Diseño e implementación de una solución de administración de tráfico de red basada en DNS y chequeos de disponibilidad.

Tesis para obtener el grado de

Magister en Redes de Datos

Universidad Nacional de La Plata

Facultad de Informática

Autor: Nicolás del Río

Directora: Mg. Lía Molinari

Co - director: Ing. Luis Marrone

Noviembre de 2015

El presente trabajo ha sido presentado y expuesto por el tesista en el XX Congreso Argentino de Ciencias de la Computación, en el marco del IX Workshop Arquitectura, Redes y Sistemas Operativos (WARSO) realizado en la Universidad Nacional de La Matanza.

Acceso al artículo: http://hdl.handle.net/10915/42219

…a mamá y papá

… a los amores de mi vida: Mariana, Agus, Santi y Emi

Agradecimientos

Quiero agradecer a la Profesora Lía Molinari por haberme guiado no sólo en este trabajo final de Tesis, sino a lo largo de toda mi carrera de Grado y Postgrado. Gracias Lía por el apoyo constante.

Al Profesor Luis Marrone por los consejos y conocimiento aportado para realizar este trabajo. A mis compañeros de trabajo y colegas que aportaron desde un mate hasta una buena idea para llevar a cabo la tarea.

Por último y no menos importante, a mi familia por su apoyo constante. Gracias por bancarme tantas horas frente a la compu.

Índice general

Índice de cuadros

Índice de figuras

Capítulo 1

Introducción

1.1 Motivación

Los servicios de red como ser publicación de contenido WWW y mail entre otros, han crecido exponencialmente en los últimos años. El acceso a la red por parte de pequeñas organizaciones, e inclusive usuarios particulares, ha propiciado que la oferta de contenidos sea cada vez mayor, sin contar, en la gran mayoría de los casos, con grandes infraestructuras de hardware, redundancia y comunicaciones. Este crecimiento conlleva a la necesidad de contar con mecanismos de replicación, que garanticen alta disponibilidad. Las grandes organizaciones tienden a configurar sus servicios críticos en servidores redundantes y redes sin puntos únicos de falla con el fin de brindar elevados niveles de SLA[1] (Service Level Agreement). Las tendencias actuales, llevan a los administradores a utilizar las técnicas anteriormente descriptas, así como también a desconsolidar los mismos en diversos centros de procesamiento de datos (CPD) con el fin de garantizar niveles mínimos de puntos únicos de falla.

Históricamente se ha utilizado el protocolo Border Gateway Protocol (BGP), como única herramienta que permite conmutar el tráfico de red que está siendo dirigido a un centro de procesamiento de datos (CPD) hacia otra locación ante eventuales fallas de red. En la actualidad, contar con un bloque de direcciones IP propio, así como también un Número de Sistema Autónomo (ASN) de BGP en Internet es muy complejo y costoso, y no cualquier usuario puede acceder a ello[2]. Es por esta razón, que surge la necesidad de contar con una alternativa de solución a la problemática planteada, que sea de mayor alcance para este conjunto de usuarios que no cuenta con una gran infraestructura de comunicaciones.

El protocolo Domain Name System (DNS), ha sido y sigue siéndolo, un gran pilar de la red de comunicaciones de Internet. Es a través del cual se permite localizar a los recursos en esta inmensa nube de contenidos. Si se pudieran cambiar los registros de resolución de DNS en base al estado de la red, sería posible direccionar los requerimientos de los usuarios hacia los enlaces o servicios que se encuentren operativos. Esta técnica podría adoptarse como opción a la utilización de un protocolo de ruteo por parte de organizaciones que posean una menor infraestructura. Es por esta razón, que se puede utilizar a este protocolo como herramienta complementaria a los protocolos de ruteo tradicionales. El protocolo BGP, por

[1] Acuerdo de nivel de servicio: http://www.sla-zone.co.uk
[2] Requerimientos para obtener un número de sistema autónomo (ASN): https://www.arin.net/resources/request/asn.html

su naturaleza, permite identificar las fallas de red y converger por sus propios medios, garantizando un nuevo camino para el ruteo de los datos solicitados. Surge entonces la pregunta:

¿Qué le falta al protocolo DNS para aproximarse al funcionamiento de BGP?

La respuesta puede parecer trivial pero no lo es. En principio es posible indicar que lo que le falta al protocolo DNS para aproximarse al funcionamiento de BGP, es una componente que se encargue de determinar los estados de la red, disparar la convergencia y garantizar la consistencia y estabilidad de la red.

1.2 Objetivos Propuestos

A lo largo del presente trabajo se analizarán los protocolos BGP y DNS, así como también los servicios de monitoreo que dotarán al servicio de resolución de nombres de una herramienta que permita contestar a la pregunta planteada. Adicionalmente se documentará el desarrollo de la herramienta "Traffic Manager", la cual permitirá gestionar servicios a través del protocolo DNS y basándose en chequeos de disponibilidad de los mismos, tomar decisiones de convergencia.

Finalmente, se realizarán y documentarán las pruebas de performance de la herramienta implementada y se evaluarán los resultados con el fin de compararlos con los resultados de convergencia arrojados por el protocolo BGP de modo tal de llegar a establecer bajo qué circunstancias es conveniente utilizar la tecnología propuesta, en qué casos resulta más favorable la utilización de BGP y qué impacto puede producir la utilización de ambas tecnologías conjuntamente.

Capítulo 2

BGP

2.1 Introducción

La red InterNet, tiene sus orígenes en su predecesora red ARPANET, la cual nació con el fin de interconectar nodos distantes para que los mismos pudieran transferir información entre ellos. Dicha red, nació en el año 1969 y en poco más de 2 años ya contaba con 40 nodos. Entre los años 1974 y 1982, se crearon una gran cantidad de redes entre las que se destacaron TELNET en 1974 que fue la versión comercial de ARPANET; USENET en 1979 para la utilización del e-mail, BITNET en 1981 que unía universidades de Estados unidos utilizando protocolos de IBM y EUNET en 1982 que unía el Reino Unido con Escandinavia y Holanda. El Protocolo utilizado por las máquinas conectadas a ARPANET se llamó NCP (Network Control Protocol o Protocolo de Control de Red), quien con el tiempo y con el fin de brindar una mayor escalabilidad, dio paso al protocolo TCP/IP en el año 1982. TCP/IP está formado no por uno, sino por varios protocolos, siendo los más importantes el protocolo TCP (Transmission Control Protocol o Protocolo de Control de Transmisión) y el Protocolo IP (Internet Protocol o Protocolo de Internet).

Con el fin de que los datos puedan viajar a través de una red TCP/IP, los mismos son divididos en fragmentos o paquetes de información que viajan de forma independiente y se ensamblan al final del proceso. Este concepto se aplica a las redes conmutadas por paquetes, que es el tipo de red que utiliza InterNet hoy en día. El protocolo IP brinda direccionamiento a los nodos de la red de modo tal que se pueda seguir un camino para que los paquetes lleguen hacia su destino. Para ello utiliza un esquema de direccionamiento y un protocolo para la toma de decisiones conocido como "protocolo de ruteo". La versión actual del protocolo IP es la 6 (definida en la RFC 2460 [1]); aunque las más utilizada, y en la que este trabajo se enfocará, es la versión 4 (definida en la RFC 791 [2]).

Con el fin de encaminar los paquetes, se define el concepto de *conmutador o router*, que opera en el nivel de red del modelo OSI[3] examinando las cabeceras de los paquetes IP. Los mismos poseen interfaces conectadas a más de una red, y su función principal es pasar los datos de una red a otra. Los routers permiten interconectar tanto redes de área local (LANs o Local Area Networks) como redes de área extensa (WANs o Wide Area Networks). Otra clave importante en este esquema, son los protocolos de ruteo. Los mismos nutren a los routers de información indispensable para encaminar los datos. En las redes LAN, se utilizan protocolos de ruteo interior (iGP o interior Gateway Protocol) como RIP[4] y OSPF[5] entre otros; o bien

[3] Modelo de interconexión de sistemas abiertos (ISO/IEC): www.iso.org
[4] RIP: Protocolo de ruteo interno definido en las RFC 1723 [3] y 2453 [4]
[5] OSPF: Protocolo de ruteo interno definido en la RFC 2328 [5]

ruteo estático. En redes del tipo WAN se utilizan protocolos de ruteo exterior (eGP o exterior Gateway Protocol). En este capítulo, se analizarán las características del protocolo BGP (Border Gateway Protocol).

2.2 Características

Los protocolos de ruteo exterior como BGP, tienen como propósito intercambiar información de ruteo entre organizaciones, conocidas también como *sistemas autónomos*. El primer protocolo de ruteo exterior que se implementó fue Gateway to Gateway Protocol (GGP) en el año 1980 y documentado en la RFC 823[6]. Lo sucedió Exterior Gateway Protocol (EGP) en el año 1982 y estandarizado en la RFC 827[7], que luego fue reemplazada por la RFC 904[8] en 1984. A medida que la red InterNet crecía, una mayor cantidad de sistemas autónomos eran interconectados con lo cual se comenzaron a divisar inconvenientes en la performance de EGP. Adicionalmente, surgieron necesidades de encaminar el tráfico de red basándose en métricas diferentes a las utilizadas, para las cuáles el protocolo no había sido implementado. Fue en ese entonces que resultó necesario crear un nuevo protocolo de ruteo conocido como BGP, que se adapte a las necesidades de crecimiento de la red. En junio de 1989 se estandarizó la primera versión del protocolo en la RCF 1105[9] (BGP1). La primera versión del estándar definió muchos de los conceptos que hoy son utilizados por el actual protocolo BGP, como formato de los mensajes y mecanismo de comunicación entre pares. La versión actual de BGP es la 4 (BGP4) y se encuentra documentada en la RFC 1771[10] y RFC 4271[11] desde el año 1995 y 2006 respectivamente. BGP 4 es el protocolo de ruteo exterior que utiliza InterNet desde el año 1995. Las principales ventajas de la versión 4 sobre sus predecesoras fueron la posibilidad de manejar *classless inter domain routing* (CIDR o enrutamiento entre dominios sin clases) y *agregación* con el fin de disminuir el tamaño de las tablas de ruteo, conceptos que serán discutidos más adelante.

BGP es un protocolo de ruteo del tipo path-vector o vector de caminos, lo que significa que cuando un router recibe información acerca de un destino, recibe la lista completa de nodos que se debe atravesar para alcanzarlo. A través de esta información se calcula el camino óptimo hacia cada punto de la red basando su decisión en la longitud del vector. Con el fin de ocultar los pormenores de cada red, BGP define el concepto de sistema autónomo como: *Un conjunto de routers que operan bajo una única política de ruteo y definen una o más organizaciones. [12]*

La métrica de BGP se basa en la cantidad de saltos, es decir, la cantidad de sistemas autónomos que debe atravesar el paquete para llegar a la red destino. De este modo, la ruta que tenga la menor cantidad de saltos, será la elegida como mejor ruta. Una característica importante de BGP, y lo que lo hace muy potente frente a otros protocolos, es la posibilidad de definir políticas de ruteo. Las políticas son implementadas en cada router, y permiten tomar decisiones diferentes a las que podrían haberse aprendido por el protocolo.

Mediante las políticas se puede manipular el ruteo y se pueden tomar decisiones acerca de dónde o hacia dónde transmitir el tráfico. Las políticas son implementadas a través de filtros, aceptando, rechazando o modificando rutas informadas por los vecinos de modo de manipular el flujo de los datos.

BGP utiliza el puerto 179 de TCP para la comunicación con sus vecinos. Esto representa una gran diferencia respecto de otros protocolos, lo cuales corren sobre IP o UDP. Para que un router pueda intercambiar información con otros, primero debe establecer una sesión con uno o más vecinos, a través de la cual intercambiarán mensajes. Dicha sesión debe ser configurada manualmente en ambos extremos y permite mantener un link virtual entre 2 dispositivos, el cual permitirá no solo el intercambio de información, sino el monitoreo del estado de la red. En caso de detectarse una caída sobre la sesión configurada, se podrá inferir que el par está presentando inconvenientes, lo que generará que deban actualizarse las tablas de ruteo para dejar de enviar información a través de él. Cuando un router BGP se enciende, el mismo establece una sesión contra cada uno de sus vecinos configurados, lo que se conoce como "establecer una vecindad", y envía los correspondientes saludos definidos en la RFC. Si la vecindad es aceptada, entonces se establece el emparejamiento o *peering*. Una vez establecida la vecindad/peering, ambos routers intercambian información, enviando una copia de sus propias tablas y recibiendo la correspondiente información de su par a través de mensajes del tipo "update". En este punto es donde las políticas de ruteo cobran un rol importante, ya que en cualquier protocolo de ruteo estándar se enviaría toda la tabla de ruteo, mientras que en BGP se enviará la información que se encuentre definida en la política, ocurriendo lo mismo al incorporar las rutas recibidas a la tabla de ruteo. Una vez que este proceso se haya completado, los routers sólo intercambiarán mensajes del tipo "keepalive" para indicar que la sesión se encuentra activa y mensajes con la porción de la tabla de ruteo que haya sufrido cambio si es que lo hubiera. Con el fin de evitar procesamiento innecesario, las actualizaciones de las tablas de ruteo solo se enviarán al iniciarse un router o bien cuando se produzca algún cambio en la red.

El protocolo BGP, define 4 tipos de mensajes que los routers pueden intercambiar con el fin de propagar la información de la red y mantener la misma consistente [13]:

- **OPEN:** Una vez establecida la conexión entre 2 routers, es el primer mensaje de BGP que se envía. Su función principal es informar al vecino acerca de la versión del protocolo que se está utilizando y el número de sistema autónomo. Adicionalmente se enviará información acerca de la duración de la sesión, identificación del router que envía el mensaje; y en el caso de configurarse autenticación, se enviará información de la misma.
- **KEEPALIVE:** Este tipo de mensaje, sirve para la confirmación del mensaje OPEN y para informar a los vecinos que la sesión se encuentra activa.

- **NOTIFICATION:** Permite cerrar la sesión, cerrando también la conexión TCP lo que causará que los vecinos deban modificar sus tablas de ruteo para dejar de enviar información a través del router que acaba de cerrar la sesión.
- **UPDATE:** Este mensaje sirve para intercambiar información de ruteo y solo es enviado a los vecinos cuando ocurre algún cambio en la topología de red. El mismo contiene las rutas a agregar o eliminar, sus atributos y la longitud de cada una entre otras. La recepción por parte de un router de este mensaje implicará la modificación de sus tablas de ruteo, y que el mismo emita un nuevo mensaje UPDATE al resto de sus vecinos notificando los cambios. Todo esto, si la política local configurada lo define.

Una de las características que hacen de BGP un protocolo potente, es la capacidad de tomar decisiones de ruteo basándose en diferentes métricas las cuales son descriptas por un conjunto de atributos y seleccionadas por una política de ruteo. Cuando un router envía información a otro a través de un mensaje UPDATE, la misma es descripta utilizando atributos. Dentro de los más importantes se pueden destacar[14]:

- **ORIGIN:** Determina si la ruta fue aprendida mediante una actualización de BGP o de un protocolo de ruteo interno iGP. Las rutas pueden ser aprendidas por otros vecinos mediante mensajes UPDATE, o bien pueden ser aprendidas a través del protocolo de ruteo interno que utilice el sistema autónomo para el manejo de rutas dentro de la organización. Este atributo permite definir prioridades al momento de seleccionar una ruta, dándole mayor prioridad a una aprendida a través de iGP y luego será preferible una aprendida por eGP (Exterior Gateway Protocol, que en este caso será BGP)
- **AS_PATH:** Contiene una lista de todos los números de sistema autónomo que debe atravesar el router para llegar al destino indicado. Este atributo tiene una especial importancia, ya que permite romper ciclos infinitos en la red (loops) y elegir la mejor ruta. Cada sistema autónomo agrega su número de ASN en este atributo para cada una de las rutas que aprende, antes de reenviarlas. De este modo se forma el camino utilizado por los routers para tomar las decisiones
- **NEXT-HOP:** Cuando un router BGP anuncia una ruta a un vecino, utiliza este atributo para indicar cuál es la dirección IP del router que se deberá utilizar para alcanzar dicho destino. Este atributo es útil, en caso de que el destino que se deba utilizar no sea el mismo router que notificó la ruta por BGP.
- **LOCAL_PREF:** Es un atributo interno definido por cada router que permite ponderar una ruta aprendida por iGP sobre otra aprendida por eGP. En caso de que la misma ruta haya sido aprendida por 2 caminos distintos, mediante este atributo se podrá definir cuál de ellas tendrá mayor prioridad sobre la otra.

- **ATOMIC_AGREGATE:** Este atributo permite indicar que la ruta en cuestión ha sido obtenida a través de la agregación de rutas más precisas. El mismo indica que las rutas han sido sumarizadas con el fin de disminuir la tabla de ruteo
- **MED (Multi-Exit-Discriminator):** Este atributo tiene ámbito de aplicación entre sistemas autónomos vecinos, y se utiliza en caso de tener más de una entrada hacia un sistema autónomo con el fin de determinar hacia donde deben ser enviados los paquetes y brindar balanceo de carga en la red.

Cuando un router recibe información de sus vecinos, lo primero que hará es verificar si su propia política de ruteo admite dicha información. En caso de no admitirla, la misma será descartada inmediatamente. Si la política acepta dicha información, entonces se verificará la nueva ruta respecto a las rutas que actualmente posee el dispositivo. Si la nueva ruta es considerada mejor que otra que actualmente existe, entonces se agregará la nueva ruta a la tabla de ruteo y se eliminará la antigua. Por último, si es que la política lo permite, se informará a los routers vecinos de dicha modificación para que los mismos repitan el procedimiento y la red pueda mantenerse en un estado coherente.

El proceso de decisión de encaminamiento de BGP respecto a otros procesos de decisión es más complejo, ya que no solo verifica la dirección de red de destino sino que también toma decisiones en base a los atributos de cada una de las rutas aprendidas. Cuando un router BGP recibe un paquete para su encaminamiento, la primera acción que realizará es verificar la dirección de red destino del mismo con el fin de evaluar si en su tabla de ruteo posee una ruta que le permita llegar a dicha red. En caso de contar con más de una ruta, se seleccionará la menor de ellas utilizando el valor de la máscara de red definida en la tabla de ruteo. En caso de seguir teniendo varias alternativas para seleccionar, analizará el atributo **LOCAL-PREF** para dicha entrada y seleccionará la ruta que posea el número mayor. Si aún tuviera más de una ruta a dicho destino, analizará el atributo **AS-PATH** (lista de sistemas autónomos que debe atravesar) y seleccionará aquella ruta que posea un camino más corto. Para finalizar el proceso, si aún hubiera que tomar una decisión respecto a que ruta utilizar, se seleccionará como ruta con mayor preferencia a aquella que haya sido aprendida a través de iGP y por último aquella que posea un atributo Multi-Exit-Discriminator (MED) más bajo. Una vez seleccionada la ruta, el paquete será conmutado al próximo salto de la red para que el proceso continúe hasta alcanzar el destino.

