INSTALACION, CONFIGURACION Y PUESTA EN MARCHA DEL SOFTWARE ZEBRA DE LIBRE UTILIZACION, ACTUANDO COMO ROUTER EN UN PC CON SISTEMA OPERATIVO LINUX

RAFAEL RICARDO MANTILLA GÜIZA

UNIVERSIDAD INDUSTRIAL DE SANTANDER
ESCUELA DE INGENIERIAS ELECTRICA ELECTRONIA Y TELECOMUNICACIONES
ESPECIALIZACION EN TELECOMUNICACIONES
BUCARAMANGA
2005

INSTALACION, CONFIGURACION Y PUESTA EN MARCHA DEL SOFTWARE ZEBRA DE LIBRE UTILIZACION, ACTUANDO COMO ROUTER EN UN PC CON SISTEMA OPERATIVO LINUX

RAFAEL RICARDO MANTILLA GÜIZA

Monografía presentada como requisito final
para optar el título de Especialista en Telecomunicaciones

Director

Msc. Juan Carlos Martínez

UNIVERSIDAD INDUSTRIAL DE SANTANDER
ESCUELA DE INGENIERIAS ELECTRICA ELECTONICA Y TELECOMUNICACIONES
ESPECIALIZACION EN TELECOMUNICACIONES
BUCARAMANGA
2005

Dedico este proyecto a mi hijo Ricardo Andrés que con sus escasos dos añitos ha sido para mi una infinita fuente de inspiración, fortaleza, progreso, éxito; mi señora Giovanna Fernanda y mi abuela María Elisa ya fallecida que me han brindado todo el amor, fuerza espiritual y la serenidad necesaria para lograr mis metas

AGRADECIMIENTOS

El autor expresa sus agradecimientos a:

Dr. JUAN CARLOS MARTINEZ, Ingeniero de Sistemas, director del trabajo de grado.

UNIVERSIDAD INDUSTRIAL DE SANTANDER, quien me ha dado la oportunidad de tener mejor desempeño profesional en el sector empresarial gracias al nuevo conocimiento adquirido.

ADMINISTRADORA DE PENSIONES Y CESANTIAS PORVENIR S.A, quien hizo posible que cumpliera este sueño en una realidad.

A MI ESPOSA, HIJOS, FAMILIARES y amigos, por su apoyo, colaboración y compañía incondicional.

TITULO: INSTALACION, CONFIGURACION Y PUESTA EN MARCHA DEL SOFTWARE ZEBRA DE LIBRE UTILIZACION, ACTUANDO COMO ROUTER EN UN PC CON SISTEMA OPERATIVO LINUX *

AUTOR: RAFAEL RICARDO MANTILLA G **

PALABRAS CLAVES:

ZEBRA. LINUX. GNU. PROTOCOLO. RED. WAN. LAN. TCP/IP. ENRUTAMIENTO. DAEMON. RIPv1. RIPv2. OSPFv2. BGP-4. AUTONOMO.

DESCRIPCION:

Este documento que comprende una parte del área de las telecomunicaciones como es la instalación, configuración y puesta en marcha de un enrutador por medio de software libre en un PC con sistema operativo Linux; con el fin de comunicar dos o más redes diferentes intercambiando información de una forma ágil, segura en comparación con otros dispositivos hardware tradicionales y costosos.

Los pasos dados para lograr el objetivo son: se presenta la historia abordando desde su origen hasta la situación actual con el propósito de mostrar el avance de la tecnología; se detalla la instalación y configuración explicando los comandos básicos y simulando diferentes configuraciones típicas en diferentes escenarios para ver su utilidad, desempeño permitiendo evaluar variables como costos, viabilidad, eficiencia y con base en estos resultados tomar decisiones sobre la forma mas conveniente de utilizar esta herramienta ya sea como un elemento en un plan de contingencia o como parte de un laboratorio para docencia.

---

* Monografía

** Escuela de Ingenierías Eléctrica, Electrónica y Telecomunicaciones.
Director PhD. Juan Carlos Martinez

TITLE: INSTALATION, CONFIGURATION AND BEGINNING ZEBRA SOFTWARE OF FREE USE, ACTING AS ROUTER IN PC WITH LINUX OPERATING SYSTEM *

AUTHOR: RAFAEL RICARDO MANTILLA G **

KEY WORDS:

ZEBRA. LINUX. GNU. PROTOCOL. NETWORK. WAN. LAN. TCP/IP. ROUTING. DAEMON. RIPv1. RIPv2. OSPFv2. BGP-4. AUTONOMO.

DESCRIPTION:

This document includes a part of the area telecommunications, however the installation, configuration and beginning of a router by means of free use software in a PC with Linux operating system; with the purpose of communicating two or more different networks, interchanging information quickly, safe mode in comparison with other traditional and expensive hardware devices.

The taken steps to obtain the objective: the history appears approaching from its home to the present situation in order to show the advance technology; one details to the installation and configuration explaining the basic commandos and simulating different typical configurations in different scenes to see his utility, performance allowing evaluate variables like costs, viability, efficiency and with base in these results of making decisions on the advisable form but to use this tool or like an element in a contingency plan or like part of a laboratory for teaching.

---

* Monograph

* * Electric, Electronic school of Engineering's and Telecommunications.
Managing PhD. Juan Carlos Martínez

CONTENIDO

LISTA DE FIGURAS

Pág.

INTRODUCCION

Con el rápido crecimiento de Internet y el incremento de demandas para acceder, muchas organizaciones están teniendo verdaderas dificultades para proveer la necesidad de hardware y software y así cubrir esas demandas. Bastantes organizaciones pequeñas no pueden afrontar la adquisición de dispositivos de red (routers) para conectar a Internet. Una solución posible, a corto e incluso a largo plazo es convertir ordenadores de sobremesa y PCS en routers por la instalación totalmente gratis del software correspondiente y las tarjetas de red correspondientes (NIC Network Interface Cards), si son necesarias.

Zebra es un paquete software de enrutamiento que provee TCP/IP basado en los servicios de enrutamiento con varios protocolos soportados como pueden ser RIP, OSPF y BGP. Soporta además la posibilidad de implementar estos protocolos de enrutamiento con IPV4 o IPV6.

Con este software se pretende que nuestra máquina sea usada como un servidor de rutas (route Server) y como un reflector de rutas (route reflector) tan bien como lo haría un router normal.

1. ASPECTOS GENERALES

OBJETIVO GENERAL

Recopilar información bibliográfica, implementar y poner en marcha el paquete Zebra bajo Linux en un computador personal que actuando como router permita operar los diferentes protocolos de enrutamiento como RIPv1, RIPv2, OSPFv2, BGP-4, entre redes.

OBJETIVOS ESPECIFICOS

- Recopilar información bibliográfica sobre los conceptos que el software Zebra utiliza (TCP/IP, protocolos de enrutamiento).
- Detallar la instalación y configuración de este software llevando a puesta de funcionamiento.
- Plantear escenarios para colocar el producto Zebra como reemplazo de un enrutador.
- Desarrollar un laboratorio de redes utilizando PC con propósitos educativos.
- Documentar las pruebas realizadas.

PLANTEAMIENTO DEL PROBLEMA

La globalización e influencia del Internet en el mundo como medio de comunicación ha creado la necesidad compartir recursos entre diferentes redes; que entre sus limitantes debe estimar variables como la distancia de una red a otra, el tiempo que llevaría establecer la comunicación, estabilidad de la conexión, capacidad de transferencia de datos y la seguridad que se brindará a esta comunicación.

Esto supone la inversión de grandes cantidades de dinero en enrutadores y en tecnología para lograr su implementación y puesta en marcha, estos equipos varían de precio según variables como capacidad de transferencia que eviten cuellos de botella a la hora de salir de una red a otra.

En la formación de personal calificado para realizar estas tareas hace indispensable laboratorios que reflejen y permitan crear el escenario para formar en el estudiante destrezas en este campo de la tecnología para una que una vez culminada su formación puedan contribuir a la comunidad para su desarrollo evitando solicitar mano de obra extranjera para el desarrollo de estas necesidades en nuestra sociedad.

JUSTIFICACION

Dada esta necesidad de conectar diferentes redes aparecen los enrutadores que actúan como herramientas que aseguran el encaminamiento de una comunicación a través de una red.

En el mercado se encuentra una amplia gama de estos dispositivos que acarrean altos costos, los cuales pueden ser disminuidos destinando un computador personal para que ejecute la función de enrutamiento por medio de un paquete software llamado Zebra, brindando excelentes características a bajo costo puesto que su distribución es GNU. Esta herramienta que corre en el sistema operacional Linux el cual también es de libre utilización.

Por sus excelentes prestaciones y ventajas la combinación de estos dos productos puede ser contemplada como un plan de contingencia, o vista como una gran alternativa para empresas de bajos recursos.

Ya conociendo las ventajas de funcionamiento y costos que conlleva implementar un computador personal para que actúe como un enrutador, se pueden plantear laboratorios que permitan probar configuraciones, dar la solución más óptima a las necesidades de una empresa, o como plan de contingencia que garantice la disponibilidad o continuidad del negocio, contribuyendo a la formación de personal calificado para desempeñar estas funciones.

2. ZEBRA

2.1 ¿QUE ES?

Zebra es un paquete de software de ruteo que proporciona encaminamiento basado en servicios de TCP/IP con protocolos de enrutamiento que soportan RIPv1, RIPv2, RIPng, OSPFv2, OSPFv3, BGP-4 y BGP-4+. Zebra también soporta el comportamiento especial de BGP Route Reflector y Route Server, además de los protocolos de enrutamiento tradicionales basados en IPv4, también soporta protocolos de enrutamiento basados en IPv6. El demonio de SNMP es soportado por el protocolo SMUX, Zebra proporciona también las MIBs correspondientes.

Zebra utiliza una arquitectura de software avanzada para proporcionar una gran calidad, con un motor multi servidor de encaminamiento. Zebra tiene un interfaz de usuario interactivo para cada protocolo de enrutamiento y soporta comandos de cliente en sus interfaces. Debido a su diseño es posible añadir nuevos demonios de protocolos fácilmente a Zebra, también se puede utilizar como librería para un programa cliente de interfaz de usuario.

Zebra es un software oficial GNU y está distribuido bajo la licencia GNU General Public License.

Internet ha sido desarrollado en muchos países, en entornos empresariales y domésticos. Cuando un usuario se conecta a Internet sus paquetes atravesarán muchos routers que utilizan la funcionalidad del enrutamiento TCP/IP. Un sistema con Zebra instalado actúa como router dedicado intercambiando información de rutas con otros routers utilizando los protocolos de enrutamiento.

Zebra utiliza esa información para actualizar el núcleo de las tablas de enrutamiento de forma que la información correcta esté en el lugar correcto. Zebra permite la configuración dinámica y es posible ver la información de la tabla de enrutamiento desde el interfaz de terminal de Zebra.

Añadiendo soporte al protocolo de enrutamiento, Zebra puede configurar las banderas (flags) de los interfaces, direcciones de los interfaces, rutas estáticas y muchas más cosas. Si se utiliza en una red pequeña o en una conexión xDSL, la configuración del software es muy sencilla. Lo único que hay que pensar es en levantar los interfaces e introducir unos pocos comandos sobre rutas estáticas y/o rutas por defecto. Si en cambio estamos utilizando una red más grande, o la estructura de la red cambia frecuentemente, entonces utilizaremos

la ventaja que nos ofrece Zebra sobre los protocolos de enrutamiento dinámicos, soportando protocolos como **RIP**, **OSPF**, o **BGP**.

Tradicionalmente, la configuración de un router basado en UNIX se realizaba mediante los comandos ***ifconfig*** y los comandos del tipo ***route***. El estado de las tablas se podía mostrar mediante la utilidad ***netstat***. Estos comandos solamente se podían utilizar trabajando como root. Zebra, sin embargo tiene otro método de administración. En Zebra existen dos modos de usuario. Uno es el modo *normal* y el otro es el modo de *enable (habilitado)*. El usuario de modo normal únicamente puede ver el estado del sistema, sin embargo el usuario de modo enable puede cambiar la configuración del sistema, Esta cuenta independiente de UNIX puede ser de gran ayuda para el administrador del router. Actualmente, Zebra soporta los protocolos de unicast más comunes. Los protocolos de enrutamiento Multicast como **BGMP**, **PIM-SM**, **PIM-DM** serán soportados en Zebra 2.0.

El soporte de **MPLS** está siendo programado actualmente. En el futuro, control de filtros TCP/IP, control de calidades de servicio QoS, la configuración de diffserv será añadida a Zebra. El objetivo de Zebra es conseguir un software de enrutamiento productivo de calidad y gratuito.

2.2 RESEÑA HISTORICA

El proyecto Zebra comenzó en 1996. La idea para la Zebra vino originalmente de Kunihiro Ishiguro.

Zebra ha lanzado la versión 1.0. de libre utilización, este software de enrutamiento que pueden responder rápidamente a los cambios en tecnología y ofrecer la funcionalidad que los usuarios requieren. En el caso de Zebra, la lista y los comentarios que enviaban usuarios e ingenieros han desarrollado el software en la forma que tiene hoy.

Los Rasgos de la Zebra:

Modularidad: debido a la naturaleza de multiproceso del software Zebra, se actualiza fácilmente y se mantiene. Cada protocolo puede actualizarse separadamente, mientras deja los otros protocolos y los enrutadores en línea. Esto ahorrará tiempo a administradores de red a la hora de actualizar y el mantenimiento del paquete.

La velocidad: el software de la Zebra permite a las enrutadores transferir mayor volumen de datos más rápidamente. La necesidad para la habilidad de transferir grandes cantidades de datos está aumentando como el Internet.

La fiabilidad: en caso de falla de cualquiera de los módulos del software, el enrutador puede permanecer en línea y los otros daemons protocolares continuarán operando. La falla puede diagnosticarse y puede corregirse sin tener el enrutador fuera de servicio.

2.3 GNU GENERAL PUBLIC LICENSE

Software Libre se refiere a la libertad de los usuarios para ejecutar, copiar, distribuir, estudiar, cambiar y mejorar el software. De modo más preciso, se refiere a cuatro libertades de los usuarios del software:

- La libertad de usar el programa, con cualquier propósito (libertad 0).
- La libertad de estudiar cómo funciona el programa, y adaptarlo a tus necesidades

  (libertad 1). El acceso al código fuente es una condición previa para esto.

- La libertad de distribuir copias, con lo que puedes ayudar a tu vecino (libertad 2).
- La libertad de mejorar el programa y hacer públicas las mejoras a los demás, de modo que toda la comunidad se beneficie. (libertad 3). El acceso al código fuente es un requisito previo para esto.

Un programa es software libre si los usuarios tienen todas estas libertades. Así pues, se tiene la libertad de distribuir copias, sea con o sin modificaciones, sea gratis o cobrando una cantidad por la distribución, a cualquiera y a cualquier lugar. El ser libre de hacer esto significa (entre otras cosas) que no tener que pedir o pagar permisos. También tener la

libertad de hacer modificaciones y utilizarlas de manera privada en el trabajo u ocio, sin ni siquiera tener que anunciar que dichas modificaciones existen. Si se quisieran publicar los cambios, no se tiene por qué avisar a nadie en particular, ni de ninguna manera en particular.

