1.ula.ve, new zone, source =
named.eslared1.ve
purge_zone(eslared1.ula.ve,1)
reloading zone
db_load(named.eslared1.ve, eslared1.ve, 1, Nil)
```

También puede vaciar el contenido de una zona o bien puede obtener estadísticas de funcionamiento de su servidor con:

```
mega# kill -SIGINT `cat /var/run/named.pid ` (Hace u n vaciado de zona en
/var/tmp/named_dump.db)
mega # kill -SIGIOT `cat /var/run/named.pid ` (Hace un
vaciado de las estadisticas de funcionamiento del servidor en
/var/tmp/named.stats).
```

Para mayor información escriba ***man named***.

Otro sitio interesante para buscar mensajes de error (Y de funcionamiento) es en */var/adm/debug*:
(Aquí se ve una transferencia de zona exitosa)

```
Jul  2 21:19:19 mega named[52]: zone transfer of
"eslared1.ula.ve" to [150.185.130.1].3514 (pid 172)
Jul  3 00:08:45 mega named[52]: zone transfer of
"eslared1.ula.ve" to [150.185.128.1].1670 (pid 241)
```

Para revisar este y otros problemas nos valemos de herramientas como nslookup:

```
mega:/var/adm# /usr/bin/nslookup
Default Server:  mega.eslared1.ula.ve
Address:  150.185.176.1

>
```

Significa que nslookup está en espera y que esta utilizando a mega como servidor por omisión.
Por ejemplo, probemos la existencia de una máquina llamada alpha:

```
>alpha.eslared1.ula.ve
Server:  mega.eslared1.ula.ve
Address:  150.185.176.1

Name:    alpha.eslared1.ula.ve
Address:  150.185.176.1

>
```

¡La máquina existe! (y nslookup me devuelve la dirección IP)

En cambio busquemos a angel.eslared1.ula.ve

```
> angel.eslared1.ula.ve
Server:  mega.eslared1.ula.ve
Address:  150.185.176.1

*** mega.eslared1.ula.ve can't find angel.eslared1.ula.ve:
Non-existent host/domain
>
```

La máquina no existe (esta prueba es útil cuando usted desea agregar nuevas máquinas a su dominio DNS)

Con nslookup puedo ver también, algunos registros particulares. Supongamos que quiero saber cuáles son los servidores de nombres del dominio eslared1.ula.ve:

```
> ls -t ns eslared1.ula.ve
[mega.eslared1.ula.ve]
 eslared1.ula.ve.                 server =
mega.eslared1.ula.ve
 eslared1.ula.ve.                 server =
giga.eslared1.ula.ve
>
```

Vemos que hay dos servidores de nombres para ese dominio.

Nslookup permite explorar otros dominios también. Por ejemplo, queremos ver que máquinas tiene el dominio ing.ula.ve:

```
> server mozart.ing.ula.ve
Default Server:  mozart.ing.ula.ve
Address:  150.185.146.1

>ls -t a ing.ula.ve
silmaril                       150.185.146.222
ampere                         150.185.146.218
cajal                          150.185.146.144
ilegal                         150.185.146.86
gormet                         150.185.146.25
condensador                    150.185.146.199
gatica                         150.185.146.74
>
```

Nslookup es una herramienta muy extensa, por lo que sí quiere aprender mas acerca de sus opciones deberá consultar en el manual (man nslookup).

## Opcional: Uso de DNSWALK

Otra herramienta que es de mucha utilidad para validar el funcionamiento de un servidor DNS es dnswalk. Una forma común de invocar esta herramienta es la siguiente:

mega:~# ***dnswalk -a -l eslared1.ula.ve. <ENTER>*** (Nótese el punto al final del dominio)

(La opción -a activa advertencias acerca de registros A duplicados, la opción -l busca "delegación débil de dominios).

El programa muestra por pantalla algo así:

Getting zone transfer of eslared1.ula.ve. from mega.eslared1.ula.ve....done.
Checking eslared1.ula.ve.
SOA=mega.eslared1.ula.ve.      contact=root.eslared1.ula.ve.

Usted quedará con un directorio llamado ./ve donde habrá mas subdirectorios dependiendo de la cantidad de subdominios analizados (En nuestro caso tenemos ./ve/ula/eslared1). Allí estará un archivo llamado axfr.
Abra el archivo con su editor favorito y examine su contenido:

**vi ./ve/ula/eslared1/axfr**

También puede hacer preguntas acerca del dominio reverso:

mega:~# **dnswalk -a -l 176.185.150.in-addr.arpa.**
Getting zone transfer of 176.185.150.in-addr.arpa. from mega.eslared1.ula.ve....done.
Checking 176.185.150.in-addr.arpa.
SOA=mega.eslared1.ula.ve.      contact=root.eslared1.ula.ve.
mega:~#

Lo cual lo dejará con el archivo /arpa/in-addr/150/185/176/axfr

Pruebe introducir errores en los archivo de configuración de DNS y vuelva a cargar el demonio named para ver los mensajes de error en los archivos */var/adm/messages* y */var adm/debug*)

## Bibliografía y herramientas:

La siguiente bibliografía está relacionada con el tema y puede resultar de mucha utilidad en su aprendizaje:


- Información acerca de DNS, RFC y herramientas en http://www.dns.net/dnsrd/
- Información adicional sobre dominios en ITU (International Telecommunication Union) en http://www.itu.int/PPI/press/releases/1997/itu-08.html
- DNS Howto para Linux en http://www.linux.org
- Hunt Craig. TCP-IP, Network Administration. Capitulo 8, pag. 167.1993. O'Reilly & Asociates.
- Albitz Paul y Liu Cricket. DNS and Bind in a Nutshell. 1994. O'Reilly & Asociates.
- RFC920: Domain Requirements
- RFC1033: Domain Administrators Operations Guide (Actualizado por el RFC1912)
- RFC1034: Domain Names - Concepts And Facilities (Actualizado por el RFC1101)
- RFC1035: Domain Names - Implementation And Specification (Actualizado por el RFC1706)
- RFC1101: DNS Encoding of Network Names and Other Types
- RFC1178: Choosing A Name For Your Computer
- RFC1183: New RR Types
- RFC1536: Common DNS Implementation Errors and Suggested Fixes
- RFC1537: Common DNS Data File Configuration Errors
- RFC1591: Domain Name System Structure and Delegation
- RFC1706: DNS NSAP Resource Records
- RFC1713: Tools for DNS debugging
- RFC1912: Common DNS Operational and Configuration Errors