2.3 Interconexión de redes

Con el fin de que las organizaciones puedan intercambiar información entre ellas, las mismas deben interconectarse entre sí utilizando tecnologías disponibles. La interconexión de redes define los pilares básicos de la arquitectura de la red InterNet.

Cuando el volumen de datos que se debe transmitir desde una red hacia otra es bajo tiene sentido contratar el servicio de transporte a uno o más Proveedores de Servicios de InterNet (InterNet Service Provider, ISP). Este modo de conectividad es el que generalmente utilizan las organizaciones para unirse a la red. Cuando los volúmenes de información se incrementan hacia determinados puntos de la red, cobra sentido ampliar el esquema de interconexión, mediante la utilización de conexiones más complejas. Es en este punto donde nace la necesidad de realizar interconexiones con otras redes, con el fin de abaratar costos, ya que la métrica utilizada por los ISP para cobrar sus servicios, se basa en el volumen de tráfico que las organizaciones transfieren. La posibilidad de gestionar el tráfico de red independientemente de los ISP, devengará en un menor costo económico para las empresas.

Existen diversas técnicas para la interconexión de redes. Desde el punto de vista técnico las configuraciones son similares. La principal diferencia entre ellas reside en cuestiones comerciales y económicas. La técnica más común es la contratación del servicio a un proveedor de servicios de InterNet. Mediante esta técnica, la organización contrata el servicio, el cuál será provisto por un único proveedor que será el responsable de transportar los datos de la organización hacia otras redes. Si la empresa no cuenta con direccionamiento de InterNet propio, el ISP le proporcionará un bloque de direcciones IP que estará asignado a la organización mientras dure el contrato con el mismo. Si la empresa cuenta con direccionamiento IP propio, entonces tendrá la opción de utilizarlo a través de la red del ISP. Para ello podrá optar por dialogar BGP directamente con otros routers, para lo cual además de poseer su propio bloque de direcciones IP tendrá que contar también con un número ASN propio; o bien podrá anunciar su bloque de direcciones IP a través del ASN del proveedor al que contrata el servicio. En esta configuración el proveedor cobrará a la organización el transporte de los datos basándose en cantidad total de tráfico transmitido, capacidad máxima de enlace que la organización necesitará y otros factores.

A medida que el volumen de tráfico crece, la técnica anteriormente descripta puede resultar costosa para las organizaciones, con lo cual deberán evaluarse nuevas alternativas. Otra técnica de interconexión de redes es la conocida como *peering*, ya citada, en la cual existe un acuerdo de cooperación voluntaria entre una o más organizaciones. Esta técnica se caracteriza por no tener costo alguno de conexión, salvo el costo físico que implique la realización del cableado y/o necesidad de contar con interfaces de red en los routers. En este esquema, las organizaciones que intercambian altos volúmenes de tráfico, acuerdan una metodología de interconexión, de modo tal que el tráfico de red entre ambas

organizaciones, no tenga que ser transportado por sus proveedores de servicio de internet, sino gestionado directamente por las organizaciones. Para este tipo de conexiones generalmente se utilizan enlaces propios definidos entre las organizaciones, de modo tal de no tener que contratar el servicio a algún ISP.

Una técnica muy buscada entre organizaciones, entidades y proveedores de servicio de internet para la interconexión es la utilización de puntos neutrales, también conocidos como Network Access Point (NAP) o Internet Exchange Point (IXP). Mediante esta técnica, diversas organizaciones acuerdan un punto común de interconexión de modo tal que permita intercambiar tráfico entre varios puntos a un costo inferior. Esta técnica puede brindar grandes beneficios a las organizaciones que generan tráfico hacia puntos interconectados a los NAP o IXP, dado que el tráfico no deberá pasar a través de diversos ISP para su transporte, sino que el mismo será canalizado localmente hacia el punto común de ruteo. Esta configuración generalmente acarrea costos de mantenimiento y administración, que son divididos entre todos los miembros del NAP con el fin de poder sustentar la arquitectura de comunicación. Para poder unirse a un NAP, generalmente se deben cumplir los requisitos de poseer un número de sistema autónomo propio y un propio bloque de direcciones IP, así como también se debe negociar con cada una de las organizaciones conectadas de qué modo intercambiarán datos.

2.4 Estructura de InterNet

InterNet es una gran red de redes cuya arquitectura está basada en la interconexión de las mismas. Generalmente las conexiones forman una estructura jerárquica que va desde conexiones locales dentro de una misma ciudad, a conexiones nacionales e internacionales. Esta jerarquía de conexión permite estructurar a los ISP en niveles, de acuerdo a su grado y tipo de conectividad. Existen genéricamente 3 niveles de clasificación como se explica a continuación y se pueden observar en la figura **2.4.1**:

- **Nivel/Tier 1:** Las redes de los grandes operadores globales están clasificadas en esta categoría. Tienen la característica de poseer conexiones (generalmente de fibra óptica) por más de un continente. Desde este tipo de redes se puede acceder a cualquier red de InterNet, ya que es requisito que todas las redes de nivel 1 estén conectadas entre sí. Este tipo de redes forman el núcleo o backbone de InterNet y tienen la característica de no pagar a otro proveedor por servicios de transporte.
- **Nivel/Tier 2:** Son las redes directamente conectadas a las de nivel 1, y que también poseen conexiones entre sí. Tienen un ámbito geográfico más acotado, generalmente a un mismo continente o un conjunto de países. Su principal función es proveer servicio a las redes de nivel 3.

- **Nivel/Tier 3:** Este tipo de redes tienen la característica de vender tráfico generalmente a organizaciones o usuarios residenciales. Son el último eslabón de la jerarquía y tienen un ámbito geográfico más acotado que las redes de nivel 2. Poseen conexiones entre ellas, a través de NAPs o IXPs y compran tráfico a redes de nivel 1 o 2.

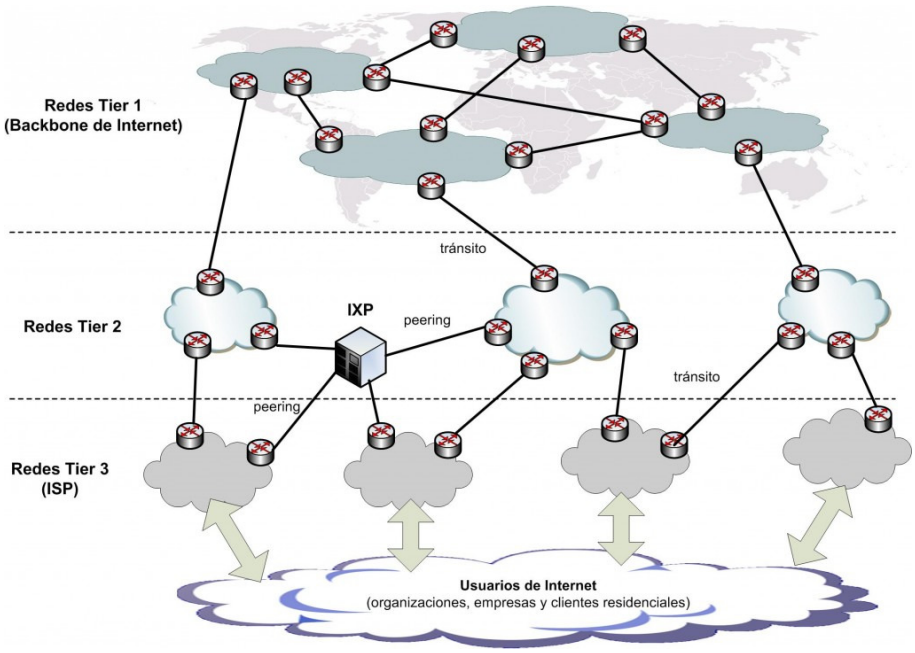


Figura 2.4.1: Estructura de la red InterNet en niveles

***Fuente**: Curso de Ruteo avanzado dictado en Enero de 2014 por la Secretaría de Postgrado en la Facultad de Informática*

La comunicación entre las redes de distinto nivel, se realiza utilizando el protocolo BGP. Cada red posee su propio conjunto de redes IP, las cuáles están representadas por uno o más números de sistema autónomo. Adicionalmente existen políticas de tráfico que indican cómo deben canalizase los datos a través de cada vínculo.

2.5 Sistemas Autónomos

2.5.1 Características

Un sistema autónomo (Autonomous System o AS) está compuesto por un conjunto de redes bajo una política única y coherente de ruteo coordinada por un administrador. Cada sistema autónomo gestiona el modo en el que fluye le información desde su red hacia InterNet y viceversa. Cada Sistema autónomo está identificado por un número único que hace referencia a un conjunto de redes administradas. Este identificador referencia a dichas redes de manera única en la red InterNet.

Hasta el año 2007, los números de sistema autónomo estaban definidos por un número entero de 16 bits, lo que permitía una asignación máxima de 65535 identificadores. Debido a la demanda, debió cambiarse el formato de los números con el fin de poder asignar una mayor cantidad de identificadores. La RFC 4983 define la implementación de números de sistema autónomo de 32 bits, los que se representan como un par de enteros *x* e *y*, números enteros de 16 bits. En la actualidad, los únicos números que se asignan son identificadores de 32 bits.

2.5.2 Clasificación

Los Números de Sistema Autónomo son asignados por la InterNet Asigned Numbers Authority[6] (IANA) a Registros regionales de InterNet (RIRs). Los RIR son los encargados de asignar regionalmente los números de sistema autónomo a las organizaciones finales. Una vez establecido, un sistema autónomo puede catalogarse de acuerdo a las siguientes definiciones, dependiendo de su modo de funcionamiento:

- **STUB:** Representa a aquellos sistemas autónomos que solo se poseen un único camino o salida para su conexión a InterNet. Tiene la característica de ser siempre el último número en un camino de sistemas autónomos para representar una ruta.
- **Mutihomed Non-Transit:** se encuentra conectado a través de varios caminos y poseen más de una conexión a InterNet. Característicamente, no rutea tráfico de terceros, es decir que todo el tráfico originado desde el propio sistema autónomo pertenece exclusivamente a sus redes.
- **Transit:** Lleva tráfico de diferentes sistemas autónomos hacia InterNet. Esta clasificación corresponde generalmente a los ISP, los cuales lucran con dicho servicio.

[6] La Internet Assigned Numbers Authority tiene autoridad sobre todo el espacio de direcciones IP utilizado en InterNet

La figura **2.5.2.1** muestra un ejemplo de cada una de las clasificaciones anteriores:

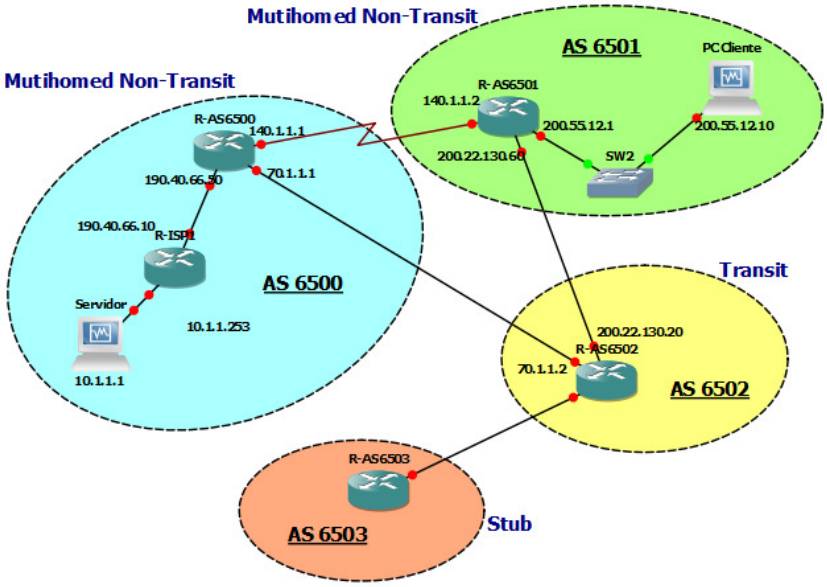


Figura 2.5.2.1: Clasificación de sistemas autónomos

2.5.3 Requerimientos para obtener un bloque de Direcciones IP

Con el fin de poder publicar un servicio en InterNet y que el mismo sea consumido por los usuarios, las organizaciones deben contar con un bloque de direcciones IP asignadas para tal fin. A través de dicho bloque de direcciones, serán identificados los servicios en la red InterNet. Generalmente las organizaciones solicitan a su ISP un subconjunto de direcciones asignadas al mismo durante el período de contrato que establezcan. Mediante dichas direcciones se publica el contenido en la red. El principal problema de este tipo de asignación, es que no permite el manejo de un esquema multiproveedor, ya que el rango de direcciones IP asignado temporalmente a la organización solo será ruteado globalmente mediante BGP a través de la infraestructura de comunicaciones de su ISP. Si la organización quisiera contratar el servicio de conectividad a un segundo proveedor, no podrá rutear el rango de direcciones que le hubiera asignado el primero, a través del nuevo ISP. Con el fin de subsanar este inconveniente, y en el caso de que la organización quisiera contar con un esquema multiproveedor, deberá solicitar y registrar un bloque de direcciones IPv4 propio, mediante la entidad de registro que lo alcance localmente de acuerdo a la distribución internacional establecida por IANA.

El registro de direcciones IPv4 para la región de América Latina y Caribe está a cargo de LACNIC[7]. Cuando una organización desea obtener un bloque de direcciones IPv4, debe solicitarlo a dicha organización, cumpliendo como mínimo con los siguientes requisitos:

- Proveer información detallada mostrando como el bloque solicitado será utilizado dentro de tres, seis y doce meses.
- Entregar "planes de subneteo" por al menos un año, incluyendo máscaras de subred y números de hosts sobre cada subred. El uso de VLSM[8] es requerido.
- Entregar una descripción detallada de la topología de la red de la organización.
- Realizar una descripción detallada de los planes de ruteo de la red, incluyendo los protocolos de ruteo a ser usados, así como también cualquier limitación existente.
- En caso que el solicitante aún no cuente con un bloque IPv6[12] asignado por LACNIC, solicitarlo para cumplir con la política aplicable.

Adicionalmente, la organización deberá demostrar que tiene planes de proveer servicios bajo un esquema multiproveedor, que utilizará y ocupará al menos el 50% de las direcciones IP asignadas en el plazo máximo de 1 año, y pagar el costo de registración inicial y renovación anual. A modo de referencia, de acuerdo al cuadro tarifario de LACNIC publicado en el siguiente sitio: http://www.lacnic.net/web/lacnic/tabla-de-precios, el costo a Noviembre/2015 asciende a U$D2500 (dos mil quinientos dólares) y U$D600 (seiscientos dólares) para registración inicial y renovación respectivamente.

2.5.4 Requerimientos para obtener un número de Sistema Autónomo

Un sistema autónomo está representado por un conjunto de routers que se muestran hacia InterNet bajo una política única de administración. De este concepto se desprende que lo primero que debe cumplimentar una organización para obtener un ASN que permita rutear bajo un esquema multiproveedor su bloque propio de direcciones IP, es una política de ruteo documentada clara y coherente. La política de ruteo definida por la organización debe ser independiente a la de sus proveedores. Junto con la documentación a entregar a la autoridad de registro, se deberá indicar y justificar la diferencia entre la política de ruteo de la organización y la de sus proveedores. Adicionalmente deberá indicarse el protocolo de ruteo externo que se va a utilizar y el conjunto de direcciones IP que serán ruteados a través del número de sistema autónomo que se está solicitando. Toda la información será enviada a la autoridad de registro,

[7] http://www.lacnic.net
[8] Variable Length Subnet Masking

quien la analizará y en caso de aprobarla solicitará el pago del costo de registración inicial [17]. A modo de referencia, de acuerdo al cuadro tarifario de LACNIC, en Noviembre/2015 ese monto asciende a U$D1000 (mil dólares).

2.6 Actualizaciones de rutas y Penalizaciones

Como se comentó en secciones anteriores del presente capítulo, los routers que corren BGP intercambian información entre sí acerca de las redes que poseen interconectadas. En la actualidad, la tabla de ruteo completa en un router BGP puede poseer hasta 500.000 entradas. Resulta obvio que mantener esta información puede ser costoso para cualquier dispositivo, inclusive si el mismo recibe una gran cantidad de actualizaciones. En este último caso podría ocurrir que un router deba avocar sus recursos a la actualización de las tablas de información y no al ruteo de los datos, causando problemas de performance en la red. Es por esta última razón que muchos dispositivos implementan un esquema de penalizaciones para aquellas rutas que produzcan reiteradas modificaciones de la tabla de ruteo de BGP (situación conocida como *flapping*). La técnica consiste en que cuando se detecta una red que produce reiteradas actualizaciones de BGP, la información no se propaga a los routers vecinos y esto último podría ocurrir por un largo período de tiempo. El *Dampening* se define en las políticas de ruteo de cada router BGP.