La libertad para usar un programa significa la libertad para cualquier persona u organización de usarlo en cualquier tipo de sistema informático, para cualquier clase de trabajo, y sin tener obligación de comunicárselo al desarrollador o a alguna otra entidad específica. La libertad de distribuir copias debe incluir tanto las formas binarias o ejecutables del programa como su código fuente, sean versiones modificadas o sin modificar (distribuir programas de modo ejecutable es necesario para que los sistemas operativos libres sean fáciles de instalar). Está bien si no hay manera de producir un binario o ejecutable de un programa concreto (ya que algunos lenguajes no tienen esta capacidad), pero se debe tener la libertad de distribuir estos formatos si se encuentra o se desarrolla la manera de crearlos.

Para que las libertades de hacer modificaciones y de publicar versiones mejoradas tengan sentido, se debe tener acceso al código fuente del programa. Por lo tanto, la posibilidad de acceder al código fuente es una condición necesaria para el software libre. Para que estas libertades sean reales, deben ser irrevocables mientras no se haga nada incorrecto; si el desarrollador del software tiene el poder de revocar la licencia aunque no se hayan dado motivos, el software no es libre. Son aceptables, sin embargo, ciertos tipos de reglas sobre

la manera de distribuir software libre, mientras no entren en conflicto con las libertades centrales. Por ejemplo, *copyleft* es la regla que implica que, cuando se redistribuye el programa, no se pueden agregar restricciones para denegar a otras personas las libertades centrales. Esta regla no entra en conflicto con las libertades centrales, sino que más bien las protege.

Así pues, quizás se haya pagado para obtener copias de software GNU, o tal vez se hayan obtenido sin ningún costo. Pero independientemente de cómo se haya conseguido las copias, siempre se tiene la libertad de copiar y modificar el software, e incluso de vender copias.

2.4 ARQUITECTURA

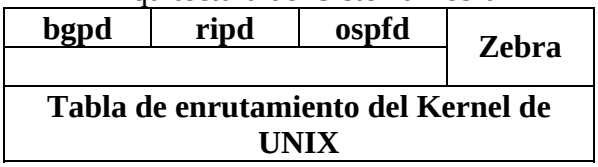
El software tradicional de enrutamiento esta compuesto por un programa o proceso único que proporciona todas las funcionalidades de los protocolos de enrutamiento. Zebra sin embargo tiene una visión distinta. Está compuesto por una colección de varios demonios que trabajan juntos para construir una tabla. Hay varios demonios de enrutamiento específicos que se ejecutan junto con zebra, el kernel gestor del enrutamiento.

El demonio ***ripd*** maneja el protocolo RIP, mientras que el demonio ***ospfd*** controla el protocolo OSPFv2. ***bgpd*** soporta el protocolo BGP-4. Para cambiar la tabla de

enrutamiento del kernel y la redistribución de rutas entre distintos protocolos de enrutamiento tenemos la tabla de enrutamiento del kernel controlada por el demonio ***zebra***. Es sencillo añadir nuevos demonios de protocolos de enrutamiento el sistema global de enrutamiento sin afectar a otro software. Para ello hay sólo es necesario ejecutar los demonios asociados a los protocolos de enrutamiento a utilizar. Realizando esta operación, el usuario puede ejecutar un determinado demonio y enviar reportes a la consola central de enrutamiento.

No es necesario ejecutar esos demonios en la misma máquina. Es posible ejecutar varias instancias del mismo demonio de enrutamiento en la misma máquina. Esta arquitectura crea nuevas posibilidades para el sistema de enrutamiento.

Arquitectura del Sistema Zebra



La arquitectura multiproceso permite un sistema más fácilmente extensible y gestionado, adicionalmente permite un sistema totalmente modular, a la vez que varios ficheros de configuración e interfaz de terminal.

Ya que cada demonio tiene su propio fichero de configuración e interfaz de terminal, cuando se quiere configurar una ruta estática, esta se configura en el fichero de configuración ***zebra***. Cuando se configura una red BGP hay que hacerlo en el fichero de

configuración ***bgpd***, esto es bastante dispendioso. Para solucionar este problema existe un interfaz shell integrado llamado ***vsh***. ***vsh*** conecta cada demonio funcionando como un Proxy para la entrada del usuario.

Zebra se planteó para ser utilizado con mecanismos de multi-hilos (multi-threaded) cuando se ejecutase en un kernel que lo soportara. Pero en este momento, la librería de hilos que viene con GNU/Linux o FreeBSD tiene varios fallos para ejecutar servicios fiables como un software de enrutamiento, así que de momento las versiones de Zebra no utilizan hilos, en vez de utilizar hilos Zebra utiliza la llamada al sistema ***select(2)*** para multiplexar los eventos. Cuando ***zebra*** se ejecuta bajo el kernel GNU/Hurd actuará como tabla de enrutamiento del kernel. Bajo GNU/Hurd, todos los servicios TCP/IP se proporcionan por los procesos de usuario llamados ***pfinet***. Zebra proporcionará toda la selección de mecanismos de enrutamiento para el proceso. Esta característica será implementada cuando GNU/Hurd sea estable.

2.4.1 Especificaciones de Funcionamiento

Plataformas Soportadas:

GNU/Zebra ha sido probado en:

- GNU/Linux 2.0.37
- GNU/Linux 2.2.x

- GNU/Linux 2.3.x
- GNU/Linux 2.4.x
- FreeBSD 2.2.8
- FreeBSD 3.x
- FreeBSD 4.x
- FreeBSD 5.x
- NetBSD 1.4
- NetBSD 1.6.x
- OpenBSD 2.5
- OpenBSD 3.x
- Solaris 2.6
- Solaris 7

Varias pilas de IPv6 están actualmente en desarrollo. Zebra soporta las siguientes pilas de IPv6. ParaBSD, se recomiendo la pila KAME IPv6. La pila Solaris IPv6 no está soportada aún.

- Pila Linux IPv6 para GNU/Linux 2.2.x y superiores
- Pila Kame IPv6 para BSD
- Pila INRIA IPv6 para BSD

Los protocolos de la asignación de ruta apoyados

El bgpd Maneja BGP-4 y protocolo de BGP-4+

El ripd Maneja RIPv1, el protocolo del v2,

El ripngd Maneja el protocolo de RIPng. [1]

El ospfd Maneja el protocolo de OSPFv2

El ospf6d Maneja el protocolo de OSPFv3 [2]

2.5 RFCs SOPORTADOS

A continuación mostramos los RFCs de los protocolos de enrutamiento soportados:

RIP-2

- RFC2453: RIP version 2 [3]
- RFC2082: RIP-2 MD5 authentication
- RFC1722: RIP version 2 protocol applicability statement

OSPF-2

- RFC1245: OSPF protocol analysis
- RFC1246: Experience with the OSPF protocol
- RFC1370: Applicability statement for OSPF
- RFC1403: BGP OSPF interaction

[1] Este protocolo RIPng es el enrutamiento RIP desarrollado para nomenclatura IPv6
[2] El protocolo OSPFv3 es el enrutamiento OSPF desarrollado para nomenclatura IPv6
[3] Rfc (Request For Comments) son documentos en los que se detalla todo lo relacionado con la tecnología, como protocolos, recomendaciones, comunicaciones, especifican estandares, informativos o que han quedado obsoletos. La principal fuente de estos documentos son el IETF (The Internet Engineering Task Force).

- RFC1584: Multicast extentions to OSPF
- RFC2328: OSPF version 2
- RFC2370: The OSPF Opaque LSA option
- RFC2740: OSPF for IPv6
- RFC3137: OSPF stub router advertisement

BGP-4

- RFC1265: BGP protocol analysis
- RFC1266: Experience with the BGP protocol
- RFC1771: A border gateway protocol 4 (BGP-4)
- RFC1772: Application of the border gateway protocol in the Internet
- RFC1773: Experience with the BGP-4 protocol
- RFC1774: BGP-4 protocol analysis
- RFC1863: A BGP/IDRP route server alternative to a full mesh enrutamiento
- RFC1966: BGP route reflection an alternative to full mesh IBGP
- RFC1997: BGP communities attribute
- RFC1998: An application of the BGP community attribute in multi-home enrutamiento
- RFC2270: using a dedicated AS for sites homed to a single provider
- RFC2796: BGP route reflection - an alternative to full mesh IBGP
- RFC3221: Commentary on inter-domain enrutamiento in the Internet

MULTICAST

- RFC1112: Host extensions for ip multicasting
- RFC2236: Internet group management protocol, version 2
- RFC2365: Administratively scoped IP multicast
- RFC2432: Terminology for IP multicast benchmarking
- RFC2588: IP multicast and firewalls
- RFC2991: Multipath issues in unicast and multicast next-hop selection
- RFC3170: IP multicast applications: challenges and solutions
- RFC3171: IANA guidelines for IPv4 multicast address assignments

OTROS DE INTERES

- RFC1058 - Enrutamiento Information Protocol. C.L. Hedrick. 1 de Julio de 1998
- RFC2080 - RIPng for IPv6. G.Malkin, R.Minnear. Enero 1997
- RFC2283 - Multiprotocol Extension for BGP-4. T.Bates, R.Chandra, D.Katz, Y. Rekhter. Febrero 1998
- RFC2545 - Use of BGP-4 Multiprotocol Extension for IPv6 Inter-Domain Enrutamiento. P. Marques, F. Dupont. Marzo 1999
- RFC1965 *Autonomous System Confederations for BGP. P. Traina. June 1996.*
- RFC2858 *Multiprotocol Extensions for BGP-4. T. Bates, Y. Rekhter, R. Chandra, D. Katz. June 2000.*
- RFC2842 *Capabilities Advertisement with BGP-4. R. Chandra, J. Scudder. May 2000.*

Cuando está habilitado el SNMP soportado, la siguiente RFC será soportada

- RFC1227 - SNMP MUX protocol and MIB. M.T. Rose. Mayo 1991.
- RFC1657 - Definitions of Managed Objects for the Fourth Version of the Border Gateway Protocol (BGP-4) using SMIv2. S. Willis, J.Burruss, J. Chu, Editor. Julio 1994.
- RFC1850 - OSPF Version 2 Management Information Base. F. Baker, R. Coltun. Noviembre 1995.
- RFC1724 *RIP Version 2 MIB Extension. G. Malkin & F. Baker. November 1994.*

3. PROTOCOLOS DE COMUNICACIÓN

Un protocolo es una descripción formal de un conjunto de normas y convenciones que determinan el formato y la transmisión de los datos entre los diferentes dispositivos de una red.

Por supuesto, Internet tiene su propio protocolo de comunicaciones: el TCP/IP, y gracias a él es posible (entre otras cosas):

- Identificar una máquina mediante una dirección (conocida como dirección IP)
- Detectar y corregir errores durante la comunicación
- Conseguir que independientemente de la máquina (PC, MAC, Amiga, Alpha,...) y del sistema operativo utilizado (Windows XP, Windows 98, Windows 95, Windows 3.1x, MAC OS, DOS, Unix,...) que todos puedan entenderse y por tanto comunicarse.

¿Qué es Internet ? se podría definir Internet como una red de ordenadores de área o ámbito mundial, compuesta por cientos de miles de ordenadores que tienen la capacidad de comunicarse entre sí. Bajo esta sencilla definición se oculta un complicado entramado de comunicaciones, que permite conectar cualquier ordenador (con unos requisitos mínimos)

con cualquier otra máquina conectada a la red, independientemente de la distancia física que los separe. Por tal motivo Internet es definido como "La Red de Redes"

Características Principales: es mundial, Heterogénea (Internet se caracteriza por la diversidad de sus participantes y la heterogeneidad de sus formas de conexión) y no tiene dueño.

Servicios: Internet provee una serie de posibilidades para el intercambio de información, que se traduce en los distintos servicios presentes en la red. Estos servicios reflejan, de alguna manera, las múltiples opciones comunicación usadas en la vida cotidiana, entre las cuales se encuentran hablar, leer y escribir. Así, los servicios representan las múltiples actividades de intercambio de información. Los servicios más conocidos y utilizados en Internet, pueden ser categorizados en:

a) Publicación y búsqueda de información:

World Wide Web.
Video bajo demanda (Streaming media).
Servicios de transferencia de archivos (FTP).

b) Comunicaciones:

Asincrónica: Correo electrónico, listas de interés y grupos de discusión. Sincrónica: Conferencia electrónica (Chat) y videoconferencia.

¿Como Funciona? Arquitectura Cliente -- Servidor

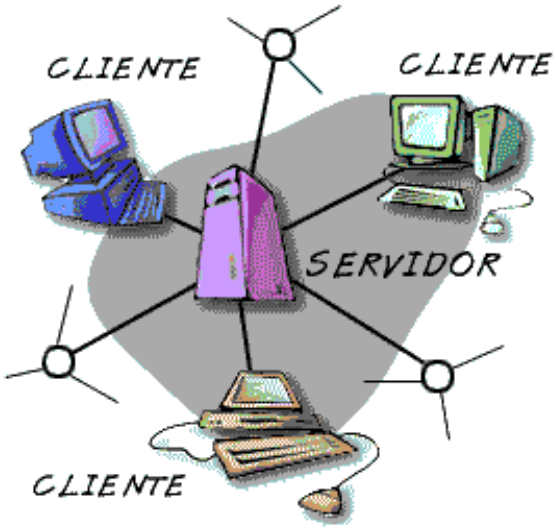


Figura 1. Arquitectura de Internet

La filosofía de funcionamiento de Internet se basa en dos conceptos: servidores y clientes. Una máquina servidor es aquella que contiene información que puede ser consultada por los usuarios. Una máquina cliente es aquella que no ofrece información, sino lo contrario, la va buscando. Por tanto, los clientes acceden a los servidores para obtener información.

3.1 TCP/IP: UNA FAMILIA DE PROTOCOLOS

La transferencia de información a través de redes se realiza de acuerdo con un Protocolo o serie de Normas. TCP/IP es el conjunto de normas utilizado en INTERNET y de hecho en otros muchos tipos de redes. Esta nomenclatura es un acrónimo de protocolo de control de transmisión / protocolo de INTERNET (Transmision Control/INTERNET Protocol). Esto se refiere a un conjunto de estándares que definen como deben establecerse las comunicaciones entre ordenadores, incluso si los ordenadores son diferentes unos de otros y están separados por grandes distancias.

Las direcciones IP están conformadas por una parte que identifica la red entre todas las que hay conectadas en Internet. La otra parte identifica el ordenador concreto de entre todos los que hay conectados dentro de esa misma red. IP y TCP son solo los más conocidos de todos los protocolos Internet.

La base de la red es IP, que es protocolo que comunica las máquinas entre sí, atravesando otras redes y pasarelas.

Del control de errores se encarga ICMP. Para localizar la dirección de cualquier máquina a partir de su nombre y dominio, se utiliza DNS.

Por encima se implementan TCP (añade fiabilidad a las comunicaciones, controlando corrección extremo a extremo, con independencia de las trayectorias de cada datagrama individual y la suerte que haya podido sufrir) y UDP (no orientado a la conexión). Estos protocolos permiten que varios programas de un mismo ordenador se pueden comunicar concurrentemente con programas de otros.

En los niveles más altos se encuentran otros protocolos, algunos estándares de TCP/IP (SMTP, SNMP) y otros específicos de vendedores, relacionados con aplicaciones finales. Las 4 clásicas (telnet, ftp, e-mail y news), van ampliándose con otras de diverso éxito (gopher, WWW...) y aplicaciones comercializadas por empresas privadas.