La técnica conocida como Dampening[18] indica que cada vez que se detecta que una red “flapea” (es decir desaparece y luego vuelve a aparecer en BGP), se le aplicara una penalización incrementando el valor de penalidad en 1000. Cada vez que un router con la funcionalidad de penalización habilitada recibe un mensaje del tipo *UPDATE* de un vecino, incrementa el valor y verifica si dicho valor no ha sobrepasado el máximo permitido (valor de supresión). Si el valor se encuentra por encima del valor de supresión, entonces la ruta no será publicada por un período de tiempo establecido. Si el valor es inferior, entonces se propagará el cambio. El valor de penalización es decrementado a instantes regulares con el fin de mantenerlo actualizado. Si una ruta es penalizada, entonces no será publicada hasta que el valor de penalización alcance un nuevo umbral mínimo conocido como límite de reúso. Cuando el valor de penalización alcance este último, el mensaje *UPDATE* recibido por parte del vecino será procesado y la información de ruteo, agregada a la tabla BGP. La figura **2.6.1** muestra el comportamiento de las penalizaciones en un sistema BGP:

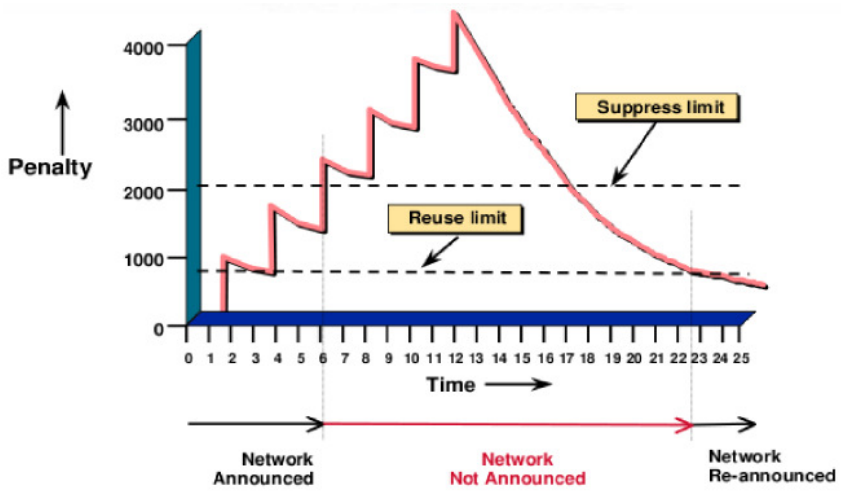


Figura 2.6.1: Penalizaciones en BGP

***Fuente**: Curso de Ruteo avanzado dictado en Enero de 2014 por la Secretaría de Postgrado en la Facultad de Informática*

2.7 Conclusiones finales del capítulo

Durante el desarrollo del presente capítulo se ha analizado la evolución del protocolo BGP y cuestiones técnicas del mismo. De dicho análisis se desprende que BGP es un protocolo de ruteo externo lo suficientemente maduro y estable para permitir los niveles de confiabilidad y disponibilidad que la red InterNet requiere. Asimismo demuestra que mediante sus atributos y la utilización de políticas, provee a los administradores de una granularidad fina para la toma de decisiones. BGP se adapta a los cambios de la red teniendo en cuenta tanto razones físicas, como la caída de un vínculo, como económicas, al tomar decisiones de conmutación y convergencia de acuerdo a las definidas por las políticas configuradas.

Los requerimientos de hardware, así como también los que hay que cumplimentar ante las autoridades de registro para la obtención de un bloque de direcciones IP y número de sistema autónomo; hacen que la utilización directa de BGP por parte de pequeñas organizaciones resulte compleja. La necesidad de contar con equipamiento con gran capacidad de memoria para almacenar las tablas de ruteo, así como también personal especializado en el protocolo, genera altos costos operativos a las organizaciones. Los requisitos que se deben cumplir ante las autoridades de registro para ser elegibles para la obtención de un bloque de direcciones IP y un número de sistema autónomo requieren una importante infraestructura de comunicaciones, la cual debe ser acompañada por grandes inversiones.

Si bien BGP es un protocolo maduro y confiable, su uso puede resultar prohibitivo para pequeñas organizaciones debido a los costos y requerimientos de infraestructura. Por otro lado, es aconsejable su uso por parte de los ISP, debido a su gran despliegue y adopción en InterNet.

Capítulo 3

DNS

3.1 Introducción

El sistema de nombres de dominio (Domain Name System, DNS), es un sistema de nomenclatura jerárquica estandarizado en la RFC 1034[21] y 1035[22] que permite asociar información con nombres de dominio. Su uso principal permite asociar nombres a direcciones IP con el fin de que los mismos puedan ser localizados más fácilmente usando un lenguaje más elegible para los humanos. Debido a que las direcciones IP son difíciles de memorizar, esta técnica permite asignar a las direcciones nombres significativos. Por ejemplo para poder acceder al sitio de la Universidad Nacional de La Plata, debería conocerse la dirección IP que aloja dicho servicio, que en este caso es 163.10.0.145. El sistema DNS asigna el nombre *www.unlp.edu.ar* a dicho sitio y permite que se pueda realizar de modo sencillo la traducción de dicho nombre a la dirección IP correspondiente. Del mismo modo ocurre con cualquier otro sitio o recurso.

En los comienzos de InterNet se utilizaba una única tabla centralizada de traducción de nombres a direcciones. En el año 1970 ARPANET estaba formada por unos cientos de máquinas y un único archivo, *HOSTS.TXT*, que contenía toda la información que se necesitaba sobre esas máquinas. El centro de información de red del Departamento de Defensa de Estados Unidos, disponía de la versión maestra de la tabla y otros sistemas realizaban una copia regularmente. Con el paso del tiempo y a medida que los protocolos TCP/IP se utilizaban más asiduamente, este método presento serios inconvenientes entre los que pueden destacarse los siguientes:

- El tráfico y la carga de red para la máquina que contenía la tabla que hacía posible el mapeo era excesivo.
- La consistencia del archivo era muy difícil de mantener.
- No se podía garantizar la no duplicidad de nombres, dado que mantener una administración central en una red Internacional era algo muy complicado.
- A medida que la red crecía, el tamaño del archivo también lo hacía.
- Si la máquina central salía de servicio, toda la red quedaba inutilizable.
- El método era claramente poco escalable.

En al año 1984 surgió un nuevo sistema de resolución de nombres llamado *Domain Name System (DNS)*. Las premisas básicas del nuevo sistema fueron solucionar los principales inconvenientes que se venían acarreando con el sistema anterior.

3.2 Características y funcionamiento

DNS es un sistema de nomenclatura jerárquica estandarizado en la RFC 1034 y 1035 que permite asociar información con nombres de dominio. Su uso principal permite asociar nombres a direcciones IP con el fin de que los mismos puedan ser localizados más fácilmente usando un lenguaje más elegible para los

humanos. Para dar soporte al servicio, se utiliza una base de datos jerárquica y distribuida que permite almacenar la información de resolución. Asimismo, el servicio permite por su naturaleza descentralizada que cada porción de la base de datos sea administrada por la entidad u organización a la que fue delegada, permitiendo de esta manera que cada administrador solo pueda manipular los datos de su propia base de datos. [23]

Cuando una aplicación cliente necesita resolver un nombre, enviará el requerimiento de resolución a algún servidor de nombres, del cual esperará recibir como resultado la dirección IP asociada a dicho nombre. Una vez obtenida la dirección IP, se procederá a la comunicación estándar que defina cada protocolo dependiendo de cada necesidad. Con el fin de brindar una alta disponibilidad al sistema se definieron 13 servidores DNS raíz, los cuáles se encuentran geográficamente dispersos en el mundo, y permiten que ante una eventual falla en alguno de ellos, otros puedan tomar el control del sistema de resolución. La lista de los 13 servidores raíz para el sistema de nombres, puede verse en la siguiente figura extraída del sitio *www.iana.org*:

List of Root Servers

Hostname	IP Addresses	Manager
a.root-servers.net	198.41.0.4, 2001:503:ba3e::2:30	VeriSign, Inc.
b.root-servers.net	192.228.79.201, 2001:500:84::b	University of Southern California (ISI)
c.root-servers.net	192.33.4.12, 2001:500:2::c	Cogent Communications
d.root-servers.net	199.7.91.13, 2001:500:2d::d	University of Maryland
e.root-servers.net	192.203.230.10	NASA (Ames Research Center)
f.root-servers.net	192.5.5.241, 2001:500:2f::f	Internet Systems Consortium, Inc.
g.root-servers.net	192.112.36.4	US Department of Defence (NIC)
h.root-servers.net	128.63.2.53, 2001:500:1::803f:235	US Army (Research Lab)
i.root-servers.net	192.36.148.17, 2001:7fe::53	Netnod
j.root-servers.net	192.58.128.30, 2001:503:c27::2:30	VeriSign, Inc.
k.root-servers.net	193.0.14.129, 2001:7fd::1	RIPE NCC
l.root-servers.net	199.7.83.42, 2001:500:3::42	ICANN
m.root-servers.net	202.12.27.33, 2001:dc3::35	WIDE Project

Figura 3.2.1: Servidores DNS raíz

En la lista puede verse el nombre asignado al servidor, cuál es su dirección IP en versión 4 y versión 6 y cuál es la organización que administra los recursos. Cabe destacar que los servidores se encuentran geográficamente distantes, bajo una organización estratégica de modo de brindar alta disponibilidad al servicio.

3.3 Espacio de nombres

La base de datos del sistema de nombres DNS se indexa utilizando nombres de dominios. Cada nombre de dominio está representado por un camino en un árbol invertido de nombres, al cual se lo conoce como espacio de nombres. Se denomina árbol invertido, ya que los nombres dentro del sistema se comienzan a leer desde las hojas (nodos inferiores) hacia la raíz. Dicho árbol posee una estructura jerárquica similar a la que se puede encontrar en un sistema de archivos de un sistema operativo. El árbol posee una única raíz denotada por el carácter "." (Punto). A partir de dicha raíz, la base de datos se estructura

jerárquicamente como se muestra en la figura **3.3.1**. Como puede apreciarse, el primer nodo del árbol se corresponde con el nodo raíz. Es de allí desde donde parte la estructura de resolución. Los servidores responsables de la administración de dicha porción de la base de datos son los 13 servidores raíz, quienes a su vez delegan porciones de la base de datos a otros nodos:

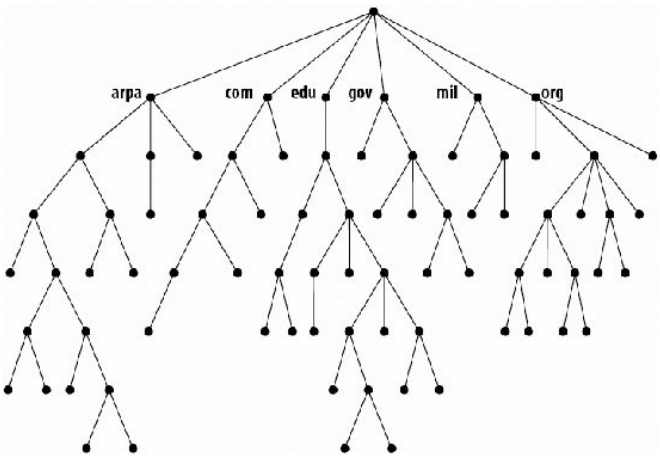


Figura 3.3.1: Estructura del sistema de nombres DNS en forma de árbol

Cada nodo en el árbol de la estructura posee una etiqueta que puede tener una longitud máxima de 63 caracteres. Los nodos intermedios generalmente denotan un nombre de dominio, mientras que los nodos inferiores (las hojas) hacen referencia a los recursos de resolución dentro del dominio. Un nombre completo de dominio está representado por el conjunto de etiquetas que van desde las hojas del árbol hasta la raíz del mismo, delimitadas por el carácter "." (Punto). Los nodos que dependen directamente de un padre (denominados también hermanos) no pueden contener en su etiqueta el mismo nombre. De este modo los caminos son unívocos. Si existe independencia entre los caminos, los nombres pueden repetirse. Tal es el caso de la etiqueta www, que generalmente se repite como nombre de nodo en las hojas, pero sobre caminos distintos del árbol en la jerarquía de nombres. La figura **3.3.2** grafica dicha situación. En la misma puede verse un caso de repetición de nombres sobre el mismo camino, lo cual no está permitido; y un caso de repetición del mismo nombre sobre un camino distinto. Este último solapamiento de nombres está permitido dado que se cumple la independencia de caminos:

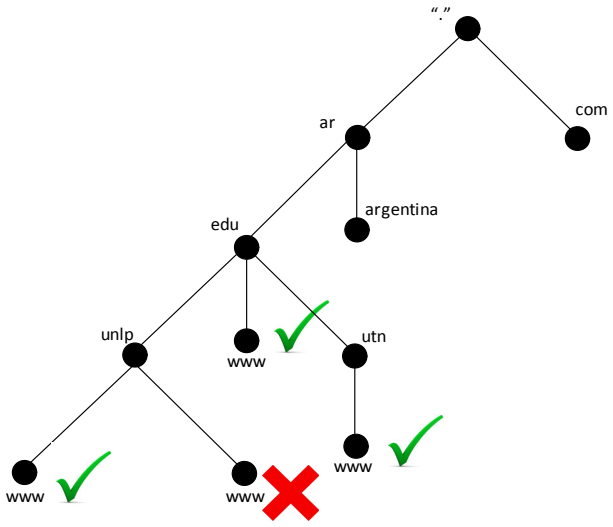


Figura 3.3.2: Inconsistencia de nombres en el sistema DNS

Los nombres de dominio, incluyendo los nombres de servidores, se estructuran siguiendo una organización lógica a diferencia de la organización del direccionamiento IP que generalmente sigue una organización física o geográfica. Este tipo de organización permite que dos hosts dependientes del mismo dominio (por ejemplo www1 y www2 dentro del dominio ejemplo.com.ar) puedan residir en redes distintas, inclusive geográficamente distantes una de otra. Adicionalmente, de cada dominio se pueden desprender otros dominios, conocidos como sub dominios. Esta estructuración define la forma de árbol del sistema de nombres. Una forma simple de determinar si un dominio forma parte de otro, es a través de la estructura jerárquica de nombres. Si en el recorrido hacia la raíz se encuentra algún punto en común, entonces se puede inferir que ambos subdominios forman parte de un dominio general. Por ejemplo el dominio ejemplo.com.ar es un subdominio del dominio com.ar.

A cada dominio, también se lo puede conocer como un subdominio de nivel N, donde N determina la distancia medida en cantidad de nodos del árbol que se deben atravesar hasta llegar a la raíz a partir de un nodo dado. Por consiguiente, se determina que la raíz "." (Punto) posee nivel 0. De la raíz se desprenden los dominios de nivel 1, que son un número acotado. Este tipo de dominio generalmente se utiliza para delegar administración de la base de datos de resolución a países, organizaciones, y otros fines que engloben lógicamente nombres relacionados a algún tipo de actividad. Si bien el sistema de nombres no impone reglas acerca del modo en el que deben llamarse los dominios, existen ciertos dominios de nivel 1 que se han tomado como referencia para la construcción del árbol. Algunos de ellos son:

- **com.:** Agrupa organizaciones comerciales como por ejemplo google.com, hotmail.com y otras.

- **edu.:** Agrupa organizaciones educativas como por ejemplo berkeley.edu, purdue.edu y otras.
- **gov.:** Agrupa organizaciones gubernamentales como por ejemplo nasa.gov y nsf.gov entre otras.
- **net.:** Originalmente agrupaba organizaciones proveedoras de infraestructura de red, como NSFNET y UUNET. En el año 1996, se modificó el criterio para permitir que organizaciones comerciales también pudieran registrar subdominios de .net, de modo similar a lo que se hace con .com.

Otros dominios de nivel 1 son **ar**, **pe**, **uy**, entre otros; que agrupan a las organizaciones de cada país. Este tipo de dominios son delegados a las autoridades de registro de Argentina, Perú o Uruguay por ejemplo, de modo tal que cada una agrupe los nombres de las solicitudes generadas ante cada entidad registrante. En Argentina, la entidad responsable de administrar el subdominio ar. es NIC Argentina, perteneciente a la Dirección Nacional de Registros de Dominios de Internet, quien a su vez subdivide la base de datos de resolución en otros dominios. Dentro de los subdominios más comunes que se pueden registrar en Argentina y a través del sitio se encuentran [24]:

- **com.ar.:** Este subdominio agrupa organizaciones comerciales o de índole genérico. De acuerdo a la normativa vigente, podrá registrar dominios en la zona .com.ar cualquier persona física o jurídica argentina o extranjera.
- **gob.ar.:** Subdominio que agrupa a organizaciones gubernamentales
- **mil.ar.:** Sólo podrán registrar dominios en la zona .mil.ar entidades pertenecientes a las Fuerzas Armadas de la República Argentina.
- **net.ar.:** Sólo podrán registrar dominios en la zona .net.ar las entidades argentinas o extranjeras que sean proveedoras de servicios de Internet y tengan licencia de la Comisión Nacional de Comunicaciones para prestar servicios de valor agregado en la República Argentina.
- **org.ar.:** Sólo podrán registrar dominios en la zona .org.ar las entidades que sean organizaciones sin fines de lucro argentinas o extranjeras
- **tur.ar.:** Sólo podrán registrar dominios en la zona .tur.ar las empresas de viajes y turismo, agencias de turismo o agencias de pasajes que se encuentren habilitadas por el Ministerio de Turismo.
- **edu.ar.:** Este subdominio agrupa organizaciones educativas como por ejemplo unlp.edu.ar. Para el caso especial del subdominio edu.ar, la ARIU (Asociación de Redes de Interconexión Universitaria) es la entidad en la que NIC Argentina delegó la responsabilidad de la operación estable y confiable de la base de datos.

Cada entidad de registro define sus reglas y aranceles. A modo de referencia, en Noviembre 2015, el arancel para el registro de dominios pertenecientes a NIC Argentina depende del tipo de dominio que se quiera registrar y va desde los $65 a $450 (pesos argentinos) anuales El reglamento para el registro de dominios de InterNet dependientes del subdominio ar. puede encontrarse en el sitio principal de NIC Argentina en http://nic.ar.

Existe también un dominio de nivel superior o de nivel 1 especial, denominado arpa. El mismo fue originalmente utilizado para realizar la transición de la tabla HOSTS.TXT utilizada en la red ARPAnet hacia el sistema de nombres DNS. De allí su nombre. Es utilizado con el fin de realizar resoluciones inversas, es decir, determinar que nombre tiene asociado una determinada dirección IP. Entonces por ejemplo la dirección IP 163.10.0.145 es mapeada al nombre 145.0.10.163.in-addr.arpa, el cual a su vez podría resolverse a un nombre como por ejemplo *www.unlp.edu.ar*.

La estructuración de nombres, pensando en una agrupación lógica y utilizando como criterios ubicaciones geográficas o tipos de organización permite inferir los nombres en base a su estructuración. Este esquema permite que dado un nombre de dominio, se pueda inferir a qué tipo de equipo o servicio hace referencia el mismo. Por ejemplo, sobre el nombre de dominio *www.unlp.edu.ar*, se puede inferir de modo sencillo que se trata de un nombre que hace referencia a un sitio o servicio que tiene referencia con el país Argentina dado que se encuentra en el subdominio de nivel superior **ar**; que referencia a dicho país. Luego podemos determinar que se trata de una organización educativa, a través del subdominio **.edu**. Si bien el nombre **unlp** no responde a ninguna de las reglas impuestas, se puede inferir que son las siglas de la Universidad Nacional de La Plata; y por último la sigla **www** que generalmente se encuentra asociada al servicio World Wide Web. Utilizando el mismo esquema de razonamiento, se puede inferir la índole de casi cualquier nombre de DNS.