La familia de protocolos en los que se basa Internet, de los que TCP e IP son los más conocidos, es también aplicable en redes privadas, no conectadas a Internet, y que pueden tener la misma variedad de tecnologías subyacentes (típicamente redes de área local interconectadas por enlaces WAN).

3.2 PROTOCOLOS DE ENRUTAMIENTO

El proceso de lograr que cada máquina de una red se pueda comunicar con otra en la Internet denomina enrutamiento. Sin éste, la máquina estaría limitada sólo a una red local,

definida por el dominio de difusión (broadcast). El enrutamiento permite que el tráfico de una red busque el camino óptimo a un destino en cualquier lugar del mundo, pasando eventualmente a través de varias redes. Como administradores de redes es necesario asegurar que las rutas del sistema estén correctamente configuradas.

Elementos Básicos del Enrutamiento:

- La determinación de la ruta óptima para alcanzar a un destino requiere un conocimiento profundo por parte del router de la topología de la red.
- En el momento en el que el router conoce la topología de la red y los diferentes caminos por los que puede discurrir la información, entonces de decide la ruta preferida en ese momento y en esas condiciones para ser utilizada por ese paquete.
- Una vez decidida la ruta el paquete se sitúa en el interfaz de salida a esa ruta (switching).
- Para poder realizar un enrutamiento de calidad es necesario que haya una coherencia en el esquema de direccionamiento de las redes.

Direccionamiento de red y de host

- Un router utiliza direcciones de red para encontrar la ubicación de un host en una

Red, dentro de esta, el router local será el encargado de entregar el paquete al host destinatario.

Selección de la ruta y conmutación de paquetes

- La función de enrutamiento establece la ruta óptima desde un origen hasta un destino.
- La función de switching en un router es la responsable de situar un paquete que ha entrado por un interfaz en el interfaz necesario para que pueda llegar a su destino una vez realizada la función de enrutamiento.

Funcionamiento del protocolo de la capa de red:

- Cuando un host quiere acceder a una red distinta, éste envía un paquete a la dirección del interfaz del router conectado directamente a la LAN.
- El router examina la cabecera del paquete que ha recibido para obtener la dirección de la red de destino y la busca en su tabla de encaminamiento.
- El paquete se vuelve a encapsular y se coloca en la interfaz de salida.

Configuraciones comunes de enrutamiento

Ruta Mínima: una red completamente aislada de otra red TCP/IP requiere solo de rutas mínimas. Las rutas mínimas son creadas por el comando ip address al momento de configurar una interfaz. Las rutas mínimas son: la ruta de red local y la ruta para loopback. En linux es necesario crear la interfaz y la ruta. Una entrada es la ruta a la red 192.168.1.0 a través de eth0. La otra entrada es la ruta loopback a localhost establecida cuando se creó el interface lo.

Observe que sólo tenemos la ruta loopback y la ruta 192.168.1.0, por lo que la máquina sólo se podrá comunicar con otras máquinas dentro de la misma red. Esto es fácil de verificar con el comando ping[4].

#ping 192.168.1.2
PING 192.168.1.2: 56 data bytes
64 bytes from 192.168.1.2: icmp_seq=0 ttl= 234 time=110.0 ms
64 bytes from 192.168.1.2: icmp_seq=1 ttl= 234 time=100.7 ms
---- 192.168.1.2 ping statistics----
2 packets transmitted, 2 packet received, 0% packets loss
round-trip (ms) min/avg/max = 100/105/110 ms

Enrutamiento Estático: una red con un número pequeño de enrutadores puede ser configurada con enrutamiento estático. Para una red con un solo gateway, la mejor opción es el enrutamiento estático. Una tabla de enrutamiento estático se construye manualmente, usando el comando ip route. Las tablas de enrutamiento estático no responden a los cambios de la red, son adecuadas cuando las rutas no cambian. Se administra de forma

[4] **Ping,** usa un mensaje **ICMP** para forzar a una máquina remota a regresar un mensaje de respuesta. Si ambos mensajes llegan significa que se pueden comunicar perfectamente.

manual por el administrador de la red. Los comandos necesarios para dar esta configuración se verán más adelante en el capitulo 4.2 COMANDOS BASICOS.

En la tabla de enrutamiento se pueden observar las rutas mínimas, las rutas por defecto y las rutas por omisión. El enrutador al recibir un paquete, busca la ruta por la cual debe enviarlo, tomando en cuenta primero las rutas mínimas, luego las específicas y de no encontrar el destino en ninguna de estas rutas, lo envía a la ruta por defecto.

¿Por qué utilizar encaminamiento estático?

- Seguridad: El administrador de la red puede querer ocultar información de la red que no quiere que otros routers aprendan mediante encaminamiento dinámico.
- Sólo existe una ruta: Si sólo existe una ruta a un destino es preferible el encaminamiento estático debido a que el encaminamiento dinámico tiene un gasto (principalmente de CPU y de ancho de banda) que no es necesario utilizar.

Los routers tienen que ser capaces de manipular paquetes encapsulados en distintas tramas de nivel inferior sin cambiar el direccionamiento de nivel 3 de los paquetes, es decir, por poner un ejemplo, un router tiene que ser capaz de conectar redes Ethernet y Token Ring independientemente de las capas de nivel inferior sin cambiar el direccionamiento de nivel 3.

Los routers retransmiten un paquete de un enlace de datos a otro y esto lo hace mediante la Función de Enrutamiento y la Función de Switching.

Protocolos de enrutamiento frente a protocolos enrutados.

Los Protocolo enrutado son los que un esquema de direccionamiento a nivel 3 y los Protocolo de enrutamiento son los que proporciona mecanismos para compartir información de enrutamiento como RIP, IGRP, EIGRP, OSPF. El éxito del enrutamiento dinámico depende en gran medida de las funciones básicas del router algunas de estas como el mantenimiento de la tabla de enrutamiento y distribución puntual del conocimiento en forma de actualizaciones de enrutamiento a otros routers

La regla general es:

- Usar enrutamiento estático donde se *puede*
- Usar enrutamiento dinámico donde se *debe*
- Usar rutas por defecto estáticas en las estaciones
- Usar protocolos dinámicos entre los ruteadores

Enrutamiento Dinámico: el administrador lo único que configura es la función de enrutamiento y es el mismo protocolo de enrutamiento el que se encarga de administrar los cambios de topología mediante el envío periódico de información de enrutamiento.

¿Por qué es necesario el encaminamiento dinámico? en un enlace caído, el router A tiene que buscar un camina alternativo (prot. Encaminamiento dinámico) o en el caso que se quiera un balanceo de carga para distribuir el tráfico entre varios interfaces.

Un protocolo de encaminamiento dinámico define las reglas por las cuales de rige el intercambio de información de enrutamiento con los vecinos como por ejemplo Cómo se envían las actualizaciones, Qué conocimientos contienen dichas actualizaciones, Cuándo se envían esos conocimientos y Cómo se ubican los receptores de las actualizaciones entre otras.

Cada algoritmo determina la ruta óptima de una forma diferente utilizando métricas para calcular la distancia:

- Ancho de Banda: Capacidad de transmisión de un enlace.
- Retraso: Lo que tarda en mover la información por el enlace.
- Carga: La cantidad de tráfico que hay en el enlace.
- Fiabilidad: La probabilidad de errores en ese enlace.
- Número de Saltos: La cantidad de nodos intermedios hasta el destino.
- Pulsos: Retraso de un enlace utilizando pulsos IBM (55 microsegundos)
- Coste: Valor arbitrario obtenido con cualquier ponderación de las métricas anteriores.

Tiempo de Convergencia: es el tiempo que tardan los routers en conocer la topología de la red una vez que ha habido un cambio, este tiempo de convergencia depende del algoritmo utilizado y por tanto depende del protocolo de encaminamiento que se esté utilizando.

¿Cómo intercambian la información los protocolos de vector distancia? lo primero que hacen los protocolos de vector es identificar las redes conectadas directamente a las cuales les asigna un coste 0, y a partir de aquí los routers describen el coste a cada una de las redes basándose en la información que reciben de cada vecino.

Propagación de cambios en la topología a través de routers

Al igual que con el proceso de descubrimiento de la red las actualizaciones se van intercambiando gradualmente, este intercambio hace que se vaya actualizando la tabla de encaminamiento también de forma gradual.

Temporizadores de Espera: se emplean para prevenir que los mensajes regulares de actualización se reintegren a una ruta que podría haber ido mal, estos temporizadores se utilizan en el caso que un enlace se levante y se caiga frecuentemente (flapping).

Esto se hace para no recalcular la tabla de encaminamiento continuamente ya que puede llegar a provocar una sobrecarga en los routers un cambio que puede que no haya sucedido.

Funcionamiento:

- Si un enlace se cae se pone en marcha el temporizador
- Si se recibe una ruta peor por otro interfaz se ignora
- Si se recibe una ruta mejor por otro interfaz se asume

- Si el enlace se levanta se elimina el temporizador
- Si el enlace no se levanta se marca como enlace inaccesible y se anuncia al resto de los routers. En el caso de RIP con un coste de 16.

Variedades de Protocolos de Enrutamiento:

- IGP: usados internamente en un sistema autónomo.
- EGP: usados entre sistemas autónomos. Intercambio de información de alcance.

Tipo de Protocolos	**Vector de Distancias**	**Estado del Enlace**
IGPs	GGP	OSPF
	HELLO	Integrated IS-IS
	RIP	
	IGRP	
EGPs	EGP	IDRP
	BGP*	

3.2.1 IGPs (RIP v1)

El protocolo RIP (Enrutamiento Information Protocol), tal cual lo conocemos actualmente, fue descrito por primera vez en el RFC 1058 (http://www.rfc-editor.org/rfc/rfc1058.txt) por C. Hedrick de la Rutgers University en Junio de 1988, y posteriormente fue mejorado en la RFC 2453 (http://www.rfc-editor.org/rfc/rfc2453.txt) por G.Malkin de la compañía Bay Networks en Noviembre de 1998. Desde el año 1998 el protocolo RIP se ha mantenido estable, aunque posteriormente salió la versión para Ipv6

RIP es un protocolo de enrutamiento de vector distancia muy extendido en todo el Mundo por su simplicidad en comparación a otros protocolos como podrían ser OSPF, BGP. RIP se trata de un protocolo abierto a diferencia de otros protocolos de enrutamiento como por ejemplo IGRP y EIGRP propietarios de Cisco Systems o VNN propietario de Lucent Technologies.

RIP está basado en el algoritmo de Bellman Ford y busca su camino óptimo mediante el conteo de saltos, considerando que cada router atravesado para llegar a su destino es un salto. RIP, al contar únicamente saltos, como cualquier protocolo de vector distancia no tiene en cuenta datos tales como por ejemplo ancho de banda o congestión del enlace. RIPv1 fue diseñado para funcionar en redes pequeñas de pasarela interior .

En cuanto al protocolo tenemos que tener en cuenta las tres limitaciones que C. Hedrick describe en la página 3 del RFC 1058:

- El protocolo no permite más de quince saltos, es decir, los dos routers más alejados de la red no pueden distar más de 15 saltos, si esto ocurriera no sería posible utilizar RIP en esta red.
- Problema del “conteo a infinito”. Este problema puede surgir en situaciones atípicas en las cuales se puedan producir bucles, ya que estos bucles pueden producir retardos e incluso congestión en redes en las cuales el ancho de banda sea limitado. El autor del RFC 1058 también comenta que en la realidad esto sólo puede ser un problema en redes lentas, pero el problema existe.

- El protocolo utiliza métricas fijas para comparar rutas alternativas, lo cual implica que este protocolo no es adecuado para escoger rutas que dependan de parámetros a tiempo real como por ejemplo retardos o carga del enlace.

Además de los problemas que cita el autor del protocolo tenemos que tener en cuenta que el protocolo RIPv1 es un protocolo classfull , con lo que existe el problema de la discontinuidad de redes. El problema de la discontinuidad de redes se produce en el momento que tenemos una red dividida en varias subredes y no pueden ser sumarizadas en una misma ruta, ya que físicamente cada una de las subredes está ubicada en un lugar que depende de un interfaz distinto una subred de la otra. Pero claro, en la época en la que se escribió este RFC, que era en 1988 estos problemas no estaban contemplados y con el tiempo se detectó este problema, esta es una de las razones de la existencia de RIPv2.

El protocolo de enrutamiento RIP tiene los siguientes campos:

- Dirección de destino
- Siguiente salto
- Interfaz de salida del router
- Métrica
- Temporizador

Mantener una tabla con una entrada por cada posible destino en la red. La entrada debe contener la distancia D al destino, y el siguiente salto S del router a esa red. Conceptualmente también debería de existir una entrada para el router mismo con métrica 0, pero esta entrada no existirá.

Periódicamente se enviará una actualización de la tabla a cada uno de los vecinos del router mediante la dirección de broadcast. Esta actualización contendrá toda la tabla de enrutamiento.

Cuando llegue una actualización desde un vecino S, se añadirá el coste asociado a la red de S, y el resultado será la distancia D'. Se comparará la distancia D' y si es menor que el valor actual de D a esa red entonces se sustituirá D por D'.

3.2.2 IGPs (RIP v2).

RFC 2453: RIP Versión 2. Diez años después de que se publicara la versión 1 de RIP se publicó la versión 2, por G.Malkin de la compañía Bay Networks en Noviembre de 1998 en el RFC 2453.

RIPv2 establece una serie de mejoras muy importantes con su antecesor que son las siguientes:

- Autenticación para la transmisión de información de RIP entre vecinos.
- Utilización de mascaras de red, con lo que ya es posible utilizar VLSM .
- Utilización de máscaras de red en la elección del siguiente salto, lo cual nos puede permitir la utilización de arquitecturas de red discontinuas.
- Envío de actualizaciones de tablas de RIP mediante la dirección de multicast 224.0.0.9.
- Inclusión de RIPv2 en los bloques de información de gestión (MIB).

Por supuesto además de estas mejoras RIPv2 nos permite la redistribución de rutas externas aprendidas por otros protocolos de enrutamiento.

Pero RIPv2 aunque haya tenido una serie de mejoras muy importantes desde la versión 1 del protocolo sigue teniendo una serie de carencias muy importantes como:

- Limitación en el tamaño máximo de la red. Con RIPv2 sigue existiendo la limitación de 15 saltos como tamaño máximo de la red, lo cual implica que no nos permite la utilización de RIPv2 en redes de un tamaño más grande.
- Conteo a infinito, RIPv2 sigue sin solucionar el problema del conteo hasta el infinito si se forman bucles, aunque existen técnicas externas al protocolo como pueden ser la inversa envenenada y el horizonte dividido, técnicas brevemente descritas por William Stallings en su libro “Comunicaciones y Redes de Computadoras”, las cuales consisten

básicamente en no anunciar una ruta por el interfaz por el que se ha recibido en algún momento.

- Métricas estáticas que pueden ser cambiadas por el administrador de la red, pero que no nos dan ninguna información del estado de la red.
- RIPv2 sólo permite al igual que su antecesor una ruta por cada destino, lo cual implica la imposibilidad de realizar balanceos de carga por ejemplo, lo que redunda en una pobre y poco óptima utilización de los enlaces.

RIPv2 es un protocolo que al igual que su antecesor genera muchísimo tráfico al enviar toda la tabla de enrutamiento en cada actualización, con la carga de tráfico que ello conlleva.

- RIP v1 (RFC-1058) no es compatible con CIDR. Declarado histórico.
- RIP v2 (RFC-2453) define extensiones para RIP:
  - Compatible con RIP v1.
  - Agrega la máscara para las direcciones destino en la tabla de rutas, permitiendo el uso de subredes y superredes (CIDR).
  - Permite autentificación de los enrutadores vecinos durante los mensajes de actualización.
  - Permite la definición de dominios de enrutamiento (Sistemas Autónomos).

- Introduce la opción de utilizar multicast para el envío de mensajes de actualización sólo a los miembros del grupo de enrutadores en un segmento (224.0.0.9).

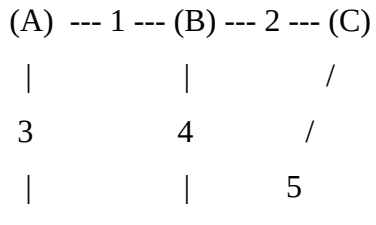
3.2.3 IGPs (OSPF)

Open Shortest Path first (OSPF) es un protocolo de enrutamiento link-state no propietario, esto quiere decir principalmente dos cosas: Primero que es de libre uso y suele estar soportados por la mayoría de los equipos destinados a ofrecer servicios a la red y Segundo el ser un link-state quiere decir que a diferencia de RIP o IGRP que son Distance-vector, no mandan continuamente la tabla de rutas a sus vecinos sino que solo lo hacen cuando hay cambios en la topología de red, de esta forma se evita el consume de ancho de banda innecesario. En un cambio de topología OSPF envía el cambio inmediatamente de forma que la convergencia de la red es mas rápida que en los distance-vector donde depende de timers asignados, de forma que en un link-state el tiempo de convergencia puede ser de 4 o 5 segundos según la red en RIP puede ser de 180 segundos. La topología de OSPF esta basada en areas conectadas de forma jerárquica. El sistema autónomo de OSPF puede ser fraccionado en diferentes areas y todas las areas estas conectadas al área de Backbone o área 0. Los router que forman parte de la red con OSPF se les denomina según su situación y su función dentro de la red de la siguiente forma:

- Internal router: Un router con todas las redes directamente conectadas a la misma área. Estos solo mantienen una copia del algoritmo de enrutamiento.
- Area Border Router: ABRs es un router que une un área al area 0 comparte la información entre las dos area y gestiona que redes se tiene que compartir entre ellas.
- Backbone Routers: Son los routers que pertenecen al area 0 y responsable de la propagación de las redes entre distintas areas. Autonomous System Boundary Routers: Son routers conectados a otros AS o Internet. También suele ser el router que intercambia entre protocolos de enrutamiento IGP y EGP.

Características:

- Es un algoritmo de estado de enlace (link state).
- En lugar de intercambiar distancias a los destinos, cada nodo mantiene un mapa de la topología de la red. Este mapa sería actualizado cada vez que haya un cambio en la topología.
- Estos mapas son utilizados para generar tablas de rutas más exactas que las que se generan con los protocolos de vector de distancias.
- Para calcular las rutas se utiliza el algoritmo de camino mas corto (Short Path First - SPF) propuesto por Dijkstra.
- Cada nodo mantiene una base de datos en la que almacenan el mapa de la red.
- Cada registro representa un nodo en la red:



De	A	Enlace	Distancia	Número
A	B	1	1	1
A	D	3	1	1
B	A	1	1	1
B	C	2	1	1
B	E	4	1	1
C	B	2	1	1
C	E	5	1	1
D	A	3	1	1
D	E	6	1	1
E	B	4	1	1
E	C	5	1	1
E	D	6	1	1

Figura 2. Diagrama de Nodos con Enrutamiento OSPF.

- Cada registro es insertado por el nodo reponsable.
- Algoritmo para poblar la base de datos:
  - Recibe el mensaje. Busca por el registro en la BD
  - Si el registro no existe, agregarlo a la BD, enviar mensaje
  - Sino, Si el número en la BD es menor que el número en el mensaje, remplazar el registro con los nuevos valores, enviar mensaje
  - Sino, Si el número en la BD es mayor, transmitir los valores en la base de datos en un nuevo mensaje a través de la interfase que recibió el mensaje original
  - Sino, Si ambos números son iguales, no hacer nada

- El algoritmo anterior permite sincronizar (bringing up adjacencies) las BD de los nodos en la red aún cuando por alguna razón la red se haya dividido temporalmente en varios segmentos.
- Seguridad en la Actualización de los mapas:
  - El algoritmo para poblar la BD incluye mensajes de confirmación
  - Los datagramas de descripción de la BD son transmitidos en forma segura
  - Cada registro de estado de enlace es protegido por un cronómetro y es removido de la BD si un datagrama de actualización no arriva en el tiempo especificado.
  - Todos los registros estan protegidos por un checksum
  - Los mensajes pueden ser autentificados, usando claves por ejemplo.

- Beneficios de OSPF:
  - Convergencia rápida y evita la creación de circulos (loops).
  - Soporte para el uso de varios tipos de medidas.
  - Se pueden calcular varias rutas para un mismo destino.
  - Permite definir jerarquías de dominios

El estándar de OSPF v.2 está descrito por John J. Moy en el RFC2328 y en el libro OSPF Anatomy of an Internet Enrutamiento Protocol, escrito por el mismo autor, y publicado por la editorial Addison-Wesley.

Entre RIP y OSPF para redes de tamaño medio-grande es preferible OSPF, ya que permite una escalabilidad muy destacada, entre otras características podemos decir que OSPF no tiene el problema de la limitación de los 15 saltos de RIP, además los tiempos de convergencia de OSPF son muchísimo mejores en todos los casos y además OSPF para el calculo de costes y rutas óptimas tiene en cuenta factores tales como el ancho de banda, lo cual permite elegir un camino con mayor número de saltos pero compuesto por conexiones de mayor velocidad.

OSPF utiliza la tecnología de estado del enlace, de forma opuesta a RIP que utiliza tecnología de vector distancia. Los router de estado del enlace mantienen una imagen común de la red e intercambian su información de enlaces desde un descubrimiento inicial hasta los cambios de la red. Los routers de estado del enlace no realizan broadcast de sus rutas periódicamente como los routers que utilizan vector distancia.

Ventajas de OSPF:

- Velocidad de convergencia: En redes grandes, la convergencia utilizando RIP puede alargarse varios minutos, hasta que la tabla completa de enrutamiento de los routers de la red se completa y se estabiliza. En OSPF el tiempo de convergencia es muchísimo menor ya que sólo se actualizan las rutas que han sido modificadas y estás son distribuidas por la red de forma rápida.

- Soporte de VLSM: RIPv1 es un protocolo de los denominados classfull, y como tal no soporta VLMS, sin embargo tenemos que recordad que RIPv2 sí soporta VLMS.
- Tamaño de la red: En un entorno RIP una red con más de 15 saltos no es viable, ya que más de 15 saltos se considera inalcanzable. Sin embargo en un entorno de enrutamiento basado en OSPF no tenemos este tipo de limitación, aunque si seguimos las especificaciones de los fabricantes de routers Cisco o Lucent Technologies nos recomiendan redes en las cuales no haya más de 400 routers por área, obviamente pueden existir más áreas, pero la única limitación física, que no de protocolo sería la de los 400 routers por área. Esta característica hace de OSPF ideal para redes medianas y grandes.
- Utilización de ancho de banda: Si utilizamos RIP estamos realizando broadcast a la red de la tabla de enrutamiento completa cada 30 segundos. Esta característica puede ser especialmente problemática sobre lentos enlaces WAN. Sin embargo OSPF utiliza multicast y sólo envía actualizaciones cuando se produce un cambio en la red.
- Selección de camino: RIP selecciona el camino óptimo contando saltos, o distancia a otros routers. Dentro de la elección de ruta óptima no entran en consideración factores como el ancho de banda restante o los retardos en la red. Sin embargo OSPF utiliza una métrica basada en ancho de banda y retardos.
- Agrupación de miembros: RIP utiliza una topología plana en la cual todos los routers forman parte de la misma red. Esta característica provoca que la comunicación entre routers tenga que navegar por la totalidad de la red, de esta forma cada cambio en un router individual afectaría al resto de los equipos de la red. Sin embargo con OSPF se introduce el concepto de áreas, lo que permite la

segmentación de la red en segmentos más pequeños. Al dividir la red en áreas se tiene que introducir el concepto de comunicación entre áreas, pero gracias a la división de la red los cambios producidos en un router de un área no afectan a la totalidad de la red, sino que sólo afecta a los routers de un área.

Ya que OSPF fue pensado y descrito para redes de un tamaño considerable al crear una red con más de 50 routers hay que tener un cuidado especial con el diseño y la planificación de la red con tal de minimizar tráfico y el constante intercambio de información de enrutamiento. La información proporcionada por OSPF a los vecinos no es la tabla de enrutamiento completa. Sin embargo, los routers que utilizan OSPF le informan a sus vecinos sobre el estado de sus conexiones o enlaces. En otras palabras los routers OSPF anuncian el estado de sus enlaces. Los routers procesan esta información y generan la base de datos de estado del enlace, la cual es esencial para poder dibujar un esquema de quien está conectado con quien. Todos los routers en un mismo área tienen que tener una base de datos del enlace idéntica. Cada router ejecuta independientemente el algoritmo SPF, también conocido como algoritmo de Dijkstra, en la base de datos del enlace con tal de determinar las mejores rutas a los destinos. El algoritmo SPF añade el coste (el cual está normalmente basado en el ancho de banda) a cada uno de los enlaces entre el router origen y el destino. Entonces el router escoge el camino con coste más bajo y añade el camino a su tabla de enrutamiento también conocida como base de datos de forwarding.

Los routers que utilizan OSPF mantienen información de sus vecinos y de sus bases de datos de adyacencia. Para simplificar el intercambio de información de enrutamiento sobre varios vecinos en la misma red, los routers que ejecutan OSPF tienen que escoger el Router Designado (DR) y el Router Designado de Backup (BDR) para servir de punto central para la actualización de rutas. Los routers que ejecutan OSPF establecen relaciones, o estados, con sus vecinos para un intercambio de información de estado más eficiente. En contraste con los protocolos de vector distancia, como RIP, los cuales realizan broadcast o multicast de su tabla de enrutamiento completa por cada interfaz, esperando que los demás routers la reciban. RIP por defecto envía cada 30 segundos sólo un único tipo de mensaje, su tabla completa de enrutamiento. Sin embargo, los routers que ejecutan OSPF disponen de cinco tipos de paquetes distintos a enviar a sus vecinos para actualizar la información de estado del enlace.

Topologías de OSPF: En OSPF podemos encontrar distintos tipos de topologías según el RFC2328, pero sin embargo ya se ha empezado a desarrollar soporte para otro tipo de topologías de forma propietaria.

- Topología de Broadcast: este tipo de topología se puede utilizar en entornos donde es posible que los routers tengan en común una red de broadcast, como podría ser una red Ethernet, Token Ring o FDDI. En este tipo de topologías los routers tienen en común una red que permite tráfico de multicast del DR con el resto de los routers.

- Topología punto a punto: este tipo de topologías son las más simples, ya que en ella sólo entran dos routers conectados de forma directa formando un único enlace. En este tipo de topologías no es necesario la elección de DB y BDR ya que sólo hay dos routers.

- Topología NBMA: en este tipo de topologías que no son de Broadcast, recordemos que NBMA son las siglas de NonBroadcast MultiAccess networks. En este tipo de topologías nos encontramos con un problema adicional, ¿Cómo envíamos mensajes de multicast en este tipo de redes?, pues bien, esta pregunta sólo tiene una contestación posible, es decir, la contestación consiste en realizar una emulación de una red de broadcast.

Estados de OSPF

- Estado Down: En el estado Down, el proceso OSPF no ha empezado a intercambiar información con ningún vecino. OSPF está esperando a entrar en el siguiente estado.
- Estado Init: Los routers que utilizan OSPF envían paquetes de tipo 1 (Hello) en intervalos regulares (por defecto 10 segundos en Zebra y en Cisco) para establecer relación con sus routers vecinos, cuando un interfaz recibe su primer paquete Hello entonces decimos que el router ha entrado en estado Init y está preparado para entrar en el siguiente estado.
- Estado Two-Way:Utilizando paquetes Hello, cada router OSPF intenta establecer una comunicación bidireccional con cada router vecino que está ubicado en la misma red IP.

Un router entra en estado two-way en el momento que se ve en una de las actualizaciones de uno de sus vecinos. El estado two-way es la relación más básica que pueden tener los routers OSPF, pero la información de enrutamiento no se intercambia en este estado. Para aprender sobre enlaces de otros routers el router tiene que tener al menos una adyacencia completa.

- Estado ExStart: Técnicamente, cuando un router y su vecino entran en estado ExStart, su conversación se caracteriza por una adyacencia, pero los routers todavía no tienen una adyacencia completa. El estado ExStart se establece utilizando paquetes de tipo 2. Entre los dos routers se utilizan paquetes hello para determinar cual de los dos es el maestro y cual es el esclavo en su relación y se intercambian paquetes de tipo 2.
- Estado Exchange: En el estado exchange se utilizan paquetes de tipo 2 para enviar al otro router su información de estado del enlace. En otras palabras, los routers describen sus bases de datos de estado del enlace al otro router. Si alguna de las rutas no está en la base de datos del enlace del router receptor de la información, este solicita una actualización completa, la cual se realiza en el estado Loading.
- Estado Loading: Después de que todas las bases de datos han sido descritas a cada router, se tiene que solicitar una información que es más completa utilizando paquetes de tipo 3. Cuando un router recibe un paquete de tipo 3, este responde con una actualización mediante un paquete de tipo 4. Los paquetes de tipo 4 describen la información de estado del enlace que es el corazón de los protocolos de enrutamiento de estado del enlace. Los paquetes de tipo 4 con respondidos con paquetes de tipo 5.
- Adyacencia Completa: Cuando termina el estado Loading, los routers están en una adyacencia completa. Cada router mantiene una lista de sus vecinos adyacentes, llamada base de datos de adyacencia. Es preciso no confundir la base de datos de

adyacencia con la base de datos de estado del enlace o con la base de datos de forwarding.

Ya que la adyacencia es necesaria para que los routers que utilizan OSPF puedan compartir su información de enrutamiento, un router tiene que estar adyacente con al menos otro router en la red IP a la que esté conectado. Si hay o no adyacencia depende del tipo de red que se esté utilizando, es decir, de qué tipo de red esté conectado los routers.

Los interfaces de un router que estén ejecutando OSPF tiene que reconocer tres tipos de redes: redes de broadcast (p.e. ethernet), NBMA (p.e. frame relay totalmente mallada) y redes punto a punto (sólo dos routers). Un administrador de red podría configurar un cuarto tipo de red: red punto a multipunto.

El tipo de red en la que esté trabajando OSPF dictará el funcionamiento del protocolo, y este a su vez puede ser optimizado por el administrador de la red.

Muchas redes se definen como redes de multiacceso porque no es posible predecir cuantos routers van a haber conectados.