3.4 Delegación de zonas

Una de las premisas principales en la implementación del servicio DNS fue que el mismo pudiera operar de forma descentralizada, de modo tal que los distintos dominios puedan ser administrados independientemente e inclusive puedan ser servidos desde distintos puntos de la red. Este principio se logra a través del concepto denominado delegación. Cada dominio de DNS es subdividido en otros, a los cuáles se los conoce como subdominio. De este modo, el dominio raíz "." (Punto), es subdividido en los dominios de nivel superior o nivel 1 (com, edu, ar entre otros). A estos últimos los llamamos subdominios de la raíz ".", los cuáles a su vez se subdividen en otros como por ejemplo edu.ar y com.ar. Cada uno de estos dominios posee una administración descentralizada y permite que sea administrado por diferentes organizaciones y

alojado en distintos servidores. Cada organización es responsable por la administración de la porción de la base de datos (subdominio) que le fue delegada. La responsabilidad en la administración incluye la definición de registros dentro de la zona, o bien la delegación de nuevos subdominios a otras organizaciones o administradores. Por ejemplo, los administradores del dominio edu.ar delegan la administración del subdominio unlp.edu.ar a la Universidad Nacional de La Plata, quien define sus propios registros de resolución como por ejemplo *www.unlp.edu.ar* y también delega porciones de la base de datos a las distintas Unidades Académicas como por ejemplo la Facultad de Informática que es la responsable de responder por el subdominio info.unlp.edu.ar. El esquema de delegación puede apreciarse de modo gráfico en la siguiente figura:

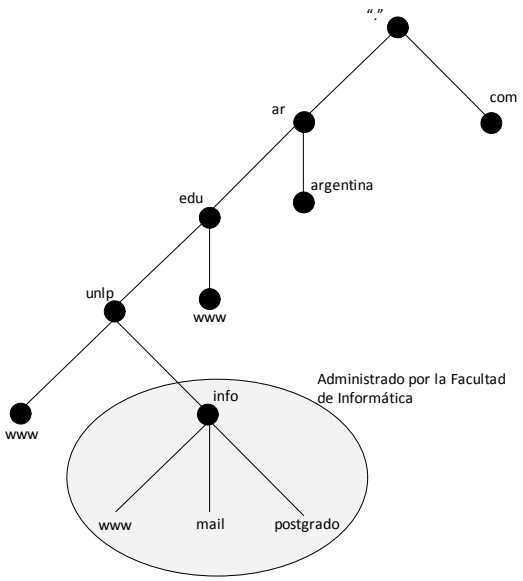


Figura 3.4.1: Delegación de administración en una estructura de DNS

3.5 Servidores DNS y zonas

Toda la información acerca de zonas, nombres de dominio y registros de resolución, es gestionada por los servidores de nombre de dominio. Cada servidor de nombres contiene en la mayoría de los casos una porción de la base de datos de resolución conocida como zona de DNS, por la cual el mismo es responsable. Se dice que un servidor es autoritativo para determinada zona, si el mismo gestiona la porción de la base de datos que se corresponde con el dominio representado por la zona. Un servidor DNS puede ser autoritativo para una o más zonas. Adicionalmente posee información que lo enlaza a otros servidores de resolución, lo cual le permite resolver otros nombres para los cuáles el mismo no es autoritativo o no posee información en su propia base de datos. Cada dominio de DNS puede ser subdividido en diversas zonas.

Cada una de las zonas se corresponde con un subdominio del dominio anterior. Adicionalmente, las zonas pueden ser gestionadas por el mismo servidor DNS, el cuál será autoritativo para las mismas, o bien pueden ser delegadas a nuevos servidores DNS, los cuales administrarán independientemente la porción de la base de resolución delegada y serán autoritativos para la resolución de nombres de dicha porción. La figura 3.5.1 muestra un ejemplo acerca del esquema de dominios y delegación de zonas:

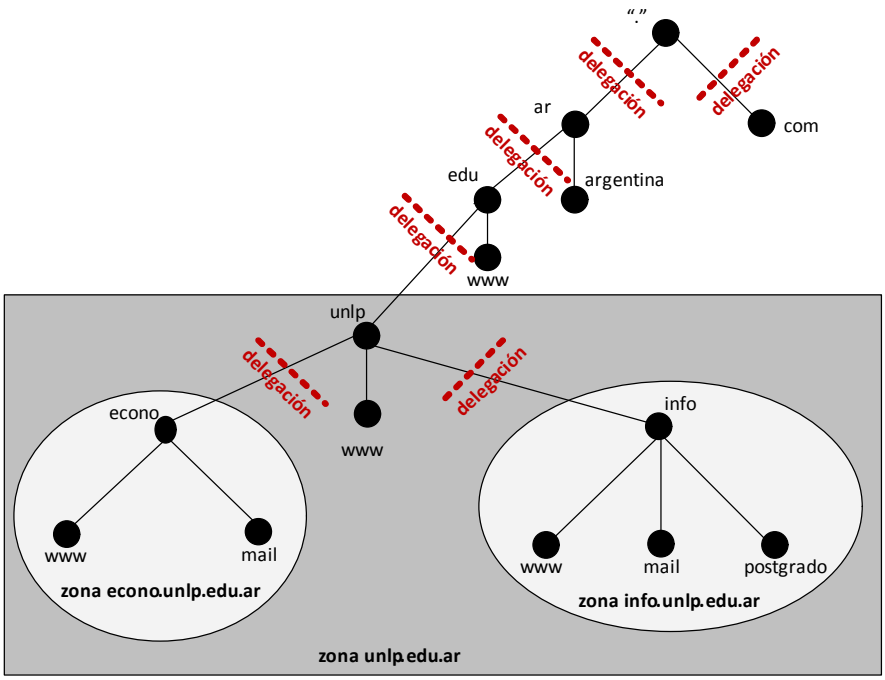


Figura 3.5.1: Esquema de dominios de DNS y delegación de zonas

La especificación de DNS, establece dos tipos de servidores de nombres: los servidores primarios y los secundarios. Un servidor DNS primario, posee la información de resolución de una determinada zona, ya que el administrador de dicho servicio ha ingresado la configuración manualmente en el mismo. Es el servidor donde se realizan los cambios para los distintos registros de resolución. El servidor secundario aprende la información de resolución a través del servidor primario. En este último, el administrador no actualiza información de resolución, sino que la configuración es aprendida del primario a través del concepto conocido como transferencia de zona. Cada vez que en el servidor primario se realice alguna modificación en la zona, la misma será notificada al servidor secundario para que el mismo actualice la información de resolución transfiriendo la porción de la base de datos que ha cambiado desde el servidor primario. Adicionalmente, el servidor secundario consultará a intervalos regulares al servidor primario con el fin de determinar si hubo algún cambio en la zona para el cual no haya sido notificado. De producirse esta situación, el servidor secundario transferirá desde el primario la zona que haya sufrido modificaciones. Pueden existir tantos servidores secundarios como se quiera configurar. El único requerimiento entre ellos

es que exista comunicación a través del protocolo UDP y TCP en el puerto 53 de modo tal que puedan transferir información de actualización de zonas entre ellos. La posibilidad de agregar nuevos servidores secundarios y sincronizar la información entre ellos provee alta disponibilidad. Ante la caída de un servidor los otros responderán a los requerimientos. También provee escalabilidad ya que permite repartir la carga de resolución entre varios equipos [25]. El concepto "servidor primario" o "servidor secundario" puede resultar confuso, ya que no se aplica a la totalidad del servidor DNS. Un servidor puede ser primario para un conjunto de zonas y secundario para otras, de modo tal que el concepto no aplica a la totalidad del servicio que corre sobre cierto equipo sino que aplica a cada zona configurada. Tanto los servidores primarios como los secundarios son autoritativos para el conjunto de zonas que tengan configuradas.

La información de resolución dentro de un servidor, puede ser almacenada utilizando distintas técnicas, dependiendo del software que se utilice. La empresa Microsoft provee software para dar soporte al servicio de DNS, la cual almacena las configuraciones en el registro de Windows. Otra implementación del protocolo DNS es la provista por el software BIND [26], que es en la cual se centrará el presente trabajo. En BIND, la información es almacenada en archivos dentro del sistema de archivos que corre el servicio. Cada archivo representa una zona de resolución, y los mismos tienen un esquema similar al que se muestra en el siguiente cuadro:

```
ejemplo.com.ar.          IN SOA  dns1.ejemplo.com.ar. root.ejemplo.com.ar. (

199609260  ; serial

28800      ; refresh (8 hours)

7200       ; retry (2 hours)

2419200    ; expire (4 weeks)

86400      ; minimum (1 day)

)
```

Cuadro 3.5.1: Configuración de zona de DNS en servidor BIND

La primera línea del cuadro indica que el servidor de nombres es el mejor origen de información para la zona ejemplo.com.ar. Esto significa que el servidor en cuestión será autoritativo para esta zona. La primera columna de la línea indica cual es el nombre completo del dominio, finalizando con el carácter "." (Punto) que indica su dependencia de la raíz del árbol. La segunda columna contiene la palabra reservada IN, que hace referencia a que el dominio es un dominio de la red **IN**ternet. La tercera columna hace referencia al tipo de registro, que en este caso es SOA (Start Of Authority) y es precedida por el nombre del servidor DNS autoritativo para la zona, que como se puede observar, en este caso, se llama dns1.ejemplo.com.ar. Por último, el campo cuyo contenido es root.ejemplo.com.ar, es traducido a la dirección de correo electrónico

root@ejemplo.com.ar, que es la dirección de mail a la que se debe escribir ante la necesidad de contactar al administrador de la zona de DNS. Seguido a esta primera línea se encuentran 5 (cinco) valores cuyo significado es [27]:

- **serial:** Corresponde al número de serie de la zona, útil para determinar actualizaciones de la información contenida en la misma. Generalmente se utiliza como formato, la convención AAAAMMDDNN, donde AAAA es el año, MM representa al mes, DD representa al día y NN representa a un número autoincremental de 00 a 99. De este modo se representan los números de serie y en el mismo se incluye la fecha en la que fue modificado. Si bien esto es una convención, permite que cada modificación del archivo incluya un número de serie mayor, que es lo que realmente cuenta al momento de validar dicho número. Cuando un servidor secundario se pone en contacto con un primario, verifica el número de serie de la zona que desea transferir. Si el servidor primario posee un archivo con número de serie superior, entonces se realizará la transferencia de la zona. Caso contrario, el servidor secundario mantendrá su archivo de configuración, dado que considerará que posee la versión más reciente de la porción de la base de datos.
- **refresh:** El valor indicado en la tercera línea del cuadro, determina el intervalo de tiempo que debe transcurrir desde que el servidor secundario realizó una transferencia de zona desde el primario, hasta que vuelva a verificar si la misma se encuentra actualizada. Si bien cada vez que se realiza una modificación de zona, el servidor primario informa a los servidores secundarios acerca del cambio; estos últimos chequean a intervalos regulares al primario por posibles cambios no notificados. Este valor define el intervalo de tiempo expresado en segundos.
- **retry:** Si el servidor secundario no pudo contactar al primario luego del tiempo estipulado en el campo refresh, comenzará intentos de reconexión cada cierta cantidad de tiempo. Este tiempo es definido por el campo retry y se encuentra expresado en segundos. Generalmente se busca que este valor sea inferior al del campo refresh.
- **expire:** Si el servidor secundario no puede contactar al servidor primario por un período mayor de tiempo al estipulado en este campo, el secundario expirará la zona de DNS. Esto significa que el mismo dejará de responder consultas para el dominio dado. El servidor secundario determina que el dato contenido en su propia copia de la zona es demasiado antiguo y decide que es mejor dejar de contestar por la zona, que contestar con datos que podrían estar desactualizados. Este valor generalmente se define alto con el fin de evitar que un servidor secundario salga de servicio en un corte de red prolongado que no le permita actualizarse con el servidor primario.

- **mínimum:** Este valor, también conocido como Negative Caching TTL, define el valor por defecto que otros servidores DNS utilizarán para almacenar los registros de la zona en sus propias caches. TTL hace referencia a Time To Live y define el tiempo por el cuál un determinado recurso es considerado válido. Este valor es de suma importancia, ya que le indica a las caches intermedias de resolución por cuánto tiempo deberán almacenar los registros. Si el dominio posee muchos cambios, es aconsejable que este valor se mantenga bajo. Este valor aplica a todos los registros contenidos en la zona. Adicionalmente, cada registro puede poseer su propio valor que determine el tiempo de vida y sobrescriba el valor global configurado.

Los datos mencionados, definen lo que se conoce como encabezado de un archivo de zona de DNS. En la siguiente sección se analizará la información contenida en la zona en forma de registros y los distintos tipos que se pueden encontrar.

3.6 Tipos de registros

Cada línea en un archivo de zona de un servidor DNS, define lo que se conoce como un Resource Record (RR o registro de recurso). Algunas líneas como se analizó la sección anterior definen parámetros de configuración global acerca del comportamiento de la zona. Otros, definen registros que permiten crear asociaciones entre un nombre y un valor. Los nombres que se definen en un archivo de zona, no hacen distinción entre letras mayúsculas y minúsculas, por lo tanto los nombres se pueden definir a gusto del administrador. Generalmente por convención se utilizan todos los caracteres en minúsculas. Si bien en la RFC de DNS los registros son presentados utilizando un orden específico, no es requerimiento de que así lo sea. Los tipos de registro pueden estar mezclados en el archivo de la zona, aunque generalmente se definen convenciones como por ejemplo agrupar todos los tipos de registro y dar un orden alfabético por grupo. Cada tipo de registro tiene un propósito y un uso particular. Un ejemplo genérico de un archivo de zona completo con algunos registros de recurso puede observarse en el cuadro **3.6.1**:

```
ejemplo.com.ar.          IN SOA  dns1.ejemplo.com.ar. root.ejemplo.com.ar. (

199609260  ; serial

28800      ; refresh (8 hours)

7200       ; retry (2 hours)

2419200    ; expire (4 weeks)

86400      ; mínimum (1 day)

)

IN       NS        dns1

@                              IN       MX        5 mail

dns1                           IN       A         192.168.1.1

mail                           IN       A         192.168.1.2

www                            IN       CNAME     mail
```

Cuadro 3.6.1: Ejemplo de definición de registros en zona de DNS

Los tipos más comunes de registro que pueden definirse son:

- **SOA (Start Of Authority o Autoridad de la zona):** Proporciona información sobre el servidor DNS primario de la zona, el correo electrónico del administrador del dominio, el número de serie del dominio, y los tiempos de refresco o actualización. La información de este registro ha sido descripta en la sección anterior.
- **NS (Name Server o Servidor de Nombres):** Define la asociación que existe entre un nombre de dominio y los servidores de nombres que almacenan la información de dicho dominio. Cada dominio se puede asociar a uno o más servidores de nombres. Indica cuál o cuáles son los servidores de nombre autoritativos para la zona.
- **MX (Mail Exchange o registro de intercambio de correo):** Asocia un nombre de dominio a una lista de servidores de intercambio de correo para ese dominio. Permite realizar un balanceo de carga y definir prioridades para el uso de uno o más servicios de correo.
- **A (Address o dirección):** Este registro se usa para traducir nombres de servidores a direcciones IPv4.
- **CNAME (Canonical Name o Nombre canónico):** Se usa para crear nombres de servidores o alias para los servidores de un dominio. Es usado cuando se están corriendo múltiples servicios (como por ejemplo mail y web) en un servidor con una sola dirección IP. Cada

servicio tiene su propia entrada de DNS (como mail.ejemplo.com.ar y www.ejemplo.com.ar).

- **PTR (Pointer o indicador):** También conocido como registro reverso, funciona a la inversa del registro A, traduciendo direcciones IP en nombres de dominio

Existen otros tipos de registro definidos en la RFC, que permiten ampliar el esquema de resolución de nombres como por ejemplo **SPF** para indicar los hosts autorizados para enviar correo para un dominio determinado, o **TXT** que se utiliza para autenticación de correo. Adicionalmente se definen registros para el protocolo IPV6 como por ejemplo **AAAA** que tiene la misma función que el registro A de IPv4.