3.2.4 EGPs (BGP)

El BGP o Border Gateway Protocol es un protocolo mediante el cual se intercambian prefijos los ISP registrados en Internet. Actualmente la totalidad de los ISP intercambian sus tablas de rutas a través del protocolo BGP. Este protocolo requiere un router que tenga configurado cada uno de los vecinos que intercambiarán información de las rutas que cada uno conozca. Se trata del protocolo mas utilizado para redes con intención de configurar un EGP (external gateway protocol)

La forma de configurar y delimitar la informacion que contiene e intercambia el protocolo BGP es creando lo que se conoce como Sistema Autónomo. Cada sistema autónomo (AS) tendrá conexiones o, mejor dicho, sesiones internas (iBGP) y además sesiones externas (eBGP).

El protocolo Border Gateway Protocol (BGP) está definido en el RFC1771 y actualmente está en su versión número 4. Es el protocolo más popular de los protocolos EGP y se utiliza casi sin cambios desde el año 1995.

La función de BGP es similar a la función de un router IGP como OSPF que aprende las rutas más óptimas para llegar al resto de los nodos y redes dentro de un sistema autónomo

(AS). La diferencia es que BGP trabaja con redes de diferentes sistemas autónomos, publicando sus propias redes y determinando a través de que otro sistema autónomo se puede llegar a un tercero.

BGP también tiene varias funciones de filtrado para permitir informar o no sobre las rutas que tiene y a que router externo AS lo dice. Debido a esta funcionalidad se recomienda el uso de BGP para interconectar distintos grupos wireless, en vez de seguir el uso de un protocolo IGP como OSPF.

Una vez que la red empiece a hacerse realmente grande, o empiece a considerar "la Internet" como su red, necesitará herramientas que encaminen los datos de forma dinámica. A menudo los sitios están conectados unos con otros mediante enlaces múltiples, y aparecen otros de cuando en cuando.

La Internet prácticamente ha estandarizado OSPF (RFC 2328) y BGP4 (RFC 1771),

- BGP utiliza un vector de caminos (path vector).
- Divide la Internet en sistemas autónomos.
- A cada SA (AS) se le asigna número cuando va a participar en la Internet. También existen números privados.

- Regularmente utilizado cuando se tiene mas de una conexión hacia fuera de nuestra red.
- Permite tomar una mejor decisión sobre la ruta que los datagramas deben de enviarse/recibirse.
- Agrupa prefijos internos y los anuncia a los SA vecinos.
- Definido en RFC-1771. Última versión es 4.

Seleccionando un Protocolo

- Los diferentes protocolos han sido creados para satisfacer necesidades específicas.
- Para LANs todavía muchas instituciones usan RIP.
- Para redes mas grandes se prefiere OSPF.
- La selección de un EGP dependerá del protocolo utilizado por los demás sistemas autonomos. BGP4 es lo que la mayoría utiliza.
- Al final la selección dependerá de los protocolos soportados por los equipos y que tan confortable uno se sienta con uno u otro protocolo.

4. PUESTA EN MARCHA

Zebra es un software que permite montar routers sobre sistemas operativos tipo Unix. Este software dispone de una interfaz de configuración basada en el Cisco IOS, por lo que será util a los administradores familiarizados con routers Cisco.

En este documento se describe como instalar y configurar Zebra en un equipo GNU/Linux.

4.1 INSTALACION Y CONFIGURACION

Para iniciar la instalación se debe asegurar que las tarjetas de red estén funcionando.

El primer paso consiste en obtener el paquete e instalarlo. La última versión estará en la web de www.zebra.org En los talleres se utilizara las fuentes de la última version en el momento de escribir este documento: zebra-0.94.tar.gz. o zebra-0.94.rpm [5]

[5] Para este caso, se detalla sobre la instalación con las dos presentaciones en que se puede conseguir el paquete zebra; pero para efectos de configuración y puesta en marcha se trabajo con la versión en rpm.

La instalación es la típica de cualquier paquete de código fuente; descomprimimos, compilamos e instalamos el paquete:

```
$tar xvfz zebra-0.94.tar.gz
$cd zebra-0.94.tar.gz
$./configure
$make
$su
$make install
```

Otra forma en que podemos encontrar el paquete es en formato RPM, y su instalación es sencilla:

```
$rpm -iUvh zebra-0_94-1+ldp0_310_i386 RPM
Quedando todo instalado.
internal MD5: d9511eb4863659f43da9c53a85d254b4
```

Archivos que Instala:

/etc/logrotate.d/zebra
/etc/pam.d/zebra
/etc/rc.d/init.d/bgpd
/etc/rc.d/init.d/mplsd
/etc/rc.d/init.d/ospf6d
/etc/rc.d/init.d/ospfd
/etc/rc.d/init.d/ripd
/etc/rc.d/init.d/ripngd
/etc/rc.d/init.d/zebra
/etc/zebra
/etc/zebra/bgpd.conf
/etc/zebra/mplsd.conf
/etc/zebra/ospf6d.conf
/etc/zebra/ospfd.conf
/etc/zebra/ripd.conf
/etc/zebra/ripngd.conf
/etc/zebra/vtysh.conf
/etc/zebra/zebra.conf
/usr/bin/vtysh
/usr/sbin/bgpd
/usr/sbin/mplsd
/usr/sbin/ospf6d
/usr/sbin/ospfd
/usr/sbin/ripd
/usr/sbin/ripngd
/usr/sbin/zebra
/usr/share/doc/zebra-0.94
/usr/share/doc/zebra-0.94/AUTHORS
/usr/share/doc/zebra-0.94/COPYING
/usr/share/doc/zebra-0.94/ChangeLog
/usr/share/doc/zebra-0.94/INSTALL
/usr/share/doc/zebra-0.94/NEWS
/usr/share/doc/zebra-0.94/README
/usr/share/doc/zebra-0.94/REPORTING-BUGS
/usr/share/doc/zebra-0.94/SERVICES
/usr/share/doc/zebra-0.94/TODO
/usr/share/doc/zebra-0.94/bgpd.conf.sample
/usr/share/doc/zebra-0.94/bgpd.conf.sample2
/usr/share/doc/zebra-0.94/mplsd.conf.sample
/usr/share/doc/zebra-0.94/ospf6d.conf.sample
/usr/share/doc/zebra-0.94/ospfd.conf.sample
/usr/share/doc/zebra-0.94/ripd.conf.sample
/usr/share/doc/zebra-0.94/ripngd.conf.sample
/usr/share/doc/zebra-0.94/tools
/usr/share/doc/zebra-0.94/tools/mrlg.cgi
/usr/share/doc/zebra-0.94/tools/rrcheck.pl
/usr/share/doc/zebra-0.94/tools/rrlookup.pl
/usr/share/doc/zebra-0.94/tools/zc.pl
/usr/share/doc/zebra-0.94/tools/zebra.el
/usr/share/doc/zebra-0.94/vtysh.conf.sample
/usr/share/doc/zebra-0.94/zebra.conf.sample
/usr/share/doc/zebra-0.94/zebra.html
/usr/share/info/zebra.info.gz
/usr/share/man/man1/vtysh.1.gz
/usr/share/man/man8/bgpd.8.gz
/usr/share/man/man8/ospf6d.8.gz
/usr/share/man/man8/ospfd.8.gz
/usr/share/man/man8/ripd.8.gz
/usr/share/man/man8/ripngd.8.gz
/usr/share/man/man8/zebra.8.gz
/var/log/zebra

Zebra instala 5 daemons que escuchan en puertos consecutivos y un daemon supervisor. Los daemons y en que puertos escuchan son:

Servicio	**Puerto Protocolo**	**Comentario**
zebrasrv	2600 tcp	Servicio zebra
zebra	2601 tcp	zebra vty (Terminales Virtuales de Interfaz).
ripd	2602 tcp	RIPd vty
ripngd	2603 tcp	RIPngd vty
ospfd	2604 tcp	ODPFd vty
bgpd	2605 tcp	BGPd vty
ospf6d	2606 tcp	OSPF6d vty

Utilice la forma que usted considere adecuada para iniciar los daemons en su sistema, estos están ubicados en /usr/local/sbin para instalaciones desde el código fuente.

En instalaciones RPM puede arrancar los daemons de la siguiente manera:

/etc/rc-d/init.d/zebra start
/etc/rc-d/init.d/ripd start
/etc/rc-d/init.d/ripngd start
/etc/rc-d/init.d/ospfd start
/etc/rc-d/init.d/bgpd start

Archivos de Configuración

Los archivos de configuración estan en /usr/local/etc/. Con las instalación de las fuentes viene un archivo de ejemplo para cada uno de los daemons. Estos no son utilizados directamente por Zebra, hay que cambiarles el nombre. Son los siguientes:

Nombre original	Fichero de configuración
zebra.conf.sample	zebra.conf
ripd.conf.sample	ripd.conf
ripngd.conf.sample	ripngd.conf
ospf.conf.sample	ospf.conf
bgpd.conf.sample	bgpd.conf

En el caso de haber realizado una instalación a partir de un RPM no es necesario que modifique los archivos de configuración. De todas formas puede localizarlos en /etc/zebra. Este paquete es un demonio que controla el primer punto de interacción con los usuarios y para configurarlo lo hacemos a través de **/ect/zebra/zebra.conf .**

Configuración del Zebra.conf (/etc/zebra/zebra.conf)

Y para modificar el archivo lo abrimos con mcedit zebra.conf, kedit zebra.conf o con vi y ponemos:

-hostname proyectozebra
-password proyecto
-enable password proyecto

-Hostname: Especifica el nombre del routher para ejecutar Zebra.
-Password: Especifica el login al terminal zebra en bajo nivel.
-Enable password: Especifica el login al terminal Zebra en alto nivel

log file /usr/local/etc/zebra.log
guarda los logs en un fichero en particular.
service password-encryption
Encripta el password.
show version
Muestra la versión de zebra.

Los caracteres "!" y "#" indican comentario; una vez modificado ejecutamos el demonio zebra:

$ service zebra start

Asegurarse que la red este conectada al routher y que la dirección Ip es vista por el routher usando el comando: #ping dirección_IP.

En cada maquina poner la configuración de la red: $redhat-configure-network

Equipo A: cliente1	RED	192.168.1.0
	IP	192.168.1.1 o 192.168.1.1/24
	MASCARA	255.255.255.0
	PUERTA DE ENLACE	192.168.1.7
Equipo B: cliente2	RED	132.168.47.0
	IP	192.168.47.1 o 192.168.47.1/24
	MASCARA	255.255.255.0
	PUERTA DE ENLACE	192.168.47.7
Equipo C: router1	IP Eth0	192.168.1.7
	MASCARA	255.255.255.0
	IP Eth1	192.168.47.7
	MASCARA	255.255.255.0

Configuración Remota del Zebra.

K *	192.168.1.0/24	eth0
C>*	192.168.1.0/24	eth0
K*	192.168.47.0/24	eth1
C>*	192.168.47.0/24	eth2

Para modificar la configuracion de zebra de forma remota ponemos: (enable)

Configuración de las Interfaces de Zebra:

Se van a configurar los interfaces de una red LAN la misma que esta compuesta por dos tarjetas ethernet asi:

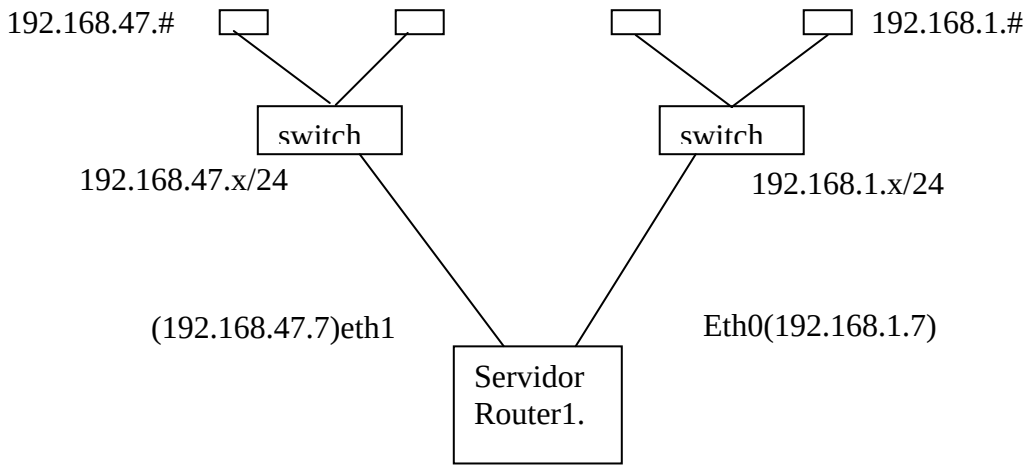


Figura 3. Modelo de 2 Redes LAN

Configuración Remota: Para esto hacemos :

$Telnet 127.0.0.1 2601
$Password proyecto
$enable
$password proyecto
$? (para ver los comandos y sus ayudas de este nivel)
$configure terminal (configuración del terminal)
Se configura las Interfaces
#interface eth0
#ip address 192.168.1.7/24
#no shutdown (subirlo)
#exit
#interface eth1
#ip address 192.168.47.7/24
#no shutdown (subirlo)
#exit
#end
#copy running –config startup-config (Guardar cambios).
#Show interface eth0 (muestra el estado de la interfaz, verifica IP).

Para realizar la configuración del router puede acceder directamente a cada uno de los protocolos (daemons) que utiliza zebra. Simplemente se hace un telnet al puerto que se desee, ejemplo para configurar rip :

$telnet localhost ripd

De todas formas esto no es necesario, puesto que zebra proporciona una herramienta que integra todos los protocolos (daemons). vtysh (Interfaz Virtual de Terminal), esta es la forma más conveniente de trabajar con zebra a través de la interfaz de usuario:

```
$ vtysh
Hello, this is zebra (version 0.94).
Copyright 1996-2002 Kunihiro Ishiguro.
proyectozebra>
```

donde "$" es el indicador de comando y "proyectozebra" el nombre de la máquina. En una sesión del intérprete pueden configurarse cualquiera de los protocolos soportados por Zebra.

4.2 COMANDOS BASICOS.

En cada modo existe un conjunto de comandos aceptados, que pueden visualizarse con "?" y cada demonio tiene sus propias opciones de invocación, todos aceptan la opción –h o --help.

A partir de aquí la configuración es similar a la de un router Cisco, si alguna vez ha configurado un router Cisco esto le resultará familiar. Hay cosas que son un poco diferentes, pero sabiendo configurar routers Cisco y con unos conocimientos básicos de redes en Linux no tendrá ningún problema.

El siguiente diagrama muestra algunos estados de Zebra y los comandos para pasar de uno a otro.

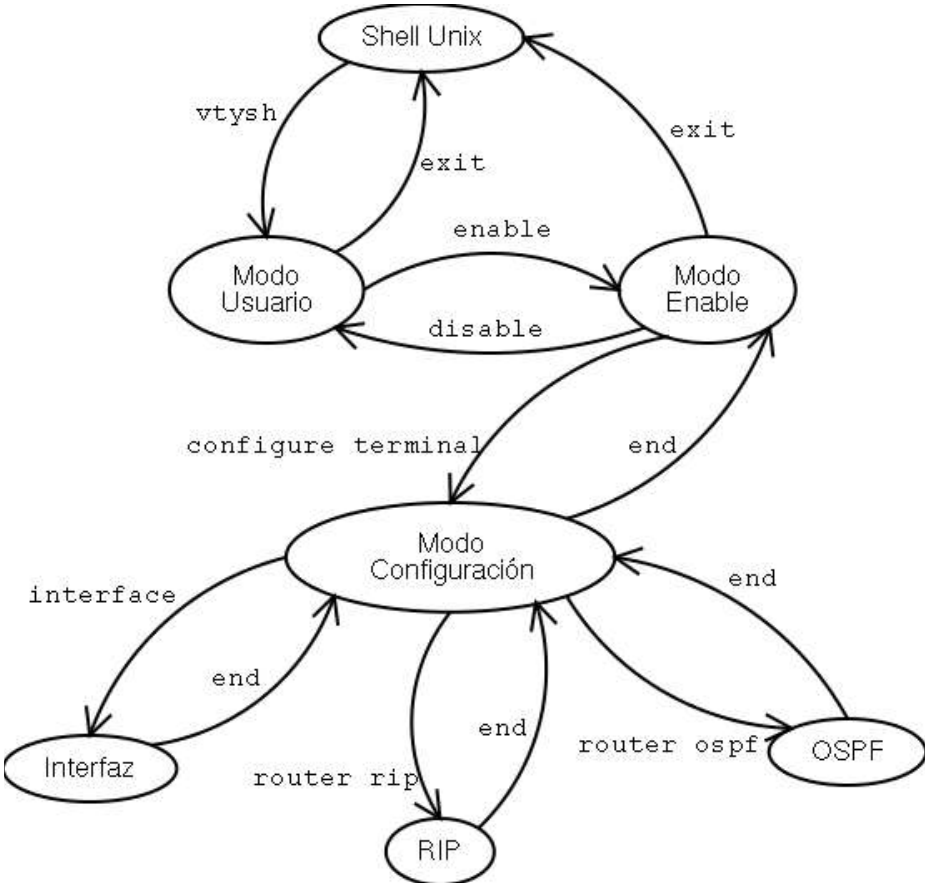


Figura 4. Algunos modos de Zebra y sus comandos de transición.

El siguiente listado muestra la mayoria de los comandos soportados por los router Cisco de la serie 2600, 1900 y Switches 2950 para recordar o familiarizarnos con estos comandos:

Modo Usuario:

enable
exit
logout
ping
show arp
show cdp
show cdp entry *
show cdp neighbors
show cdp neighbors detail
show cdp traffic
show clock
show flash
show history
show hosts
show interface
show ip protocol
show ip route
show ip route rip
show protocol
show protocols
show sessions
show users
show version
telnet
terminal
terminal history size

Priviledged Mode

clear
clock
clock set
configure
configure terminal
copy
copy flash tftp
copy running-config startup-config
copy running-config tftp
copy startup-config running-config
copy tftp flash
copy tftp run
debug ip igrp events
debug ip igrp transactions
debug ip rip
delete nvram
delete vtp
disable
disconnect
erase
erase startup-config
exit
logout
ping
reload
show
show access-list
show arp
show cdp
show cdp entry
show cdp neighbors
show cdp neighbors detail
show cdp traffic
show clock
show controllers
show flash
show frame-relay map
show frame-relay pvc
show frame-relay lmi
show history
show hosts
show interface
show interface fastethernet
show interface f0/0
show interface f0/1
show interface serial
show interface s0/0
show interface s0/1
show ip
show ip protocol
show ip route
show ip route rip
show protocol
show protocols
show running-config
show sessions
show startup-config
show termina
show trunk a
show trunk b
show users
show version
show vlan
sh vlan-membership
sh vtp status
telnet
terminal
terminal editing
terminal history size
terminal no editing
terminal monitor
trace
trace ip
vlan database
vlan # name name
write memory

Modo Configuración

access-class
access-list
banner
banner motd #
cdp
cdp enable
cdp holdtime
cdp run
cdp timer
configs register
confreg
enable
enable password
enable password level 1 password
enable password level 15 password
enable secret
end
exec-timeout
exit
frame-relay switching
hostname
interface
interface f0/0

interface f0/0.
interface fastethernet
interface s0/0
interface s0/0.
interface s0/1
interface s0/1.
interface serial
interface vlan1
ip classless
ip domain-lookup
ip host
ip route
line
line aux 0
line console 0
line vty 0 4
login
logging synchronous
network
no
no access-list
no banner
no banner motd
no cdp run
no enable
no enable password
no enable secret
no ip
no ip classless
no ip domain-lookup
no ip host
no ip route
no login
no router
no router eigrp
no router igrp
no router rip
service password encryption
router
router eigrp
router igrp
router rip
username
vtp domain domain
vtp mode server
vtp server

Modo Interface

bandwidth
clock rate
description
duplex full
encapsulation
encapsulation hdlc
encapsulation ppp
encapsulation frame-relay
encapsulation frame-relay ietf
encapsulation isl
encapsulation ppp
end
exit
frame-relay interface-dlci
frame-relay intf-type dce
frame-relay lmi-type
interface
interface f0/0
interface f0/0.
interface fastethernet
interface s0/0
interface s0/0.
interface s0/1
interface s0/1.
interface serial
ip
ip access-group
ip address
no
no cdp enable
no clock rate
no description
no encapsulation frame-relay
no frame-relay interface-dlci
no ip
no ip access-group
no ip address
no ppp authentication
no shutdown
ppp
ppp authentication
ppp authentication chap
shutdown
spanning-tree portfast
speed 100
switchport access vlan #
switchport mode access
switchport mode trunk
switchport trunk allowed vlan except # trunk on
vlan-membership static #

Ejemplo Práctico de Comandos Básicos:

```
[root@win98celeron root]# service zebra start
Iniciando zebra:                                [  OK  ]
[root@win98celeron root]# vtysh

Hello, this is zebra (version 0.94).
Copyright 1996-2002 Kunihiro Ishiguro.

win98celeron.uis.zebra.edu.co>?
  enable      Turn on privileged mode command
  exit        Exit current mode and down to previous mode
  list        Print command list
  ping        send echo messages
  quit        Exit current mode and down to previous mode
  show        Show running system information
  telnet      Open a telnet connection
  traceroute  Trace route to destination
```

La siguiente secuencia de comandos va pidiendo ayuda a medida que se digita el comando:

```
win98celeron.uis.zebra.edu.co> show ?
  bgp         BGP information
  debugging   Debugging functions (see also 'undebug')
  interface   Interface status and configuration
  ip          IP information
  ipv6        IP information
  table       default enrutamiento table to use for all clients
  version     Displays zebra version
  zebra       Zebra information

win98celeron.uis.zebra.edu.co> show ip?
  ip    IP information
  ipv6  IP information

win98celeron.uis.zebra.edu.co> show ip ?
  bgp               BGP information
  community-list    List community-list
  extcommunity-list  List extended-community list
```

```
  forwarding        IP forwarding status
  ospf              OSPF information
  prefix-list       Build a prefix list
  protocols         IP enrutamiento protocol process parameters and statistics
  rip               Show RIP routes
  route             IP enrutamiento table
```

```
win98celeron.uis.zebra.edu.co> show ip route ?
  <cr>
  bgp           Border Gateway Protocol (BGP)
  connected     Connected
  kernel        Kernel
  ospf          Open Shortest Path First (OSPF)
  rip           Enrutamiento Information Protocol (RIP)
  static        Static routes
  A.B.C.D       Network in the IP enrutamiento table to display
  A.B.C.D/M     IP prefix <network>/<length>,  e.g.,  35.0.0.0/8
  supernets-only  Show supernet entries only
```

Comando para mostrar rutas existentes:

```
win98celeron.uis.zebra.edu.co> show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP, O - OSPF,
       B - BGP, > - selected route, * - FIB route

K * 127.0.0.0/8 is directly connected, lo
C>* 127.0.0.0/8 is directly connected, lo
K>* 169.254.0.0/16 is directly connected, eth0
K * 192.168.1.0/24 is directly connected, eth0
C>* 192.168.1.0/24 is directly connected, eth0
C>* 192.168.47.0/24 is directly connected, eth0
```

Finalmente se termina la sesión.

```
win98celeron.uis.zebra.edu.co> exit
[root@win98celeron root]#
```

Los siguientes ejemplos muestran algunos comandos corrientes:

Modo Usuario.

>ping 192.168.1.1

Prueba la conectividad hacia la máquina con este número de IP.

>traceroute 192.168.1.1

Muestra los nodos atravesados hasta alcanzar la máquina destino.

>show ip forwarding

Indica si la máquina está configurada para reenviar paquetes IP, necesario para actuar como enrutador.

>show interface eth0

Muestra estado y configuración de la interfaz indicada.

>show ip rip

Muestra rutas RIP.

Modo "enable": el siguiente conjunto de comandos muestra una sesión vtysh para fijar el IP de una tarjeta ethernet en la máquina win98celeron.uis.zebra.edu.co. Los indicadores de comando permiten determinar la tarea en curso.

>enable

Entra en modo habilitado, permite configurar y realizar cambios.

#configure terminal

Inicia la configuración

#(config)#hostname win98celeron.uis.zebra.edu.co

Permite establecer el nombre para el hosts.

(config)#interface eth0

Indica la interfaz a configurar, en este caso la primera tarjeta Ethernet (nomenclatura de Linux)

(config-if)# ip address 192.168.1.7/24

Fija la dirección IP de la interfaz

(config-if)# exit

Termina la configuración de la interfaz eth0, sigue en configuración.

(config)# end

Sale del modo configuración y queda en modo enable.

#win98celeron.uis.zebra.edu.co# disable

Sale de modo enable.

#win98celeron.uis.zebra.edu.co> show interface eth0

```
Interface eth0
  index 2 metric 1 mtu 1500 <UP,BROADCAST,RUNNING,MULTICAST>
  HWaddr: 00:02:e3:13:af:3d
  inet 192.168.1.7/24 broadcast 192.168.1.255
  inet 192.168.47.7/24 broadcast 192.168.47.255 eth0:0
    input packets 0, bytes 0, dropped 0, multicast packets 0
    input errors 0, length 0, overrun 0, CRC 0, frame 0, fifo 0, missed 0
    output packets 1619, bytes 155529, dropped 0
    output errors 2, aborted 0, carrier 2, fifo 2, heartbeat 0, window 0
    collisions 0
```

Muestra el estado de la interfaz eth0, verifica la asignación de IP.

#win98celeron.uis.zebra.edu.co> exit

termina sesión vtysh.

Comandos de Interfaz: estos estan disponibles en modo configuración de terminal.

#interface eth0

Inicia configuración de la interfaz indicada, los siguientes comandos refieren a esta interfaz:

#shutdown

#no shutdown

Activa / Desactiva la interfaz.

#ip address 192.168.1.1/24

#no ip address 192.168.1.1/24

Asigna / Desasigna número IP a la interfaz.

[no] multicast

Habilita o deshabilita la bandera de multicast para el interfaz.

#exit

Termina configuración de esta interface.

#end

Sale del modo configuración.

#win98celeron.uis.zebra.edu.co# copy running-config startup-config

Guardar los cambios efectuados en el modo configuración.

Comandos de Rutas Estáticas: estos comandos estan disponibles en modo configuración de terminal

#ip route 192.168.1.0/24 192.168.1.7

Fija una ruta hacia la red 192.168.1.0/24 por la puerta de enlace (pasarela, gateway) 192.168.1.7.

#ip route 192.168.1.0/24 eth0

Fija una ruta hacia la red 192.168.1.0/24 por la interfaz eth0.

#ip route 192.168.47.0 255.255.255.0 192.168.47.7

Fija rutas indicando la máscara en notación decimal con punto.

#show ip route

Muestra las rutas disponibles, que actualmente tiene zebra en su base de datos.

Codes: K - Kernel route, C - connected, S - Static, R - RIP, B - BGP * -FIB route.

#show ipv6 route.

4.3 DIFERENTES DEMONIOS (RIPD, OSPFD, BGPD).

Demonio RIP.

Configuración de RIP.

#router rip

#no router rip

Habilita/deshabilita RIP. Para disponer de los comandos de RIP es necesario habilitarlo previamente.

#version 1

#version 2

Fija la versión de RIP a utilizar. La versión por defecto es 2.

#[no] network 192.168.47.0/24

Habilita/Deshabilita para RIP el conjunto de direcciones 192.168.47.0 a 192.168.47.255. Envia paquetes RIP por cualquier interfaz con una IP del conjunto, publica todas las redes conocidas del conjunto, independientemente de si está habilitado "redistribute connected".

#[no] network eth0

Habilita/Deshabilita el envio y recepción de paquetes RIP por esa interfaz.

#neighbor 192.168.2.3

Asigna/Desasigna un vecino RIP, necesario cuando el vecino no entiende multidifusión, no es capaz de recibir un paquete enviado a un conjunto de direcciones.

[no] passive interface eth0

Configura/Desconfigura la interfaz en modo pasivo. La interfaz en modo pasivo procesa los paquetes recibidos pero no envía información RIP más que a los vecinos especificados por el comando neighbor; no usa multidifusión.

[no] ip split-horizon

Habilita/deshabilita el horizonte dividido, esta habilitado por defecto.

Anuncio de rutas RIP.

#[no] redistribute kernel

redistribute kernel metric <0-16>

#[no] redistribute kernel route-map [route-map]

Distribuye / no distribuye las rutas existentes en el kernel del sistema operativo a las tablas de RIP.

#[no] redistribute static

#[no] redistribute static metric <0-16>
Distribuye / no distribuye las rutas estáticas existentes en el kernel del sistema operativo a las tablas de RIP.

#[no] redistribute connected
#[no] redistribute connected metric <0-16>
#[no] redistribute connected route-map [route-map]
Distribuye / no distribuye las rutas de conexiones directas del kernel del sistema operativo a las tablas de RIP. Por defecto estas rutas son anunciadas.

#[no] redistribute ospf
#[no] redistribute bgp
Distribuye / no distribuye las rutas existentes en las tablas de OSPF o BGP hacia las tablas de RIP.

#default-information originated
Propaga / no propaga rutas por defecto a través de esta máquina.
#[no] route 192.168.1.0/24
Crea / Elimina una ruta estática sólo dentro de RIP, para uso de usuarios avanzados. Se recomiend crear una ruta estática en Zebra (fuera de RIP) y redistribuírla hacia RIP mediante redistribute static.

Mostrar información de RIP.

#show ip rip
Muestra rutas de RIP. Para rutas recibidas por RIP muestra hora de envío de paquetes e información de etiquetas. Muestra también información para rutas redistribuídas por RIP.
#show ip protocols
Muestra estado actual de RIP: temporizador, filtros, versión, interfaces habilitadas para RIP e información de pares RIP.

>show ip protocols

Enrutamiento Protocol is "rip"

Sending updates every 30 seconds with +/-507,, next due in 35 seconds

Timeout after 180 seconds, garbage collect after 120 seconds

Outgoing update filter list for all interface is not set

Incoming update filter list for all interface is not set

Default redistribution metric is 1

Redistributing: kernel connected

Default version control: send version 2, receive version 2

Interface Send Recv

Enrutamiento for Networks:

ethO

ethl

1.1.1.1

203.181.89.241

Enrutamiento Information Sources:

Gateway BadPackets BadRoutes Distance Last Update

Demonio OSPFv2.

Configuración de OSPF.