3.7 Proceso de resolución

Cuando un servidor DNS recibe un requerimiento de resolución, verifica su propia base de datos con el fin de analizar si puede satisfacer el requerimiento con alguna de las zonas que en él se encuentran definidas. Si el requerimiento de resolución es sobre un registro de una zona para la cual el servidor no es autoritativo, entonces comenzará el proceso de resolución. Para ello, el servidor DNS realizará la consulta en nombre del cliente, hasta obtener el dato solicitado o bien un mensaje de error indicando que el nombre no puede ser resuelto. El servidor DNS realizará tantas consultas como sean necesarias a la base de datos jerárquica, con el fin de alcanzar la porción de la base que pueda devolver la solicitud completa. Suponiendo el nombre **www.ejemplo.com.ar**, la primera acción que se realizará será consultar al servidor raíz **.** (Punto), la dirección del servidor responsable de responder por el subdominio **ar**. Una vez obtenida dicha dirección, se consultará la dirección del servidor responsable por responder el subdominio **com.ar**. Por último se consultará la dirección del servidor responsable del subdominio **ejemplo.com.ar**, al cual se le realizará una última consulta por el registro **www**. Este último servidor, conocido como autoritativo para el dominio, responderá al primer servidor DNS, quién se encargará de entregar la respuesta final al cliente. La respuesta obtenida, estará compuesta por el nombre resuelto, la dirección IP asociada a dicho nombre y el tiempo por el cual dicho nombre será válido previo a que deba realizarse una nueva consulta para validarlo. La figura **3.7.1** muestra el proceso de resolución descripto:

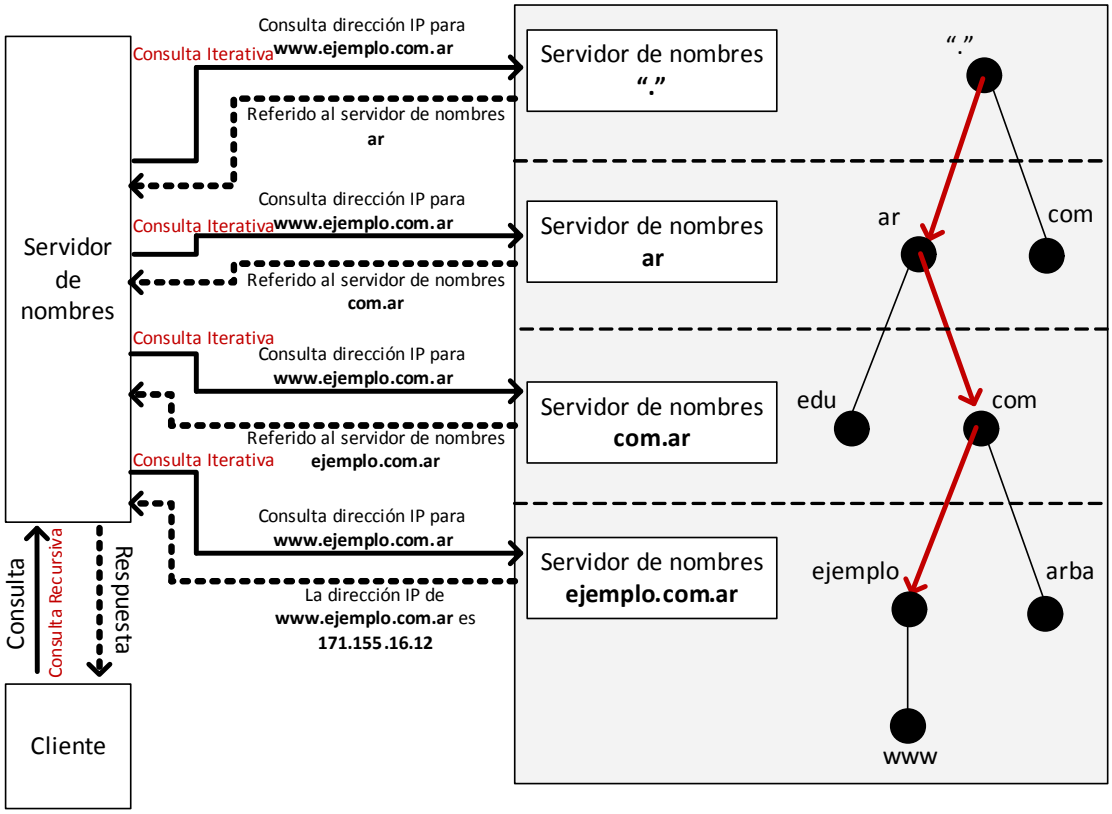


Figura 3.7.1: Proceso de resolución de un nombre de DNS

Las consultas que un cliente realiza a un servidor DNS pueden ser recursivas o iterativas. Las consultas recursivas, o el proceso de resolución recursivo, es el nombre que se le da al proceso que realiza un servidor DNS cuando recibe un pedido de resolución. Se conoce como recursiva, ya que el servidor DNS realizará el mismo procedimiento, consultando a otros servidores DNS, hasta que el mismo obtenga la respuesta deseada y pueda ser pasada al cliente. El resultado de una consulta recursiva, es siempre el resultado final esperado (la asociación de la dirección IP al nombre), o bien un mensaje de error indicando que el registro consultado no existe. La resolución iterativa, sin embargo, hace referencia al proceso de resolución utilizado por el servidor de nombres cuando recibe una consulta iterativa. Cuando un servidor recibe una consulta iterativa, el mismo no está obligado a dar la respuesta final, sino que lo que puede hacer es referir al cliente a otro servidor DNS con el fin de que se realice una nueva consulta y determinar si el segundo puede dar la respuesta buscada. Si el segundo servidor referido no puede dar la respuesta buscada, entonces devolverá el nombre de otro servidor para que el proceso de resolución continúe, hasta encontrar a algún servidor autoritativo que pueda dar una respuesta final por el registro solicitado. La figura 3.7.1 muestra el proceso de resolución completo, diferenciando las consultas iterativas de las recursivas para la resolución del nombre *www.ejemplo.com.ar*. [28]

3.8 Caching

El proceso de resolución de nombres es un proceso complejo que en la mayoría de los casos implica realizar numerosas consultas a diversos servidores, con el fin de servir la consulta de un cliente. Cada una de las consultas intermedias, agrega retardos medidos en tiempo a la respuesta final. De este modo, si la consulta que realiza un cliente a su servidor DNS requiere tres consultas adicionales a otros servidores con el fin de resolver el nombre solicitado, el tiempo total de respuesta estará dado por la sumatoria de los tiempos de respuesta de los 3 servidores consultados. La técnica conocida como caching tiene como finalidad disminuir los tiempos anteriores a través del almacenamiento temporal de las consultas ya realizadas. De este modo si dos clientes consultan el mismo nombre a un servidor DNS en un lapso de tiempo acotado, el primer requerimiento de resolución deberá esperar hasta que se complete la totalidad del proceso. El segundo requerimiento será servido directamente por el servidor DNS sin necesidad de realizar nuevamente el procedimiento de búsqueda, ya que el mismo ha almacenado temporalmente en su base de datos (cache) el resultado del requerimiento previo.

Cada vez que un servidor DNS recibe un requerimiento de resolución recursiva, el mismo debe realizar diversas consultas a otros servidores con el fin de buscar una respuesta a la consulta original. Cada consulta que realiza, le permite descubrir información acerca del espacio de nombres en el árbol de resolución. Cada vez que el servidor es referido a otro servidor DNS, el mismo almacena la información acerca de los servidores autoritativos para cierta porción del espacio de nombres y sus correspondientes direcciones IP. La información almacenada le permitirá acelerar las búsquedas a futuro para nuevos requerimientos de resolución. El tiempo de validez que tendrá un recuso resuelto, dependerá del valor establecido en el campo TTL para la respuesta. Una vez expirado el tiempo de vida, el servidor deberá repetir el proceso ante un nuevo requerimiento de resolución. También son aprendidos y almacenados temporalmente por el servidor DNS, los requerimientos de resolución cuyo resultado han arrojado un error, ya sea porque el servidor DNS referido no se encuentra disponible o porque el registro solicitado no existe. A este último concepto se lo conoce como negative caching, y permite almacenar por lapsos de tiempo acotados este tipo de resoluciones con el fin de brindar más rápidamente los resultados.

El almacenamiento de las consultas realizadas por parte del servidor, implica que cada vez que se desee resolver un nombre, el servidor DNS buscará en su propia base de datos de resolución temporal (caché) si posee el resultado a la próxima consulta a realizar. Si el resultado se encuentra en la caché y es válido, entonces será servido inmediatamente, caso contrario el servidor realizará la consulta de modo habitual. Esta técnica principalmente utilizada por los servidores DNS, ha sido extendida a los sistemas operativos y aplicaciones. Los sistemas operativos de los clientes que realizan consultas también almacenan

las resoluciones realizadas, así como también lo hacen las aplicaciones. El objetivo es brindar una mejor experiencia al usuario disminuyendo los tiempos de respuesta en las consultas realizadas.

Las caché deben poseer un mecanismo de expiración que permita determinar cuándo un dato es válido o no. Este mecanismo permite mantener los datos temporales actualizados con el fin de mantener la coherencia en las respuestas. Para ello, se utiliza el valor TTL (Time To Live) de las respuestas, para almacenar los resultados temporalmente por el lapso de tiempo indicado en este valor. Cuando el administrador de una determinada zona configura el valor TTL para la misma o un registro particular, indica a otros servidores DNS cuál es el lapso de tiempo que los mismos tienen autorizado a almacenar temporalmente los recursos resueltos. Luego de este tiempo los datos almacenados ya no tendrán validez, debiendo descartarse y resolverse nuevamente. Un valor de TTL demasiado alto, causará que los recursos sean almacenados en caches intermedias por largos lapsos de tiempo, lo cual puede causar que una actualización de datos no se vea reflejada inmediatamente para los clientes. Adicionalmente, el hecho de permitir que los datos se cacheen temporalmente en otros servidores, causará que el servidor DNS autoritativo reciba una menor cantidad de consultas para determinada zona. En contrapartida, un valor de TTL bajo causará que los recursos se almacenen por cortos lapsos de tiempo asegurando un alto grado de actualización en las respuestas; pero causará que el servidor autoritativo tenga que contestar una mayor cantidad de consultas.

Si bien se determina que los registros resueltos deben ser almacenados o cacheados únicamente por el lapso de tiempo indicado en el campo TTL de la respuesta, algunos sistemas operativos e inclusive aplicaciones no respetan esta imposición. Esto se debe a que el proceso de resolución de nombres implica un retardo al momento de establecer una conexión. Previo a establecer la comunicación, debe realizarse la traducción del nombre dado a una dirección IP. En el ejemplo de acceso a un recurso en un servidor web, cuando el usuario ingresa una dirección en su navegador la primera acción que se realiza es la resolución del nombre solicitado. Esto disparará una serie de consultas a servidores DNS, de modo tal de obtener la dirección IP a la cual deberá realizarse la conexión utilizando el protocolo HTTP. Debido a que las resoluciones implican un retardo en la comunicación, los navegadores web implementan su propia caché de resolución, del mismo modo que lo hace el sistema operativo cliente y los servidores no autoritativos.

El principal problema de la implementación de caches de resolución por parte de las aplicaciones, reside en la necesidad de solicitar la resolución de los registros de DNS a través del sistema operativo en el que corren. Es el sistema operativo quien realizará la resolución en nombre de la aplicación y luego le devolverá el resultado a la misma. Con el fin de proveer un sistema de comunicación estándar entre el sistema operativo y las aplicaciones para la resolución de nombres, se utilizan las funciones estándar **gethostbyname** y **gethostbyaddr**. Las mismas permiten traducir nombres a direcciones IP y direcciones IP a nombres respectivamente. Estas funciones han sido utilizadas por años para realizar el proceso de

resolución. En la actualidad, dichas funciones han sido reemplazadas por las funciones **getnameinfo** y **getaddrinfo** respectivamente. Esto se debe a que las funciones anteriores no hacían un correcto manejo de la resolución sobre la pila de IP versión 6. La principal particularidad de las 4 funciones nombradas, es que si bien se encargan de enmascarar el proceso de resolución de nombres, cuando devuelven el resultado a la aplicación no pasan el valor del campo TTL devuelto en la respuesta por parte del servidor DNS. El prototipo de las 2 funciones que permiten realizar la traducción de un nombre a su correspondiente dirección IP puede verse en el siguiente cuadro:

```
struct hostent {
char      *h_name;                    /*official name of host */
char      **h_aliases;                /* alias list */
int       h_addrtype;                 /* host address type */
int       h_length;                   /* length of address */
char      **h_addr_list;              /* list of addresses from name server */
};

struct hostent *gethostbyname(const char *name);

struct addrinfo {
int                 ai_flags;         // AI_PASSIVE, AI_CANONNAME, etc.
Int                 ai_family;        // AF_INET, AF_INET6, AF_UNSPEC
Int                 ai_socktype;      // SOCK_STREAM, SOCK_DGRAM
Int                 ai_protocol;      // use 0 for "any"
size_t              ai_addrlen;       // size of ai_addr in bytes
struct sockaddr     *ai_addr;         // struct sockaddr_in or _in6
char                *ai_canonname;    // full canonical hostname
}

int getaddrinfo(const char *node, const char *service, const struct addrinfo *hints, struct
addrinfo **res);
```

Cuadro 3.8.1: Prototipo de las funciones gethostbyname y getaddrinfo

Como puede visualizarse, dentro de la información devuelta por cada una de las funciones, no existe información acerca del campo TTL en la respuesta que el sistema operativo expone a las aplicaciones. Esto último determina que aquellas aplicaciones que quieran implementar su propia cache de resolución, deberán seleccionar un valor determinado para definir el tiempo de vida que posee cada recurso en la caché. Esto último puede suponer problemas de inconsistencia entre la información que pudiera almacenar la aplicación y la información que pudiera poseer el sistema operativo, ya que un registro expirado en la cache del sistema operativo podría seguir siendo considerado válido por una aplicación que haga caching.

Históricamente los navegadores web han optado por almacenar los registros de DNS resueltos por intervalos de tiempo estipulados, en vez de solicitar cada resolución al sistema operativo. Esto último se debe a la necesidad de incrementar la performance al momento de devolver los resultados. Por ejemplo, el navegador web Internet Explorer en sus versiones anteriores a la 3.0 define el período de validez de los registros en 24 horas. Versiones posteriores a la 4.0 almacenan la información por un período de 30 minutos. Este comportamiento ha sido documentado por parte del fabricante en el sitio https://support.microsoft.com/en-us/kb/263558. Si bien esta técnica permite acelerar la navegación, atenta

contra la necesidad de realizar un cambio en un registro de DNS y que el mismo sea reflejado en cortos lapsos de tiempo por parte de los clientes. Los desarrolladores de navegadores web generalmente no exponen la información a los usuarios. Para el caso particular del navegador Google Chrome, no existe documentación oficial que indique cual es la postura adoptada frente a esta situación. En capítulos posteriores se intentará analizar a través de la captura de paquetes utilizando la herramienta Wireshark[9] cuál es el tiempo utilizado por los mismos.

3.9 Actualización de Zonas

Los registros de recursos dentro de una zona determinada deben ser administrados. A menudo surgen cambios que implican modificar la base de datos de resolución, de modo tal que un nombre definido con ciertos valores comience a resolverse con otros. Tal es el caso del cambio de dirección IP de un servidor, el cual deberá reflejarse en el servicio DNS para que cuando los usuarios quieran acceder al mismo utilizando su nombre, el servidor DNS devuelva la nueva dirección IP. Adicionalmente, puede resultar necesario agregar nuevos registros de resolución en una determinada zona.

Los cambios en la base de datos de resolución pueden realizarse editando los archivos con un editor de texto, o bien a través de alguna interfaz definida, como por ejemplo NSUPDATE que permite la realización de actualizaciones dinámicas sobre una zona dada. La primera metodología, implica editar el archivo de zona con un editor de texto, modificar el registro deseado, incrementar el número de serie de la zona y salvar los cambios al archivo. Por último debe indicarse al proceso BIND que relea nuevamente la configuración de la zona, y en caso de que la misma posea algún servidor secundario, notifique los cambios a dicho servidor. Es muy importante modificar el número de serie de la zona, por un número de serie superior al que se encuentra configurado al momento de realizar el cambio. Para ello, se utiliza la convención **AAAAMMDDNN**, donde **AAAA** es el año, **MM** representa al mes, **DD** representa al día y **NN** representa a un número autoincremental de 00 a 99. Si el número de serie no es actualizado, los servidores DNS no transferirán la zona modificada, e inclusive otros servidores DNS no serán notificados del cambio.

Otra opción para la realización de modificaciones en las zonas de DNS, es la utilización de la interfaz NSUPDATE. Se trata de una utilidad incluida a partir de la versión 8 del servidor BIND, que permite realizar modificaciones a los archivos de zona de manera sencilla. La utilidad permite agregar o quitar un registro de recurso de una zona de DNS, sin la necesidad de editar los archivos manualmente. A través de la ejecución del comando, se pueden enviar requerimientos de actualización dinámica de acuerdo al estándar definido en la RFC 2136 [29]. Cuando una zona es administrada a través de actualizaciones dinámicas, no debe ser

[9] Wireshark: Analizador de protocolos de red - http://www.wireshark.org

editada su configuración manualmente. El siguiente cuadro muestra un ejemplo de ejecución del comando nsupdate**:**

```
# nsupdate
> server dns1.ejemplo.com.ar
> update delete www.ejemplo.com.ar A
> update add www.wjwmplo.com.ar 86400 A 192.168.1.2
> send
```

Cuadro 3.9.1: Ejemplo de ejecución del comando nsupdate en GNU/Linux

La primera línea muestra la ejecución del comando, bajo un entorno GNU/Linux. La segunda línea selecciona el servidor DNS sobre el cuál se realizará la actualización. Luego se puede observar la sintaxis de eliminación y agregado de un registro en la segunda y tercera línea respectivamente. Por último la sentencia *send*, enviará al servidor DNS el requerimiento de actualización automática.

Tanto el servidor DNS como la herramienta nsupdate, proveen mecanismos de seguridad que permiten la autenticación de los requerimientos de actualización. Para ello se utilizan les extensiones de seguridad definidas en DNSSEC[30]. Con el fin de autenticar los requerimientos de actualización de zonas y proveer mecanismos de seguridad se utiliza un conjunto de claves compartidas. La extensión DNSSEC se explica en detalle más adelante en el presente capítulo. El cuadro 3.9.2 muestra un ejemplo de actualización dinámica utilizando un esquema de autenticación a través de certificados digitales sobre un servidor BIND, que previamente configuró su zona para permitir actualizaciones dinámicas:

```
# nsupdate –k /ruta/hacia/la/clave/clave.key
> server dns1.ejemplo.com.ar
> update delete www.ejemplo.com.ar A
> update add www.wjwmplo.com.ar 86400 A 192.168.1.2
> send
```

Cuadro 3.9.2: Ejemplo de ejecución del comando nsupdate en GNU/Linux utilizando mecanismos de autenticación

Adicionalmente, al momento de definir la zona dentro del servidor DNS debe indicarse qué clave tiene permitida la actualización de datos sobre la misma zona. Esta configuración será analizada en la siguiente sección.

3.10 Extensiones de Seguridad DNSSEC

Las Extensiones de Seguridad para el Sistema de Nombres de Dominio (DNSSEC) o Domain Name System Security Extensions son un conjunto de especificaciones de la IETF (Internet Engineering Task Force) que agregan funciones de seguridad al protocolo DNS. El hecho de que la funcionalidad se agregue al protocolo ya existente, permite que DNSSEC y DNS sean compatibles entre sí. Las extensiones, agregan las características de integridad y autenticidad a DNS, utilizando técnicas de criptografía de clave pública. De este modo es posible determinar que una traducción de nombre de dominio a IP o viceversa es legítima,

está autorizada por la entidad que debe responder por la zona de resolución y no ha sido modificada en el camino. Las extensiones fueron creadas con el fin de proteger a los clientes de datos falsificados tales como los que se producen por envenenamientos de cache o suplantación de DNS.[31]

DNSSEC utiliza para garantizar la legitimidad de las zonas, la misma jerarquía de delegación que utiliza el servidor DNS. Los servidores de nivel 1 firman digitalmente sus zonas y los subdominios que se generan sobre las mismas. De este modo cada responsable de un subdominio determinado podrá realizar la misma acción con los subdominios que de él dependan. Es importante destacar que las extensiones de seguridad están orientadas a brindar autenticidad de los datos y no cifrado o encripción de los mismos.