#[no] router ospf

Habilita / deshabilita el proceso OSPF. ospfd todavía no soporta múltiples procesos OSPF; no es posible especificar el número de proceso de OSPF.

#ospf router-id 192.168.2.23

#ospf no router-id

Fija / elimina un número IP como identificador del enrutador; puede ser cualquiera de las interfaces conectadas a la máquina.

#[no] passive interface eth0

Habilita / deshabilita el modo pasivo de la interfaz.

#[no] timers ospf <0-4294967295> <0-4294967295>

Habilita / deshabilita nuevos valores de temporizadores. Los números indican el intervalo posible.

#[no] network 192.168.1.0/24 area 4

Habilita / deshabilita OSPF en todas las interfaces con IP en esa red; asigna número de área a la red indicada. Este comando inicia el intercambio de información de rutas con los enrutadores.

Anuncio de rutas OSPF.

#redistribute kernel

#redistribute connected

#redistribute static

#redistribute rip

#redistribute bgp

Redistribuye rutas del sistema operativo (kernel), interfaces conectadas, estáticas, RIP o BGP.

Mostrar información de OSPF.

#show ip ospf

Muestra información de OSPF.

#show ip ospf interface eth0

Muestra información de OSPF correspondiente a la interfaz eth0.

#show ip ospf neighbor

#show ip ospf neighbor ppp0

#show ip ospf neighbor detail

#show ip ospf neighbor ppp0 detail

Muestra información de OSPF correspondiente a los vecinos, eventualmente indicando la interfaz y mayor detalle.

#show ip ospf database

Mestra datos contenidos en la base de datos de OSPF.

#show ip ospf route

Muestra rutas OSPF.

Demonio de OSPF Versión 2

OSPF versión 2 es un protocolo de encaminamiento, el cual está descrito en la RFC2328-OSPF Versión2. OSPF es un protocolo de encaminamiento IGP (Interior Gateway Protocol - Protocolo de Pasarela Interna). Comparado con RIP, OSPF puede proporcionar soporte a una red escalable y un tiempo convergencia más corto. La utilización de OSPF está muy extendida en grandes redes como podrían ser backbones de ISP y redes de grandes empresas.

Demonio de BGP-4

bgpd es un demonio de BGP-4 (Border Gateway Protocol 4, Protocolo de Pasarela Fronteriza 4). BGP-4 se describe en el RFC1771. bgpd también soporta las Extensiones Multiprotocolo para BGP-4 (también conocidas como BGP-4+ o MBGP) que se describen en el RFC2283. BGP-4 es uno de los EGPs (Exterior

Gateway Protocols, Protocolos de Pasarela Exterior) y se usa para el encaminamiento entre dominios.

Empezando con bgpd

El fichero de configuración por defecto de bgpd es bgpd.conf. bgpd busca en el directorio actual antes de buscar en /usr/local/etc/bgpd.conf. Todos los commandos de bgpd deben estar configurados en bgpd.conf.

Lo primero que se debe hacer es activar el encaminamiento BGP con el comando *router bgp.* Para configurar el encaminamiento BGP, necesitas un número de SA (AS number). El número de SA es una identificación de sistema autónomo. El protocolo BGP usa el número de SA para detectar si la conexión BGP es interna o externa.

El número de SA es un número entero entre 1 y 65535. Como usar un número se describe en el RFC1930. Los números de SA del 64512 al 65535 se definen como números de uso privado. Los SA privados no deben nunca anunciarse a la Internet global. La *RFC1930 - Guidelines for creation, selection, and registration of an Autonomous System (AS)* describe como utilizar el SA. I*nternet* representa a las bien conocidas comunidades con valor 0.

#[no] router bgp NÚMERO-SA

Habilita/Destruye el proceso de protocolo BGP con el número de SA especificado. Después de este co mando, puedes introducir cualquier comando BGP. No pueden crearse un proceso BGP bajo un SA diferente a menos que se especifique Multi-instancia

#bgp router-id ROUTER-ID

Este commando especifica el router-ID (Indetificador de router). Si bgpd conecta con zebra, obtiene la información de los interfaces y de dirección. En ese caso el router-id por defecto se obtiene tomando la dirección de mayor numeración de todos los interfaces. Cuando router zebra no está habilitado, bgpd no puede obtener la información de los interfaces y router-id se fija en 0.0.0.0. En ese caso se debe especificar a mano.

#distance bgp <l-255> <l-255> <l-255>

Este comando cambia el valor de la distancia de BGP. Cada argumento es un valor de distancia para rutas externas, rutas internas y rutas locales.

#distance <l-255> A.B.C.D/M-Q

#distance <l-255> A.B.C.D/M word

Este comando configura el valor de la distancia a un destino.

Ruta BGP

#[no] network *A.B.C.D/M*

Este comando activa/desactiva el anuncio de esa red.

#network 192.168.1.0/24

Esta configuración de ejemplo nos dice que la red 192.168.1.0/24 será anunciada a todos los vecinos.

Agregación de Rutas

#[no] aggregate-address *A.B.C.D/M*

Este comando especifica que se agregen un conjunto de rutas recibidas en un PREFIJO menos especifico.

#[no] aggregate-address A.B.C.D/M summary-only

Este comando especifica una dirección agregada. Las rutas agregadas no serán anunciadas.

Redistribución a BGP

#redistribute kernel

Inyecta las rutas del kernel que no pertenecen a zebra en el proceso de BGP.

#redistribute static

Inyecta las rutas estáticas de zebra en el proceso de BGP.

#redistribute connected

Inyecta las rutas conectadas de zebra en el proceso de BGP.

#redistribute rip

Inyecta las rutas de ripd en el proceso BGP.

#redistribute ospf

Inyecta las rutas de ospfd en el proceso de BGP.

Vecino BGP

#[no]neighbor *VECINO* remote-as *NUMEROSA*

Crea un nuevo vecino/o elimina vecino cuyo SA es NÚMERO-SA. VECINO puede ser una dirección IPv4 ó IPv6.

router bgp 1

>neighbor 192.168.1.1 remote-as 2

En este caso mi encaminador, en AS-1, trata de conectar contra el encaminador de AS-2 con dirección 192.168.1.1.

Este debe ser el primer comando cuando se configura un vecino. Si remote-as no se especifica, bgpd se quejará de esta forma:

can't find neighbor 192.168.1.1

#[no] neighbor *VECINO* shutdown

Desactiva el vecino manteniendo la configuración asociada a este. Podemos desactivar un vecino con "no neighbor VECINO remote-as NÚMERO-SA pero a la vez borraremos la configuración asociada. Se utiliza esta sintaxis cuando se desee cancelar la sesión con un vecino pero preservando toda su configuración. Con la operación "no neighbor VECINO shutdown" activaremos la negociación de nuevo.

#[no] neighbor *VECINO* ebgp-multihop [TTL]

Activa/Desactiva el modo ebgp-multihop, usado para establecer sesiones BGP entre sistemas de distintos SA, con no se encuentran directamente conectados al mismo segmento de red y en ciertas configuraciones de tunnel. TTL establece el número de saltos al los cuales se encuentra el vecino, si no se especifica, el valor por defecto es el máximo, 255. Con "no" se desactiva la configuración ebgp-multihop.

#[no] neighbor *VECINO* description

Establece/Elimina una descripción del vecino en texto libre.

#[no] neighbor *VECINO* version *VERSION*

Fija la versión BGP del vecino que puede ser 4, 4+ o 4-. BGP version 4, es el valor por defecto para conecxiones BGP. BGP version 4+ significa que el vecino soporta Extensiones Multiprotocolo para BGP-4. BGP version 4- es similar pero el vecino habla Extensiones Multiprotocolo para BGP-4 en la antigua versión "Internet-Draft revision 00". Algún software de enrutamiento todavía lo usa.

#[no] neighbor *VECINO* interface *NOMBREIF*

Cuando conecta con un vecino BGP sobre una dirección IPv6 de enlace-local, debes especificar el nombre del interfaz usado para esa conexión.

#[no] neighbor *VECINO* next-hop-self

Este comando fuerza al encaminador a anunciarse como el próximo salto, para las rutas que distribuya a sus vecinos.

#[no] neighbor *VECINO* default-originate

El comportamiento por defecto de bgpd es no anunciar la ruta por defecto (0.0.0.0/0), incluso si esta se encuentra en la tabla de rutas. Este comando anunciará la ruta (0.0.0.0/0) a ese vecino concreto, si existe, o le generará una.

#[no] neighbor *VECINO* port *PUERTO*

Especifica que puerto tcp se usará para establecer la sesión BGP con ese vecino si es distinto del puerto estándar (179).

#[no] neighbor *VECINO* send-community

Instruye al demonio BGP a enviar las comunidades, asociadas a los distintos prefijos, este el comportamiento por defecto de bgpd, por lo que, debemos instruirle "no neighbor VECINO send- community" para deshabilitar el envío de comunidades.

Mostrando Rutas BGP con Camino SA

#show ip bgp
#show ip bgp *A.B.C.D*
#show ip bgp *X:X::X:X*

Este comando muestra las rutas BGP. Cuando no se especifica ruta se muestran todas las rutas BGP IPv4.

BGP table version is 0, local router ID is 10.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal

Origin codes: i - IGP, e - EGP, ? - incomplete

Network	Next Hop	Metric	LocPrf	Weight	Path
*> 1.1.1.1/32	0.0.0.0	0		32768	i

Total number of prefixes 1

#show ip bgp summary

Muestra los atributos generales de la tabla de BGP y el estado de los vecinos configurados.

#show ip bgp neighbor *[VECINO]*

Muestra un detalle extendido de un vecino y sus parámetros de configuración.

#clear ip bgp *VECINO*

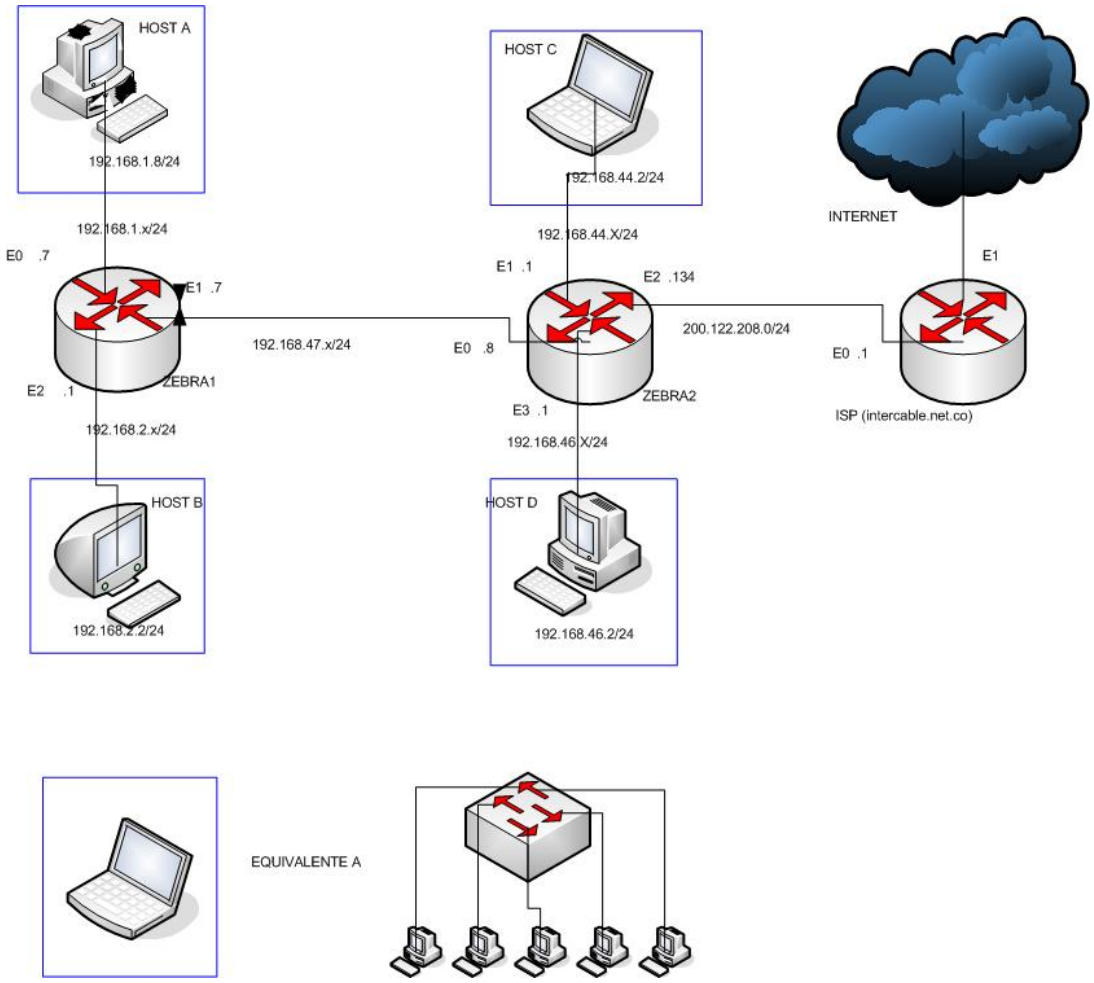
Reinicia la sesión en frío (desde cero) con VECINO

5. LABORATORIO DE REDES Y PROTOCOLOS DE ENRUTAMIENTO

DISEÑO DE LA RED Y CONEXIONES FISICAS

El diseño a continuación presenta un dos máquinas linux configuradas como router con el paquete zebra y otro como proveedor de servicio de Internet; el router1 tiene acceso a dos redes privadas (192.168.1.x/24, 192.168.2.x/24), se comunica por la red privada 192.168.47.x/24 con el router2 que tiene acceso a dos redes privadas (192.168.44.x/24, 192.168.46.x/24) y tiene acceso a un ISP (intercable.net.co) por la red pública 200.122.208.x/24 y este a su vez con Internet. Este modelo involucra 5 redes privadas y una ip pública para acceder a Internet lo que crea un excelente escenario para poner en funcionamiento los diferentes modelos de enrutamiento que garanticen la comunicación entre los diferentes nodos que hacen parte de este modelo.

Nota: Los Host del modelo puede ser incrementados sin mayor complejidad conectando un Switch, desde el cual se puede dar acceso a muchos PC más sin alterar la configuración de los routers y garantizando acceso a este modelo. Y mucho más dependiendo de la capacidad del switch (creando VLAN, o conectando varios switch en cascada, etc).



DISEÑO DE LAS SUBREDES Y ASIGNACION DE DIRECCIONES

Para configurar los dispositivos debera previamente realizar la asignación de las direcciones ip, máscaras de subred y las puertas de enlace.

HOSTS	A	B	C	D
Dirección IP	192.168.1.8	192.168.2.2	192.168.44.2	192.168.46.2
Máscara de Subred	255.255.255.0			
Puerta de Enlace	192.168.1.7	192.168.2.1	192.168.44.1	192.168.46.1

CONFIGURACION DE LOS DISPOSITIVOS

zebra1:

Tarjetas de Red

Dispositivo	eth0	eth1	eth2
Dirección IP	192.168.1.7	192.168.47.7	192.168.2.1
Mascara de Subred	255.255.255.0		

#redhat-config-network

#service network reload

Confirmar que la máquinas Linux este habilitada para ip forward:

#cat /proc/sys/net/ipv4/ip_forwarding

En 0 es deshabilitado y en 1 habilitado.