Otra funcionalidad de DNSSEC es la de brindar autenticación. Históricamente se ha utilizado el concepto de autenticar a los usuarios que pueden realizar modificaciones sobre una zona de DNS utilizando su dirección IP. Es de este modo, que generalmente para que un servidor primario pueda comunicar cambios de zonas a un secundario o viceversa, en la configuración para la definición de la zona de DNS se indica cuál es la dirección IP de la que se permite realizar modificaciones. El cuadro **3.10.1** muestra un ejemplo de definición de zona que permite que la misma se actualizada desde una determinada dirección IP:

```
zone "ejemplo.com.ar" in {
type master;
file "/etc/bind/zonas/ejemplo.com.ar.conf";
allow-update  { 192.168.1.254 };
};
```

Cuadro 3.10.1: Ejemplo de definición de zona que permite actualizaciones basadas en direcciones IP

Utilizando la funcionalidad provista por las extensiones de seguridad, la definición anterior puede modificarse de modo tal que la actualización sea permitida a aquel que posea la llave criptográfica adecuada para realizar la actualización. El cuadro **3.10.2** muestra el ejemplo de configuración:

```
key “ejemplo” {
algorithm HMAC-MD5;
secret “xqaD2Hg0DsDdDFfF9Gr8r3j4rGgG09Gerf2SgSe3erA4E62ee3q4Eq2dq3q4GfD9==”
}

zone "ejemplo.com.ar" in {
type master;
file "/etc/bind/zonas/ejemplo.com.ar.conf";
allow-update  { key ejemplo };
};
```

Cuadro 3.10.2: Ejemplo de definición de zona que permite actualizaciones basadas en certificados digitales

3.11 Conclusiones finales del capítulo

Luego de haber analizado los conceptos del protocolo DNS, resulta claro que se trata de un protocolo de aplicación estable y debidamente documentado, el cual puede considerarse uno de los pilares más importantes de la red InterNet. Las premisas con las que fue creado, han permitido que el protocolo provea la alta disponibilidad que la red necesita, teniendo en cuenta que el 99% de las comunicaciones que se realizan requieren de una resolución de nombres previa. La posibilidad de delegar la administración de la base de datos de resolución en organizaciones de diversas características, sin que una pueda afectar el comportamiento de otra es clave para lograr la estabilidad deseada. La sencillez de su configuración y la posibilidad de escalar sus funcionalidades a través del agregado de nuevos registros de recurso, ha hecho que el protocolo se adapte acordemente a las constantes demandas tecnológicas de la red. Tal es así, que dentro de sus premisas básicas de creación no se estipuló la necesidad de proveer mecanismos de seguridad, y hoy en día el protocolo la provee a través de una extensión conocida como DNSSEC y documentada en las RFC 4033.

Tanto los requerimientos de hardware como los que hay que cumplimentar ante las autoridades de registro cuando se desea registrar de un nombre de dominio son accesibles a pequeñas organizaciones y usuarios finales al ser de muy bajo costo.

El servicio DNS es provisto por diversas aplicaciones sencillas de administrar que puede instalarse en plataformas de hardware pequeñas e inclusive con distintos sistemas operativos, sin necesidad de contar con equipamiento específico y dedicado para tal fin. El esquema de delegación de dominios hace que diversas organizaciones permitan a usuarios registrar subdominios a costos bajos e inclusive de forma gratuita.

DNS demuestra ser un protocolo confiable y de bajo costo, accesible a pequeños usuarios y grandes proveedores de servicio. Su amplia adopción y adaptabilidad, han permitido que su uso se extienda en el tiempo y se adapte a los requerimientos. La posibilidad de administrar tiempos de expiración ha permitido que pequeños usuarios lo adopten como una herramienta simple para brindar alta disponibilidad de servicios cambiando los registros de resolución ante cada necesidad.

Capítulo 4

Sistemas de Monitoreo

4.1 Introducción

A medida que las infraestructuras de redes y servidores se tornan más complejas, resulta indispensable contar con herramientas que permitan realizar el monitoreo de las mismas de forma automatizada. Es común encontrar en infraestructuras pequeñas procedimientos de verificación para los sistemas, que generalmente son llevados a cabo por personal dedicado a la tarea. Dichos procedimientos determinan frecuencia, equipos y servicios que deben ser verificados manualmente por algún operador destinado casi exclusivamente a realizar el monitoreo. Este esquema es claramente poco escalable y se torna prohibitivo de uso a medida que la infraestructura crece. En la actualidad existe una amplia gama de productos de carácter comercial y libre que permiten programar el monitoreo, de modo tal que el mismo pueda ser llevado a cabo por la herramienta, y ante determinados eventos pueda actuar en consecuencia. Estas herramientas permiten minimizar los tiempos de caída de servicio al notificar de inmediato a través del correo electrónico o mensaje de texto a los administradores acerca del error. Adicionalmente proveen interfaces con el fin de que los operadores puedan verificar el estado global de la infraestructura a través de diversos reportes.

A lo largo del presente capítulo se analizará la importancia de los sistemas de monitoreo, así como también la funcionalidad básica de la herramienta de monitoreo *Icinga*. *Icinga[34]* es una bifurcación del proyecto *Nagios[35]*, que hereda todas las cualidades del anterior y agrega importantes funcionalidades dentro de las cuales se destacan sus interfaces y la posibilidad de interoperar con otras aplicaciones. *Nagios* es un sistema de monitoreo de redes y servicios ampliamente utilizado. Es de código abierto y ha demostrado gran robustez y escalabilidad desde su fecha de lanzamiento inicial en el año 1999.

4.2 Características y funcionamiento

Icinga es un sistema de monitoreo que se encarga de verificar continuamente el estado de los dispositivos y servicios con el fin de detectar sistemas lentos o fuera de servicio. Ante la detección de un evento, el sistema de monitoreo notificará al administrador acerca de la anomalía para que el mismo dé una solución al problema. Se trata de un sistema de código abierto escrito mayoritariamente en el lenguaje de programación C[10]. Su modo de licenciamiento y distribución están encuadrados bajo los términos de la GNU Public License Version 2 (GPLv2[11]).

Fue originalmente creado como una bifurcación del proyecto *Nagios* en el año 2009. Durante los años 2010 y 2011 el proyecto evolucionó agregando nuevas funcionalidades que su antecesor no preveía, lo que captó un importante número de usuarios. A fines del año 2012 fue completamente reescrito con el fin de proveer una mayor performance que permita garantizar la escalabilidad del sistema. En Junio de 2014 el

[10] C: Lenguaje de Programación creado en el año 1972 por Dennis Ritchie y Laboratorios Bell
[11] GPLv2: http://www.gnu.org/licenses/gpl-2.0.html

proyecto Icinga lanzó la primera versión estable del sistema denominado “Icinga 2”. Dentro de sus características principales se encuentran [36]:

- **Monitoreo de Servicios:** Permite la realización de chequeos a los servicios más comunes como por ejemplo HTTP, SMTP, POP y FTP entre otros), así como también el agregado de nuevos servicios programables por parte de los usuarios.
- **Monitoreo de recursos:** Provee monitoreo de recursos para los sistemas operativos más comunes. Dentro de los recursos a monitorear se encuentran estado de CPU, memoria y disco entre otros.
- **Monitoreo adaptable:** Debido a que los programas que realizan el monitoreo son de código abierto, es posible escribir cualquier tipo de monitoreo que se adapte a las necesidades.
- **Programación de los chequeos:** Permite definir los períodos de tiempo en los que deben realizarse los chequeos. De este modo se podría excluir una franja horaria o bien un día para la realización del chequeo.
- **Notificaciones:** Permite definir notificaciones ante los eventos. Cuando se produce la caída de un servicio monitoreado, la herramienta puede enviar un mail o mensaje de texto al administrador para notificar la anomalía.
- **Manejadores de Eventos:** Cada vez que ocurra un evento, la herramienta puede ejecutar el comando definido para manejar el evento, que permita por ejemplo reiniciar un servicio o realizar alguna configuración específica.
- **Interfaz de monitoreo:** Provee una interfaz web sencilla e intuitiva de usar que permite verificar el estado general de la infraestructura monitoreada de modo gráfico. La misma permite programar chequeos y visualizar registros de eventos pasados.
- **Interfaz para Integración:** Provee una Interfaz de Programación de Aplicaciones o API por sus siglas en inglés, que permite integrar los resultados del monitoreo con aplicaciones externas. A través de la realización de consultas utilizando el estilo de arquitectura REST, se puede acceder al estado de monitoreo de los equipos o servicios incluidos en Icinga. La arquitectura REST se describe en el **Anexo A** del presente trabajo.

4.3 Arquitectura

Icinga ha sido desarrollado utilizando una arquitectura modular, con el fin de beneficiar el trabajo colaborativo sin afectar la labor entre los distintos grupos de desarrollo. Sus componentes principales son el core o corazón de la aplicación, la interfaz de usuario y la base de datos, tal como se muestra en la figura **4.3.1**:

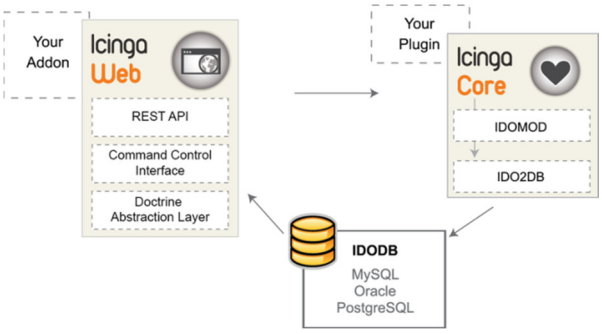


Figura 4.3.1: Arquitectura de Icinga versión 2

Fuente: https://www.icinga.org/2011/08/05/farewell-icinga-api/

El módulo Core o Núcleo es el encargado de administrar las tareas de monitoreo y obtener los resultados de los chequeos programados para cada servicio. Es el módulo que provee la lógica de monitoreo a la aplicación. Cuando se realiza la instalación de Icinga, se puede instalar únicamente el módulo Core e IDODB (Icinga Data Out Database) que es el módulo responsable de la administración de los datos. El Core interactúa con el módulo IDODB, a través de su componente IDOMOD (Icinga Data Out Module) e IDO2DB (Icinga Data Out to Database). Esta arquitectura permite encapsular la funcionalidad de cada módulo y abstraerla de los otros. A través de este diseño se permite que los datos de monitoreo de Icinga puedan residir en diversos motores de base de datos, como ser MySQL[12], Oracle[13] o PostgreSQL[14], sin que ello afecte a la comunicación con el módulo Core. El mecanismo de envío y consulta de datos hacia la base de datos es encapsulado por las componentes IDOMOD e IDO2DB incluidas en el core de Icinga. Los plugins permiten la extensión del modelo de chequeos que provee Icinga. Los mismos se agregan como funcionalidad al Core y pueden ser programados en cualquier lenguaje de programación, respetando la interfaz establecida en el core. Los mismos son ejecutados desde la línea de comandos del servidor que ejecuta el software Icinga, con el fin de verificar el estado de un equipo o un servicio. Existe una gran diversidad de plugins ya desarrollados, los cuales pueden ser descargado de distintas fuentes y anexados a la instalación.

La interfaz está representada por el módulo web el cual interactúa con el core enviando comandos y consume información de la base de datos con el fin de graficar el estado de monitoreo. Provee una interfaz gráfica accesible a través del protocolo HTTP, que permite la definición de usuarios y roles. A través de la interfaz puede verificarse el estado de monitoreo de los chequeos ejecutados por el core. También provee una API para la integración con otras aplicaciones y la posibilidad de interactuar con el core y la base de datos a través de la línea de comandos. La posibilidad de mantener un desarrollo modular, permite también que Icinga provea una interfaz para dispositivos móviles. A través de dicha interfaz se puede acceder a un subconjunto de opciones que provee la interfaz web.

[12] MySQL: Motor de base de datos Relacional de código abierto – http://www.mysql.com
[13] Oracle: Motor de base de datos Relacional comercial – http://www.oracle.com
[14] PostgreSQL: Motor de base de datos Relacional de código abierto –http://www.postgresql.org

4.4 Principios de monitoreo en Icinga

El módulo Core de Icinga es el responsable de ejecutar cada plugin con sus correspondientes parámetros, de acuerdo a la programación realizada. Su función principal es programar periódicamente la invocación de los plugins y obtener el resultado de su ejecución. Los plugins encapsulan la funcionalidad necesaria para la realización del chequeo en base a los parámetros definidos por el módulo Core y los resultados obtenidos de cada entidad monitoreada. El proceso de ejecución de plugins, puede verse gráficamente en la figura 4.4.1:

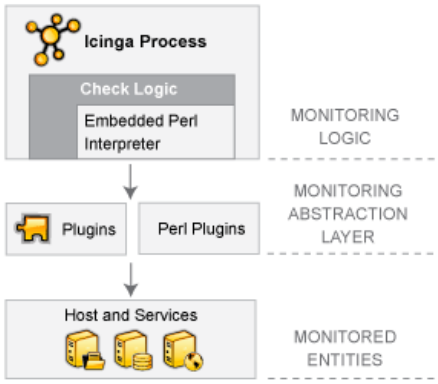


Figura 4.4.1: proceso de ejecución de chequeos a través de plugins en Icinga

Fuente: http://docs.icinga.org/latest/en/plugins.html

Los plugins actúan como una capa de abstracción entre la capa lógica de monitoreo presente en el núcleo de Icinga y los servidores y servicios que son efectivamente monitoreados. La lógica de esta arquitectura permite que cualquier entidad u objeto pueda ser monitoreado independientemente del tipo de dispositivo y servicio que provea. El único requerimiento que debe poseer un objeto para que pueda ser monitoreado por Icinga, es que permita la creación de un plugin que pueda ser invocado y devuelva los datos necesarios para su procesamiento.

Con el fin de que un plugin pueda ser presentado a Icinga para su utilización, debe cumplir al menos con 3 requisitos [37]:

1. Que el mismo pueda ser ejecutado localmente en el servidor que corre el software de monitoreo Icinga. Esto implica la instalación de intérpretes en caso de ser necesario con el fin de que el script pueda ser ejecutado.
2. Al momento de finalizar su ejecución, debe devolver un valor numérico que indique el resultado de la ejecución del chequeo. Los posibles estados son:
    - **0 - OK:** Indica que el servicio o servidor chequeado se encuentra en estado correcto.
    - **1 - WARNING:** Indica que el servicio o servidor chequeado está presentando algún inconveniente.
    - **2: - DOWN:** Indica que el servicio o servidor chequeado se encuentra caído o fuera de servicio.
    - **3 - UNKNOWN:** Indica que no se puede obtener información del servicio o servidor chequeado. Generalmente porque el mismo está presentando una falla, lo cual también es considerado como DOWN.

3. El plugin también deberá devolver como valor de retorno una cadena de texto que indique en palabras el resultado de la ejecución.

El cuadro **4.4.1** muestra un plugin de ejemplo escrito en BASH, el cual verifica el espacio en disco disponible de la partición donde se encuentra montado el sistema de archivos raíz / en un servidor GNU/Linux. En caso de detectar que la ocupación es superior al 85%, el plugin arrojará una alerta:

```
#!/bin/bash

used_space=`df -h / | grep -v Filesystem | awk '{print $5}' | sed 's/%//g'`

case $used_space in

[1-84]*)

        echo "OK - $used_space% of disk space used."

        exit 0

        ;;

[85-90]*)

        echo "WARNING - $used_space% of disk space used."

        exit 1

        ;;

[91-100]*)

        echo "CRITICAL - $used_space% of disk space used."

        exit 2

        ;;

*)

        echo "UNKNOWN - $used_space% of disk space used."

        exit 3

        ;;

esac
```

Cuadro 4.4.1: Código de ejemplo de script para Icinga escrito en BASH

Con el fin de que el mismo pueda ser ejecutado por ICINGA, el script debe ser definido lógicamente para el núcleo de monitoreo. Para ello deberá definirse un nombre lógico a través de la sintaxis **define command** y asociarla a la ruta completa hacia el script a ejecutar para obtener los resultados. Una vez realizado esto, el comando estará disponible para ser utilizado como un chequeo. La sintaxis para la definición del comando se muestra en el cuadro **4.4.2**:

```
define command{

command_name check_disk_space

command_line /usr/lib/Icinga/pugins/check_disk_space.sh

}
```

Cuadro 4.4.2: Definición de comando en Icinga

Adicionalmente al resultado que pueda proveer un comando para el monitoreo de un objeto del tipo host (UP, DOWN), Icinga maneja un estado adicional que permite evitar los falsos positivos. Cuando se detecta un problema en un objeto del tipo host o un servicio, Icinga vuelve a chequear el objeto con el fin de confirmar efectivamente la caída y evitar falsos positivos. La cantidad de re chequeos e intervalo entre los mismos dependerá de las opciones configuradas en el objeto. Durante el período de tiempo en el cual se realicen los chequeos para confirmar el problema, el objeto chequeado se encontrara bajo un estado conocido como **SOFT**, que indica el objeto está presentando alguna anomalía, pero que la misma aún no ha sido confirmada. Una vez que la anomalía haya sido confirmada, el objeto pasará a estar en un estado conocido como **HARD**.

4.5 Configuración de Icinga

La configuración del servidor de monitoreo Icinga, está basada en archivos dentro del sistema de archivos. Una vez instalado el servicio, se lo configura a través de su archivo principal de configuración llamado **icinga2.conf**, que generalmente se encuentra en el directorio **/etc/icinga2/** en una instalación estándar. En este archivo se definen los parámetros básicos que determinarán el funcionamiento del servicio de monitoreo. El modo de configuración del servicio ha cambiado en la versión 2 del software, diferenciándose del modelo de configuración que utilizaba Icinga versión 1 y el software de monitoreo Nagios. En versiones anteriores, el archivo principal de configuración contenía cientos de configuraciones globales. El nuevo modelo simplifica la configuración, definiendo solo un conjunto de configuraciones globales que pueden ser manipuladas. Otro cambio significativo reside en la posibilidad de mantener las configuraciones en archivos separados. Esto permite agrupar las configuraciones similares en diversos archivos, los cuales pueden ser administrados independientemente sin la necesidad de tener un único archivo global de configuraciones y de gran tamaño. Dentro del directorio de configuración del servicio, se crean una serie de subdirectorios que contienen archivos de configuraciones específicos. Dentro de los más importantes se destacan:

- **conf.d:** En este directorio se encuentran los archivos de configuración que permiten definir objetos, chequeos, períodos de tiempo para ser utilizados en los chequeos, etc. La particularidad de este directorio es que cualquier archivo cuyo nombre termine con las siglas **.conf**, será evaluado e incluido en la configuración global al momento de iniciar el software de monitoreo.
- **features-available:** Define un conjunto de configuraciones globales que pueden ser utilizadas por Icinga. Los archivos de este directorio permiten agregar funcionalidad al software de monitoreo, de acuerdo a las necesidades de la instalación y sin la necesidad de realizar grandes configuraciones. Generalmente los archivos de este directorio no deben editarse, ya que contienen las configuraciones necesarias por defecto para incluir al archivo de configuración gobal

- **features-enabled:** Contiene enlaces simbólicos a los archivos de configuración del directorio **features-available**. Lo enlaces de este directorio no deben ser manipulados por el usuario directamente, sino que deben manejarse a través de la interfaz de línea de comandos **icinga2 feature**. Esta interfaz permite consultar los módulos disponibles, habilitar y/o eliminar módulos configurados. Su sintaxis básica de uso se muestra la figura 4.5.1

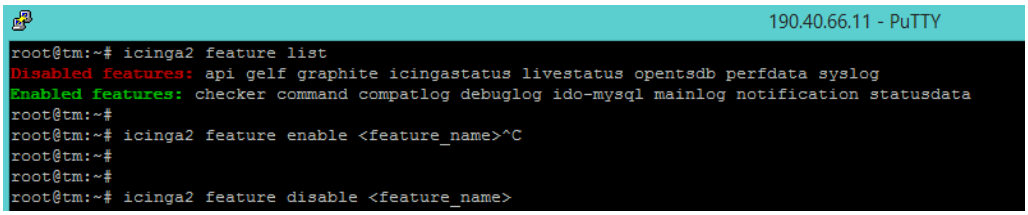


Figura 4.5.1: Utilización de la interfaz **icinga2 feature** en Icinga

Con el fin de poder comenzar a realizar monitoreos, deben definirse los objetos que utilizará Icinga. Los objetos son elementos que se encuentran involucrados en la lógica de monitoreo, como por ejemplo hosts, servicios, comandos, períodos de tiempo, contactos, etc. Los objetos pueden ser definidos en un único archivo de configuración o en varios incluidos en el directorio **/etc/icinga2/conf.d/**. Dentro de los objetos más importantes que se pueden definir se encuentran:

- **Hosts:** Representan generalmente a dispositivos físicos que se desean monitorear. Ejemplos de este tipo pueden ser una impresora, un servidor, un router, etc. Los objetos del tipo host tienen asociado un nombre, una dirección IP que permite referenciarlos y un conjunto de comandos que definen los chequeos que se realizan sobre el mismo. Su sintaxis básica de definición se muestra en el cuadro de configuración 4.5.1:

```
object Host “nombreDeHost” {
        check_command = “nombreDeComando”
        address = “direccionIP”
        event_command = “nombreDeComando”
        enable_event_handler = [true|false]
        check_interval = intervaloEnSegundos
        retry_interval = intervaloEnSegundos
        max_check_attempts = numeroDeIntentos
        check_period = “periodoDeTiempo”
        vars.<nombreDeVariablePersonalizada> = “unValor”
}
```

Cuadro 4.5.1: Definición de objeto Host en Icinga

La lista completa de atributos que pueden configurarse sobre un objeto de tipo **Host** puede consultarse directamente del sitio de Icinga, en la siguiente URL. Notar que existen un conjunto de atributos obligatorios para la definición del objeto, y otro conjunto opcional: http://docs.icinga.org/icinga2/latest/doc/module/icinga2/chapter/object-types#objecttype-host

- **CheckCommands**: Representan el comando dentro de un objeto del tipo Host que es utilizado para realizar el monitoreo del mismo. Los objetos del tipo Host pueden poseer diversos comandos para su chequeo, los cuales permiten realizar monitoreos independientes o inclusive asociarlos con el fin de definir esquemas de dependencia de servicios. Ejemplos de chequeos son uso de disco, uso de CPU, protocolo HTTP, protocolo SMTP, etc. Su sintaxis básica de

definición es:

```
object CheckCommand "nombreDeComando" {
        import "plugin-check-command"
        command = "/usr/lib/nagios/plugins/<plugin> -H '$address$'"
}
```

Cuadro 4.5.2: Definición de objeto CheckCommand en Icinga

La lista completa de atributos que pueden configurarse sobre un objeto de tipo **CheckCommand** puede consultarse directamente del sitio de Icinga, en la siguiente URL. Notar que existen un conjunto de atributos obligatorios para la definición del objeto, y otro conjunto opcional: http://docs.icinga.org/icinga2/latest/doc/module/icinga2/chapter/object-types#objecttype-checkcommand

- **EventCommands**: Definen un comando que será ejecutado cada vez que se detecte un cambio en el estado de monitoreo de un Host determinado. Los objetos del tipo **EventCommand** son utilizados comúnmente con el fin de subsanar un problema detectado. Un ejemplo de utilización podría ser el de iniciar un determinado servicio, luego de haberse detectado una caída. La sintaxis de definición es similar a la de un objeto del tipo **CheckCommand**

```
object EventCommand "nombreDeComando" {
        import "plugin-event-command"
        command = "/usr/lib/nagios/plugins/<notificationplugin>"
}
```

Cuadro 4.5.3: Definición de objeto EventCommand en Icinga

- **Timeperiods**: Este tipo de objeto es utilizado para controlar cuando se ejecutarán los chequeos o bien cuando se notificará ante un cambio de estado de host o servicio. Su sintaxis básica de definición es:

```
object TimePeriod “unNombre”{
        import "legacy-timeperiod"
        ranges = {
        sunday          unaFranjaHoraria
        monday          unaFranjaHoraria
        tuesday         unaFranjaHoraria
        wednesday       unaFranjaHoraria
        thursday        unaFranjaHoraria
        friday          unaFranjaHoraria
        saturday        unaFranjaHoraria
        }
}
```

Cuadro 4.5.4: Definición de objeto TimePeriod en Icinga

Una vez definidos los objetos necesarios, se puede iniciar el servicio de monitoreo, para que el mismo comience a programar los chequeos e informe los resultados a través de alguna de sus interfaces

4.6 Interfaces de Acceso

La arquitectura de Icinga, permite que la lógica de monitoreo sea totalmente independiente del modo en el que se accede a los datos para su visualización. Gracias a esta separación de funciones, diversas interfaces pueden ser creadas con el fin de acceder a los datos almacenados. En versiones anteriores a la 2 de Icinga, la instalación del motor de monitoreo ya incluía una interfaz conocida como "Classic Interface". A partir de la versión 2, se desacopló la lógica de monitoreo de la de visualización de modo tal de dar lugar a diversas interfaces que puedan acceder a los datos y mostrarlos de acuerdo a cada necesidad. En la sección **4.5** del presente capítulo se indicó la posibilidad de habilitar o deshabilitar características en el núcleo de Icinga. En particular las características **statusdata**, **compatlog** y **command** son las que permiten exponer la funcionalidad del core de Icinga para que pueda ser accedida por diversas interfaces para obtener datos acerca del estado del monitoreo y configuraciones de la herramienta.

La interfaz más conocida de Icinga se denomina **Classic Interface**. Se trata de la interfaz más sencilla para la visualización de eventos. Una de sus pantallas más comunes es la de visualización de estado general (Status View), donde se puede consultar el estado de los objetos monitoreados y sus servicios. Un ejemplo de esta pantalla es el siguiente, donde se pueden ver los objetos del tipo host, con sus servicios y el estado:

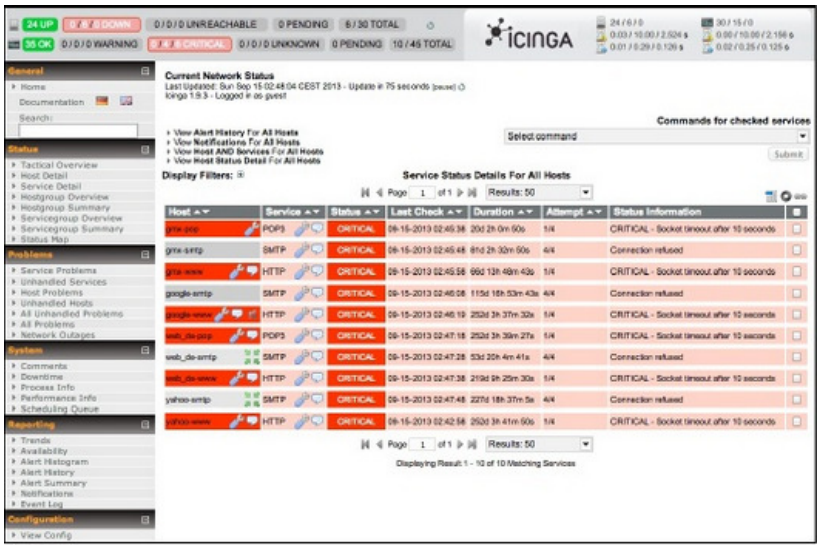


Figura 4.6.1: Ejemplo de interfaz **Classic Interface** de Icinga

Otra interfaz que puede ser utilizada para monitorear el estado general de los servicios es la **Web Interface**. La misma debe instalarse como un módulo adicional al servicio de monitoreo y provee una funcionalidad similar a la interfaz Classic. Se trata de una interfaz escrita en lenguaje PHP que utiliza HTML5 para la visualización y dentro de sus ventajas principales se encuentra la posibilidad de acceder a la misma de modo gráfico o a través de su interfaz REST. A diferencia de la anterior interfaz, Icinga Web hace un manejo más granular de los usuarios de modo tal que cada uno pueda definir su propia interfaz de monitoreo agrupando los reportes y estadísticas de acuerdo al gusto del usuario. La siguiente figura muestra los objetos del tipo Host monitoreados junto con sus servicios y estado

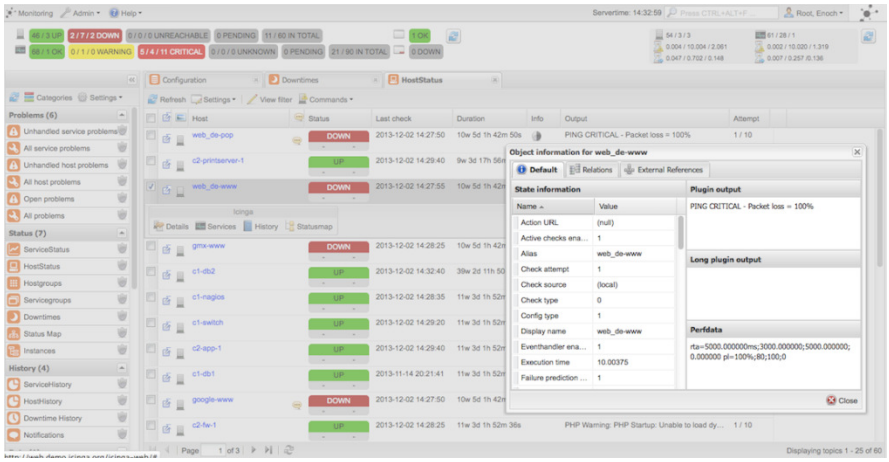


Figura 4.6.2: Ejemplo de interfaz **Web Interface** de Icinga

La interfaz REST que provee la versión Web de la visualización, permite que los datos de monitoreo puedan ser accedidos externamente con el fin de integrarlos a otras aplicaciones. Esta funcionalidad permite la creación de nuevas interfaces (por ejemplo para dispositivo móviles) o bien el acceso directo a los resultados de monitoreo por parte de otras aplicaciones con el fin de tomar decisiones en base a los datos obtenidos. Dentro de sus características se encuentran la posibilidad de consultar el 90% de los datos de monitoreo a través de la interfaz, la posibilidad de exportar los datos en formato json o xml, capacidad de realizar filtro de los resultados y brindar acceso en base a mecanismos de autorización. Con el fin de acceder a la interfaz REST, se deben realizar requerimientos HTTP del tipo GET o POST siguiendo un formato establecido. El formato básico documentado para el acceso define que el formato de la URL debe seguir el siguiente patrón [38]:

```
localhost/icinga-web/web/api/ TARGET / COLUMNS / FILTER /
```

Cuadro 4.6.1: Formato genérico de requerimiento REST a Icinga

A continuación se describe el significado de cada uno de los parámetros de la URL:

- **TARGET:** Define el tipo de objeto que se desea obtener. Un ejemplo de este valor es **host**.
- **COLUMNS:** Define la lista de atributos del objeto que se desea incluir en el resultado del requerimiento
- **FILTER:** Permite filtrar la consulta siguiendo algún criterio específico como por ejemplo solicitar los datos de un determinado host.

El siguiente cuadro muestra un ejemplo de URL formada para realizar un requerimiento utilizando el paradigma REST, a la interfaz web de Icinga:

```
http://localhost/icinga-web/web/api/service/
filter[AND(HOST_CURRENT_STATE|=|0;OR(SERVICE_CURRENT_STATE|=|1;SERVICE_CURRENT_STATE|=|2)
)]/
columns[SERVICE_NAME|HOST_NAME|SERVICE_CURRENT_STATE|HOST_NAME|HOST_CURRENT_STATE|HOSTGRO
UP_NAME]/
authkey=APITEST123456/xml
```

Cuadro 4.6.2: Requerimiento REST de ejemplo a Icinga

4.7 Conclusiones finales del capítulo

Luego de estudiar las características del sistema de monitoreo Icinga, resulta claro que el mismo cumple con todas las cualidades para ser considerado un servicio confiable, estable y robusto. La utilización de este tipo de servicios permite realizar el monitoreo de las infraestructuras de red y cómputos de cada organización. Teniendo en cuenta la variedad de configuraciones que se pueden encontrar dependiendo del tipo de organización, resulta de suma utilidad contar con sistemas de monitoreo que sean adaptables. Icinga demuestra su adaptabilidad, a través de la posibilidad de definir chequeos a medida. Esta última cualidad provee a la herramienta un gran valor agregado, ya que cada organización puede programar sus propios chequeos de acuerdo a su necesidad.

La instalación y configuración del servicio Icinga puede realizarse de modo sencillo, lo que permite que sea accesible a pequeños usuarios y grandes proveedores de servicio. Su amplia adopción, así como también su adaptabilidad, han permitido que su uso se extienda en el tiempo y se adapte a los requerimientos. La posibilidad de contar con interfaces externas basadas en arquitecturas estándares como REST permite la integración con diversas herramientas, las que pueden consumir datos de monitoreo. Esta característica extiende la funcionalidad de Icinga, permitiendo que la misma sea casi ilimitada.

Capítulo 5

Propuesta de administración de tráfico basada en DNS

5.1 Introducción

El protocolo DNS, ha sido y sigue siéndolo, un gran pilar de la red de comunicaciones de Internet. Es a través del cual se permite localizar a los recursos en esta inmensa nube de contenidos, dirigiendo el tráfico que los usuarios demandan hacia un servidor u otro. Es por esta razón, que podemos comparar a este protocolo con un protocolo estricto de ruteo como BGP (salvando las diferencias que existen entre ellos), reconociendo que como resultado final, y actuando en distintos escenarios, ambos permiten que los requerimientos de los usuarios lleguen al destino solicitado. El protocolo BGP, por su naturaleza, permite identificar las fallas de red y converger por sus propios medios garantizando un nuevo camino para el ruteo de los datos solicitados. El protocolo DNS, provee únicamente localización de recursos mediante la resolución de nombres. Basándose en la técnica de localización se pueden encaminar los requerimientos de los usuarios hacia el punto de la red que se desee, modificando las respuestas del servicio de DNS y basando las mismas en la disponibilidad que presente un servicio determinado. De este modo, si un servicio es publicado a través de más de un ISP y cada ISP provee un direccionamiento distinto, el mismo será accesible a través de más de una dirección IP. Adicionalmente, si dependiendo del estado de la red, el servicio de resolución de nombres pudiera responder con una IP u otra, se lograría cambiar la ruta de los requerimientos hacia otra red, permitiendo la convergencia hacia un nuevo punto de acceso a los recursos. Por lo tanto podemos concluir que lo que le falta al protocolo DNS para aproximarse al funcionamiento de BGP, es una componente que se encargue de determinar los estados de la red, disparar la convergencia; y garantizar la consistencia y estabilidad.

5.2 Arquitectura general de la solución planteada

La herramienta “Traffic Manager”, permite realizar la gestión del tráfico de redes, utilizando el protocolo DNS para el encaminamiento de los datos. Se basa en la premisa de que cambiando los registros de resolución, se puede encaminar el flujo de los datos desde un enlace hacia otro punto de la red que permita acceder al servicio productivo de un modo similar al que lo hacen los protocolos de enrutamiento. Con el fin de determinar los estados de la red se utilizan herramientas de monitoreo que verifican constantemente a los servicios gestionados.

La solución implementada ha sido pensada bajo un desarrollo modular, adoptándose como premisa la reutilización de software que ya realice parte de las tareas de la implementación, con el fin de no

"reinventar la rueda[15]". Es por esta razón que se tomó la decisión de aprovechar el funcionamiento del servicio de monitoreo Icinga, así como también la implementación del servidor de nombres *Bind* y no reimplementar funcionalidad que ya se encuentra desarrollada, probada y estandarizada. En su totalidad se optó por utilizar software de código abierto provisto por la comunidad. Asimismo, todas las modificaciones y desarrollos adicionales realizados con el fin de interconectar las componentes que forman parte de la solución se distribuirán bajo la modalidad de código abierto.

La figura **5.2.1** muestra un diagrama acerca de las componentes que forman parte de la solución y la interacción entre ellas.