Habiliar ip forward:

#echo "1">/proc/sys/net/ipv4/ip_forwarding

Iniciamos el servicio zebra:

#service zebra start

#vtysh

Hello, this is zebra (version 0.94).

Copyright 1996-2002 Kunihiro Ishiguro.

#enable

#configure terminal

(config)#hostname zebra1

(config)#interface eth0

(config-if)#ip address 192.168.1.7/24

(config-if)#no shutdown

(config-if)#exit

(config)#interface eth1

(config-if)#ip address 192.168.47.7/24

(config-if)#no shutdown

(config-if)#exit

(config)#interface eth2

(config-if)#ip address 192.168.2.1/24

(config-if)#no shutdown

(config-if)#exit

(config)#write terminal

(config)#end

#copy running-config startup-config

#show running-config

#show interface

#show ip forwarding

IP forwarding is on

#disable

zebra2:

Tarjetas de Red

Dispositivo	eth0	eth1	eth2	eth3
Dirección IP	192.168.47.8	192.168.44.1	DHCP	192.168.46.1
Mascara de Subred	255.255.255.0			

#redhat-config-network

#service network reload

Confirmar que la máquina Linux este habilitada para ip forward:

#cat /proc/sys/net/ipv4/ip_forwarding

En 0 es deshabilitado y en 1 habilitado.

Habiliar ip forward:

#echo "1">/proc/sys/net/ipv4/ip_forwarding

Iniciamos el servicio zebra:

#service zebra start

#vtysh

Hello, this is zebra (version 0.94).

Copyright 1996-2002 Kunihiro Ishiguro.

#enable

#configure terminal

(config)#hostname zebra2
(config)#interface eth0
(config-if)#ip address 192.168.47.8/24
(config-if)#no shutdown
(config-if)#exit
(config)#interface eth1
(config-if)#ip address 192.168.44.1/24
(config-if)#no shutdown
(config-if)#exit
(config)#interface eth2
(config-if)#ip address 200.122.208.134/24
(config-if)#no shutdown
(config-if)#exit
(config)#interface eth3
(config-if)#ip address 192.168.46.1/24
(config-if)#no shutdown
(config-if)#exit
(config)#write terminal
(config)#end
#copy running-config startup-config
#show running-config
#show interface
#show ip forwarding
IP forwarding is on
#disable

PRUEBAS DE CONECTIVIDAD

Verficamos la comunicación existente hasta el momento entre los diferentes puntos (interfaces, estaciones) del escenario, por medio del comando : ping [dirección ip destino].

ORIGEN	DESTINO	CONECTIVIDAD
192.168.1.8	192.168.1.7	SI
192.168.2.2	192.168.1.8	SI
192.168.2.2	192.168.47.8	NO
192.168.1.8	192.168.44.2	NO
192.168.1.8	192.168.47.7	NO
192.168.2.2	www.yahoo.com	NO

ENRUTAMIENTO IP.

La estrategia de enrutamiento puede ser de tipo estático o dinámico. En el enrutamiento de tipo estático es tarea del administrador de la red la creación y actualización de la tabla de rutas de cada uno de los dispositivos, mientras que en el de tipo dinámico se emplean protocolos de enrutamiento como RIP, OSPF, BGP entre otros, de esta manera el algoritmo de enrutamiento es el encargado de la creación de la tabla de rutas y su continua actualización mediante el intercambio de dicha informacion con los enrutadores vecinos.

Para asegurarse que el enrutador no tenga configurada ninguna entrada en su tabla de rutas:
#vtysh
#enable
#show ip route
[no] ip route [direccion red destino] [máscara] [puerta de enlace]
Para verificar que protocolo de enrutamiento esta corriendo:
#show ip protocols

ENRUTAMIENTO ESTATICO

La creación de modo manual de la tabla de rutas, implica que debe ser creada una entrada que permita la conexión con cada una de las redes existentes (tanto redes LAN, como WAN o de cualquier tipo).

zebra2:
#vtysh
#enable
#configure terminal
#ip route 192.168.2.0/24 192.168.47.7
Con esta linea para ir a la red 192.168.2.0 con máscara 255.255.255.0 se va al siguiente router por la dirección 192.168.47.7.
#ip route 192.168.1.0/24 192.168.47.7
#ip route 0.0.0.0/0 200.122.208.1
Con esta linea para salir a Internet.
#write terminal
#copy running-config startup-config
#show ip route
Muestra las rutas estaticas definidas en el router.

zebra1:
#vtysh
#enable
#configure terminal
#ip route 192.168.44.0/24 192.168.47.8
Con esta linea para ir a la red 192.168.44.0 con máscara 255.255.255.0 se va al siguiente router por la dirección 192.168.47.8.
#ip route 192.168.46.0/24 192.168.47.8
#ip route 0.0.0.0/0 192.168.47.8
Con esta linea para salir a Internet.

#write terminal

#copy running-config startup-config

#show ip route

Muestra las rutas estaticas definidas en el router.

ENRUTAMIENTO DINAMICO RIP

Caundo el número de redes que se requiere interconectar es mayor, las rutas estáticas resultarian bastante extensas, implicando trabajo adicional para el administrador, si se tiene en cuenta que esta tabla de rutas se actualiza periódicamente, al menor cambio en la red.

zebra1:

#service zebra start

Es el archivo en /etc/zebra/zebra.conf

#service ripd start

Es el archivo en /etc/zebra/ripd.conf

#telnet 127.0.0.1 2602

#enable

#configure terminal

#router rip

#network 192.168.1.0/24

#network 192.168.2.0/24

#network 192.168.47.0/24

#write terminal

#end

#copy running-config startup-config

#vtysh

#enable

#configure terminal
#router rip
#network 192.168.1.0/24
#network 192.168.2.0/24
#network 192.168.47.0/24
#version 2
#write terminal
#end
#copy running-config startup-config
#show ip protocols
#show ip route
Las rutas leídas a través de RIP son marcadas con una R
#disable

zebra2:
#service zebra start
Es el archivo en /etc/zebra/zebra.conf
#service ripd start
Es el archivo en /etc/zebra/ripd.conf
#telnet 127.0.0.1 2602
#enable
#configure terminal
#router rip
#network 192.168.44.0/24
#network 192.168.46.0/24
#network 192.168.47.0/24
#network 200.122.208.0/24
#write terminal
#end
#copy running-config startup-config

#vtysh
#cnable
#configure terminal
#router rip
#network 192.168.44.0/24
#network 192.168.46.0/24
#network 192.168.47.0/24
#network 200.122.208.0/24
#write terminal
#end
#copy running-config startup-config
#show ip protocols
#show ip route
Las rutas leídas a través de RIP son marcadas con una R
#disable

ENRUTAMIENTO DINAMICO OSPF

Se verifica que no se encuentren previas configuraciones tanto de rutas estaticas como de rutas dadas bajo el enrutamiento RIP para esto usando el comando show ip route y show ip protocols verficamos y si es el caso se para el servicio de ripd service ripd stop; eliminando rutas estaticas con no ip route a.b.c.d/m a.b.c.d o rutas dinamicas con no network a.b.c.d/m. Para poner en funcionamiento el servicio de enrutamiento ospf es que haga broadcast a todos las rutas que conoce, el puerto para configuracion de ospf es el 2604.

zebra1:
#service zebra start
#service ospfd start
#telnet 127.0.0.1 2604

#enable
#configure terminal
#router ospf
#network 192.168.1.0 0.0.0.255 area 0
#network 192.168.47.0 0.0.0.255 area 0
#network 192.168.2.0 0.0.0.255 area 0
#write terminal
#end
#copy running-config startup-config
#show run
#exit

#vtysh
#enable
#configure terminal
#no router rip
#router ospf 100
#network 192.168.1.0 0.0.0.255 area 0
#network 192.168.47.0 0.0.0.255 area 0
#network 192.168.2.0 0.0.0.255 area 0
#write terminal
#end
#copy running-config startup-config
#show ip route
#show ip protocols
#show run
#exit
Las rutas que se encuentran con el protocolo ospf son marcadas con O.

zebra2:
#service zebra start

#service ospfd start

#telnet 127.0.0.1 2604

#enable

#configure terminal

#router ospf

#network 192.168.1.0 0.0.0.255 area 0

#network 192.168.47.0 0.0.0.255 area 0

#network 192.168.2.0 0.0.0.255 area 0

#write terminal

#end

#copy running-config startup-config

#show run

#exit

#vtysh

#enable

#configure terminal

#no router rip

#router ospf 100

#network 192.168.1.0 0.0.0.255 area 0

#network 192.168.47.0 0.0.0.255 area 0

#network 192.168.2.0 0.0.0.255 area 0

#write terminal

#end

#copy running-config startup-config

#show ip route

#show ip protocols

#show run

#exit

Las rutas que se encuentran con el protocolo ospf son marcadas con O.

ENRUTAMIENTO DINAMICO BGP

Se implementa el protocolo BGP para los routers zebra1 y zebra2, para dos areas, aunque lo cierto es que este protocolo es para implementar en redes grandes.

Esta es la relación de nodos con Sistemas Autónomos configurados.:

Zebra1 = AS 1
Zebra2 = AS 2

Zebra1:

```
!
! Zebra configuration saved from vty
! 2005/02/25 16:13:27
!
hostname zebra1-bgp
password zebra
log file /var/log/zebra/bgpd.log
!
router bgp 1
bgp router-id 192.168.47.7
redistribute connected
redistribute ospf
neighbor 192.168.47.8 remote-as 2
neighbor 192.168.47.8 ebgp-multihop 255
!
line vty
!
```

Zebra2:

!

hostname zebra2

password zebra

log file /var/log/zebra/bgpd.log

!

router bgp 2

bgp router-id 192.168.47.8

redistribute connected

redistribute ospf

neighbor 192.168.47.7 remote-as 1

neighbor 192.168.47.7 ebgp-multihop 255

!

line vty

!

Para este modelo se usan las opciones de redistribute connected y redistribute ospf, las sesiones BGP permanecieron establecidas, anunciando a los diferentes vecinos BGP las rutas aprendidas por OSPF y por kernel.

Para saber las redes que conocemos por el protocolo BGP

#show ip route bgp

B> - nos indica que es aprendida por BGP.

/24 - la mascara de red.

[200/200] - términos de métricas de redes.

vía - a través de quien conocemos la red.

3w4d - desde cuando la conocemos.

Se pueden ver los vecinos bgp configurados y su estado actual:

#show ip bgp summary

BGP router identifier xxx.xxx.xxx.xxx, local AS number XXXXX

BGP table version is 394523949, main enrutamiento table version 394523949
126358 network entries and 644473 paths using 35584112 bytes of memory
71823 BGP path attribute entries using 4027072 bytes of memory
53 BGP rrinfo entries using 1272 bytes of memory
38825 BGP AS-PATH entries using 987022 bytes of memory
1067 BGP community entries using 57186 bytes of memory
18624 BGP route-map cache entries using 297984 bytes of memory
35570 BGP filter-list cache entries using 426840 bytes of memory
Dampening enabled. 75 history paths, 8 dampened paths
1 received paths for inbound soft reconfiguration
BGP activity 417007/33881310 prefixes, 29096388/28451915 paths, scan interval 15 secs
Neighbor V AS MsgRcvd MsgSent TblVer InQ OutQ Up/Down State/PfxRcd

Se pueden ver las rutas aceptadas y aprendidas a través de un vecino bgp:
#show ip bgp neighbors xxx.xxx.xxx.xxx routes
BGP table version is 394526830, local router ID is xxx.xxx.xxx.xxx
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal
Origin codes: i - IGP, e - EGP, ? - incomplete
Network Next Hop Metric LocPrf Weight Path
* 3.0.0.0 192.168.0.1 80 0 4563 2323 8 i
*> 4.0.0.0 192.168.0.1 80 0 4563 1 i

Se pueden ver las rutas anunciadas a un vecino bgp:
#show ip bgp neighbors xxx.xxx.xxx.xxx advertised-routes
BGP table version is 394528289, local router ID is xxx.xxx.xxx.xxx
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal
Origin codes: i - IGP, e - EGP, ? - incomplete
Network Next Hop Metric LocPrf Weight Path
*>4.0.0.0/18 192.168.0.1100 100 0 4926 10718 10718 10718 10718 10718 i

Para concer una red:

#show ip route 10.10.10.10

Enrutamiento entry for 10.10.10.10/8

Known via "bgp 10121", distance 200, metric 50

Tag 3352, precedence priority (1), type internal

Last update from 192.168.0.1 3d04h ago

Enrutamiento Descriptor Blocks:

*10.10.10.10, from192.168.0.1, 3d04h ago

Route metric is 50, traffic share count is 1

AS Hops 1, BGP network version 0

CONCLUSIONES

Se brinda el soporte teórico necesario para dejar claro conceptos y temas que se relacionan con los router con los diferentes protocolos de enrutamiento de tal modo que se pueda utilizar la herramienta zebra en escenarios de la vida real.

Este documento cuenta con ejemplos prácticos, detallados desde su instalación, configuración y ejecución para facilitar la comprensión del material expuesto.

Se cumple el propósito de esta monografía que es simular escenarios que van desde escenarios sencillos hasta uno final con alto grado de complejidad, explicando paso a paso su funcionamiento; pudiéndose aplicar estos modelos en escenarios reales o en simulaciones pedagógicas para desarrollar destrezas en los temas vistos de manera muy practica y a bajos costos.

Gracias a la característica de Zebra y de Linux de software de libre utilización se puede conformar muy rápidamente un enrutador el cual puede utilizarse como herramienta pedagógica o como elemento de un plan de contingencia en una red pequeña o mediana, ya que los protocolos implementados : RIP , OSPF, BGP son los de mayor utilización en este tipo de redes.

Adicionalmente Linux aparte de permitir configurar la máquina como un router por medio del paquete zebra, permite configurar muchos otros servicios en la misma máquina sin generar más gastos de espacio y dinero.

BIBLIOGRAFIA

GALLEGO DE TORRES, Antonio. Enrutadores Cisco. España: Anaya Multimedia, Marzo de 2003. 288p.

T. Bates, R. Chandra, E. Chen. RFC2796. BGP Route Reflection. Abril 2000.

G. Malkin. RFC2453. RIP versión 2. Noviembre 1998.

J. Moy. RFC2328. OSPF Versión 2. Abril 1998.

R. Chandra, P. Traina. RFC1997. BGP Communities Attribute. Agsoto 1996

F. Baker, Editor. RFC1812. Requirements for IP Version 4 Routers. Junio 1995.

G. Malkin. RFC1722. RIP V 2 Protocol Applicability Statement. Noviembre 1994.

K. Varadhan. RFC1403. BGP OSPF Interaction. Enero 1993.

Y. Rekhter. RFC1266. Experience with the BGP Protocol. 1991

Y. Rekhter. RFC1265. BGP Protocol Analysis. Octubre 1991.

C. Hedrick. RFC1058. Enrutamiento Information Protocol. 2005.

Cybergrafía:

Cisco System, Inc. Cisco Certified Network Associate.

www.cisco.com/global/ES/edu/**ccna**_home.shtml

www.cisco.com ó www.cisco.org

GNU Zebra – Enrutamiento Software.

www.zebra.org

The Linux Documentation

www.linuxdoc.org

The Linux Home Page at Linux Online

www.linux.org

www.linux.com