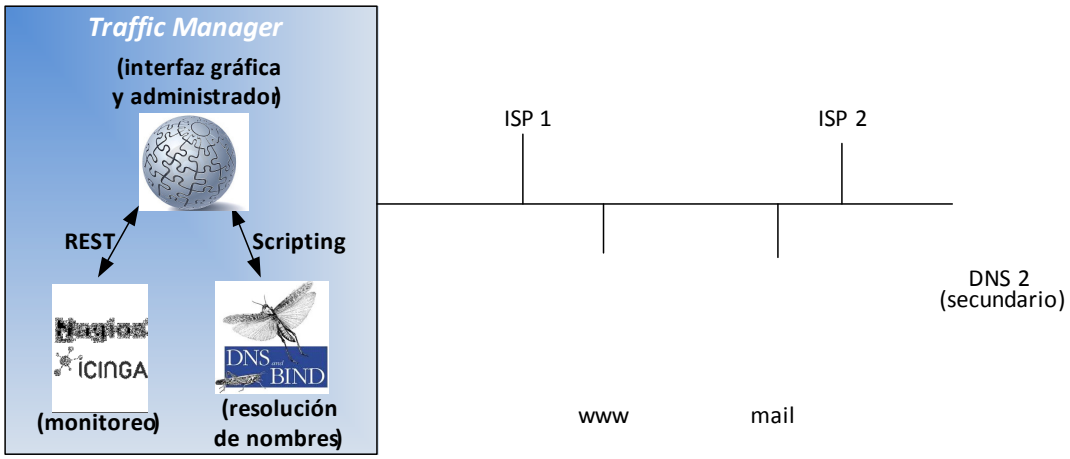


Figura 5.2.1: Componentes que forman parte de la solución Traffic Manager

Cuando se desea gestionar un servicio a través de la herramienta Traffic Manager, el mismo debe ser dado de alta usando la interfaz gráfica de administración. Al momento de registrar el servicio, debe proveerse un nombre que lo identifique y el dominio de DNS en el cuál será publicado, así como también el tipo de servicio a través del cual será monitoreado (http, icmp, pop o smtp entre otros). Adicionalmente deberán indicarse las direcciones IP que permitirán alcanzar el servicio. La primera dirección indicada corresponderá a la dirección IP que redirecciona al servidor a través del enlace de InterNet primario (ISP2 por ejemplo), y la segunda dirección será la que se utilizará cuando el servicio entre en contingencia (proveedor de servicios de InterNet ISP1). Por último, deben definirse las direcciones IP a las que se realizarán los chequeos con el fin de determinar la estabilidad de la red. Estas direcciones pueden ser las mismas que las anteriores, o diferir en caso de ser necesario chequear otro host con el fin de verificar la disponibilidad del vínculo. Cuando el servicio es dado de alta, el módulo de administración notificará inmediatamente a los módulos de monitoreo y resolución de nombres para que comiencen a realizar los chequeos de disponibilidad y resolver el nombre indicado en la dirección IP registrada. Ante la detección de

[15] Expresión que se refiere al fenómeno de necesitar encontrar solución a algún problema que seguramente haya sido resuelto por alguien que se ha enfrentado anteriormente al mismo.

una caída de servicio por parte del módulo de monitoreo, se informará al módulo administrador. El mismo determinará en base a la información provista por el módulo de monitoreo, la necesidad de invocar al servicio DNS para que modifique su registro de resolución apuntando a la segunda dirección IP registrada. De este modo, sucesivos requerimientos de resolución devolverán la dirección IP del enlace que se encuentra productivo. Cuando se detecte que el problema haya sido resuelto, el módulo de monitoreo informará nuevamente al módulo de resolución de nombres con el fin de que nuevamente vuelva a cambiar los registros de resolución de DNS apuntando a la dirección IP primaria.

Las funciones y particularidades de cada módulo que compone la solución son:

- **Monitoreo:** Es el responsable de realizar los chequeos de disponibilidad sobre los servicios configurados. Cuando se agrega un nuevo servicio a través de la interfaz gráfica, se generan los archivos de configuración correspondientes para que el módulo de monitoreo comience a realizar los chequeos. Es adicionalmente quien invoca al módulo administrador ante la ocurrencia de un evento. Para la funcionalidad de este módulo, se utiliza el software de monitoreo Icinga, al cual se ha hecho referencia en el capítulo 4 (cuatro) del presente trabajo. La decisión acerca de la utilización de Icinga sobre Nagios se basó en la posibilidad de acceder a los resultados de monitoreo a través de una interfaz REST. Si bien Nagios provee dicha funcionalidad, se requieren ciertas modificaciones al código que implican su recompilación. Icinga provee una interfaz de comunicación (API) nativa, la cual permite exportar los datos de monitoreo en formato JSON o XML. Esta funcionalidad es de vital importancia, ya que pueden consultarse datos externamente desde la lógica de programación de los módulos administrador e interfaz gráfica sin necesidad de acceder a estructuras internas del módulo de monitoreo.
- **Resolución de nombres:** Tiene como función principal la traducción de nombres basados en un esquema de DNS a direcciones IP. Es la componente del sistema que se encarga de la localización de los servicios y recursos gestionados. Para brindar dicha funcionalidad, se utilizó el software ISC Bind, que permite de manera sencilla gestionar los recursos de resolución de nombres y ha sido abarcado en el capítulo 3 (tres) de este trabajo. El mismo interactúa con el módulo administrador a través de la interfaz nsupdate y la utilización de llaves para la autenticación. Ante la ocurrencia de un evento, el módulo de monitoreo notifica al módulo administrador sobre el cambio. Este último tomará la decisión de actualizar o no los recursos de DNS con el fin de que, a través del sistema de resolución de nombres, se pueda encaminar el tráfico de red hacia otro destino que sea accesible por los clientes.
- **Administrador:** Tiene como función principal la toma de decisiones de convergencia. Es directamente invocado por el módulo de monitoreo ante la ocurrencia de un evento sobre un servicio monitoreado. Este módulo es responsable de recabar la información necesaria e

interactuar con el servicio DNS para la modificación de registros de resolución. Adicionalmente registra su actividad con el fin de que la misma pueda ser consultada.

- **Interfaz gráfica:** Se trata de una interfaz basada en la web que utiliza como transporte el protocolo HTTP. Es a través de la cual se accede a la funcionalidad de registro de servicios, zonas de DNS y estado de los chequeos. Es posible accederla utilizando un navegador web, previa validación del acceso a través de usuario y contraseña. Con el fin de desacoplar la lógica de programación del modelo de visualización, se utilizó el motor de plantillas Smarty[16] para PHP. El mismo permite separar el código PHP que contiene la lógica de la aplicación, de las plantillas HTML que permiten la visualización. Toda la funcionalidad de la aplicación se encuentra encapsulada y dividida en clases de PHP, las cuáles son instanciadas a partir de objetos que se crean bajo demanda. Para la visualización se utilizó el framework Twitter Bootstrap[17] que permite diseñar plantillas, formularios y menús de navegación. La utilización de este framework permitió simplificar la tarea de diseño y validación del sitio de administración, así como también garantizar y estandarizar el acceso desde distintos navegadores y dispositivos, lo que hace que la aplicación sea más accesible. Por último y con el fin de brindar estadísticas y reportes gráficos, se utilizó la herramienta Google Charts[18] que permite crear distintos tipos de gráficos, en base a los datos que envíe el usuario.
- **Resolución de nombres secundaria:** Se trata de un servidor DNS configurado bajo el rol de secundario o esclavo. El mismo recibirá del servidor primario las actualizaciones de zonas que se realicen y responderá a los clientes con los datos solicitados. Esta funcionalidad permite que ante la caída del servidor DNS principal o el vínculo de comunicación que permite acceder al mismo, este segundo servidor pueda responder los requerimientos de los usuarios y direccionar el tráfico de red hacia el punto que se encuentre establecido por los registros de resolución.

5.3 Proceso de instalación y configuración de componentes

La presente sección del trabajo tiene como propósito documentar la tarea realizada en lo que se refiere a instalación y configuración de componentes. Si bien el producto final se verá reflejado en una máquina virtual, la documentación que a continuación se realizará permitirá que la solución pueda instalarse bajo otro sistema operativo u otra versión de componentes realizando mínimas modificaciones.

[16] Motor de plantillas bajo licencia GPL: http://www.smarty.net
[17] Framework para desarrollo de plantillas web: http://getbootstrap.com/
[18] Google Charts: https://developers.google.com/chart/

5.3.1 Sistema Operativo y entorno base

Con el fin de brindar portabilidad a la solución y que la misma pueda ser adaptada a cualquier infraestructura, se optó por la utilización de un entorno de virtualización basado en Virtual Box[19]. El mismo permite crear una máquina virtual que trabaje independientemente del sistema operativo base (Windows, GNU/Linux), e inclusive de la solución final de virtualización. Con este esquema, la máquina virtual puede ser portada y adaptada a una instalación que utilice software comercial de virtualización como por ejemplo VMware[20] o Citrix[21] corriendo sobre cualquier plataforma. También permite que la misma pueda ser instalada utilizando un hipervisor libre como por ejemplo VirtualBox de la empresa Oracle.

La máquina virtual ha sido configurada para que contenga un adaptador de red, 2 procesadores virtuales y 1 GiB de memoria RAM. Asimismo, fue creada para una arquitectura de 64 bits, con el fin de poder escalar recursos y aprovechar al máximo las características de dicha arquitectura. Dependiendo de la cantidad de recursos que gestione la solución, puede resultar necesario ampliar las características de hardware de la máquina virtual. Bajo esta configuración, la solución ha sido probada con la gestión de 100 (cien) recursos a los cuáles se les realiza monitoreo a intervalos promedio de 1 minuto. Al momento de crear el disco virtual donde se almacenarán los datos, se optó por crear un disco con una capacidad de 100 GiB (Gibibytes) bajo la configuración de aprovisionamiento delgado o liviano; lo que significa que el disco irá creciendo a medida que sea necesario almacenar datos en el mismo.

El sistema operativo que utiliza la máquina virtual está basado en GNU/Linux y se optó por utilizar la distribución Debian[22], debido a que es en la que el autor del presente trabajo posee más experiencia de administración. Esta distribución provee un sistema de instalación de software por paquetes sencillo y ágil para utilizar y gestionar. Además, todo el software utilizado se encuentra paquetizado, con lo cual no es necesario realizar compilaciones manuales del software que forma parte de la solución. Otro factor importante al momento de decidir la distribución de GNU/Linux a utilizar, fue la de valerse de las actualizaciones de software a los paquetes que componen la misma. Con el fin de realizar una instalación mínima del sistema operativo que contenga sólo el software necesario, se optó por la opción conocida como "netinst" que provee un conjunto de software base para que el sistema operativo pueda funcionar y delega en el usuario la instalación del software adicional. La imagen del sistema operativo fue descargada desde el sitio web del fabricante, en la siguiente dirección:

https://www.debian.org/distrib/netinst

[19] Virtual Box: www.virtualbox.org
[20] VMware: www.vmware.com
[21] Cistrix: www.citrix.com
[22] Debian: www.debian.org

Una vez descargada la imagen se procedió a crear la máquina virtual y realizar la instalación del sistema operativo utilizando las configuraciones que mayormente vienen por defecto. Se trata del sistema operativo base, con el conjunto de utilidades estándar. Luego de finalizar la instalación, y por una cuestión de administración y acceso a la máquina se procedió con la instalación del siguiente software a través de paquetes, de modo tal que el sistema pueda ser actualizado de manera sencilla:

Nombre del Paquete	**Descripción**
openssh-server	Permite realizar conexiones administrativas a la línea de comandos del servidor GNU/Linux que aloja la aplicación
mc	Midnight Commander. Conjunto de utilitarios para la manipulación de archivos. Incluye el editor de textos mcedit
tcpdump	Permite realizar capturas de tráfico de red
ntp	Provee servidor de hora y utilidades cliente

Cuadro 5.3.1.1: Software instalado en el sistema base

Las credenciales de usuario utilizadas para la instalación del sistema operativo son:

Nombre de usuario	**Contraseña**
root	trafficManager

Cuadro 5.3.1.2: Credenciales de usuario del Sistema Operativo Utilizado

5.3.2 Módulo de monitoreo

Como se mencionó anteriormente, el módulo de monitoreo se corresponde con el software Icinga, en su versión 2.0. Para la instalación de dicha aplicación, se siguió la documentación publicada en el sitio oficial del desarrollador en:

http://docs.icinga.org/icinga2/latest/doc/module/icinga2/toc

Al momento de realizar la instalación, el desarrollador ofrece diversos métodos. Con el fin de valerse de las actualizaciones que se puedan publicar a futuro, se optó por realizar la instalación del

software usando los paquetes publicados en el repositorio Debmon[23]. Para ello, en la línea de comandos se ejecutaron los siguientes comandos para instalar y actualizar el repositorio:

```
1 # wget -O - http://debmon.org/debmon/repo.key 2>/dev/null | apt-key add -

2 # echo 'deb http://debmon.org/debmon debmon-wheezy main' > /etc/apt/sources.list.d/debmon.list

3 # apt-get update
```

Cuadro 5.3.2.1: Ejecución de comandos previos a la instalación de Icinga

Descripción de comandos:

1. Obtiene e instala en el sistema la llave que se utiliza para verificar la firma de los paquetes del repositorio Debmon
2. Agrega el repositorio Debmon a la lista de repositorios de software que utiliza el sistema operativo
3. Actualiza la lista de paquetes que se pueden instalar en el sistema operativo en base a los repositorios configurados.

Por último, se ejecutaron los siguientes comandos para realizar la instalación del software de monitoreo. El primero se corresponde con la instalación de Icinga, mientras que el segundo instala los plugins que utilizará Icinga para realizar los monitoreos programados. De estos últimos se seleccionará un conjunto para integrar a la herramienta "Traffic Manager".

```
# apt-get install icinga2

# apt-get install nagios-plugins
```

Cuadro 5.3.2.2: Ejecución de comandos para la instalación de Icinga

La responsabilidad de realizar el monitoreo corresponde al software que se instala con el paquete icinga2. El mismo provee la lógica de programación de chequeos y notificaciones. En versiones anteriores también incluía una interfaz gráfica para acceder a los resultados de monitoreo, conocida como "Classic UI" (Classic User Interface). A partir de la versión 2 del software se optó por desacoplar la lógica de monitoreo de la lógica de visualización. Con el fin de poder acceder a las estructuras internas de monitoreo, se instala la interfaz "Icinga Web 2", que provee una interfaz visual basada en HTML5 y otra basada en el paradigma REST[24], que permite exportar los datos utilizando el formato JSON o XML. Esta interfaz, requiere la

[23] Debmon: Debian Monitoring Project. Repositorio que unifica paquetes de software relacionados a Nagios e Icinga sobre sistemas Debian.

[24] REST: Información acerca de este paradigma puede ser encontrada en el **Anexo A** del presente trabajo

instalación del motor de base de datos MySQL, que también será utilizado por la aplicación "Traffic Manager" para almacenar sus datos de configuración. La ventaja de instalar todo el software bajo la opción de paquetes Debian, permite que el sistema determine las dependencias de software e instale el software adicional para poder ejecutar las aplicaciones requeridas. Por ejemplo, al momento de instalar el paquete icinga-web, el sistema determinó que el paquete php5 no se encontraba instalado y propone la instalación del mismo. Los comandos que se ejecutaron con el fin de instalar y configurar el software mencionado son:

```
1 # apt-get install mysql-server mysql-client icinga2-ido-mysql

2 # icinga2 feature enable ido-mysql

3 # icinga2 feature enable command

4 # usermod -a -G nagios www-data

5 # service icinga2 restart

6 # apt-get install apache2 icinga2-classicui icinga-web icinga-web-config-icinga2-ido-mysql
```

Cuadro 5.3.2.3: Ejecución de comandos para la instalación del motor de base de datos y la interfaz Icinga Web

Descripción de comandos:

1. Instala el servidor y cliente de base de datos MySQL, así como también la funcionalidad en Icinga para que lea y almacene información de configuración en la base de datos instalada.
2. Habilita la función de lectura y almacenamiento de configuraciones en la base de datos
3. Habilita la función para que Icinga pueda interpretar comandos externos enviados por otras aplicaciones. Esta opción es útil, por ejemplo, para indicarle a icinga2 que recargue su configuración debido al agregado o eliminación de algún chequeo
4. Agrega al usuario del sistema operativo www-data (usuario bajo el cual corre el proceso del servidor web) al grupo de usuarios nagios. Esto permitirá que el usuario www-data pueda enviar comandos externos a icinga2, a través de la escritura de los mismos en el archivo **/var/run/icinga2/cmd/icinga2.cmd**
5. Reinicia el servicio Icinga para que el mismo arranque nuevamente con la configuración de base de datos realizada
6. Instala los paquetes de software indicados, dentro de los que se encuentran el servidor web Apache y las interfaces classic y web de Icinga.

Las tareas descriptas en la presente sección del trabajo permitieron realizar una instalación estándar del servicio de monitoreo Icinga2 junto con las interfaces gráficas "Classic UI" y "Web". Las mismas

podrán ser accedidas utilizando un navegador web y accediendo a las siguientes direcciones con sus correspondientes credenciales:

Interfaz	Dirección	Usuario	Contraseña
Classic	http://<ip o nombre del servidor>/icinga2-classicui/	icingaadmin	adminicinga
Web	http://<ip o nombre del servidor>/icinga-web/	root	adminicinga

Cuadro 5.3.2.4: Direcciones y credenciales de acceso a las interfaces Classic y Web de Icinga

Adicionalmente, y con el fin de que el módulo administrador y la interfaz web puedan acceder a la los resultados de monitoreo, se creó el siguiente usuario en la Interfaz web de Icinga, con su correspondiente clave de autenticación. La misma permite que los requerimientos utilizando el paradigma REST puedan ser autenticados y solo los usuarios autorizados puedan acceder a los datos:

Usuario	Contraseña	Clave de Autentucación (auth_key)
trafficmanager	managertraffic	d3OuOUFo5slDsfpDsXUXV15Z7SZ

Cuadro 5.3.2.5: Credenciales de acceso a la interfaz REST de Icinga

La configuración de chequeos sobre el servicio de monitoreo, se gestiona desde la interfaz gráfica de administración de la solución implementada. Previo a realizar la gestión a través de la interfaz web, fue necesario la creación de configuraciones básicas para el servicio de monitoreo. Icinga permite que las configuraciones puedan ser divididas en archivos, ubicados en el directorio **/etc/icinga2/conf.d/**. Sobre este directorio se pueden crear tantos archivos como se quieran, los cuales contienen configuraciones propias para el servicio de monitoreo. Para que la herramienta Traffic Manager pueda crear archivos de configuración que se correspondan con los servicios a monitorear, se modificaron los permisos del directorio mencionado con el fin de que el usuario que ejecuta el servicio web pueda escribir información. También se generaron 2 archivos con configuraciones iniciales que utiliza la herramienta:

- **commandsTrafficManager.conf:** Contiene la definición de los comandos que se utilizan en la herramienta Traffic Manager para realizar los chequeos. El agregado de nuevos chequeos deberá realizarse utilizando este archivo de configuración. A modo de ejemplo se transcriben algunas líneas de dicho archivo:

```
object CheckCommand "check-fast-alive" {

import "plugin-check-command"

command = "/usr/lib/nagios/plugins/check_fping -H '$address$'"

}

object CheckCommand "check_ftp" {

import "plugin-check-command"

command = "/usr/lib/nagios/plugins/check_ftp -H '$address$'"

}

object CheckCommand "check_http" {

import "plugin-check-command"

command = "/usr/lib/nagios/plugins/check_http -H '$address$'"

}

object CheckCommand "check_pop" {

import "plugin-check-command"

command = "/usr/lib/nagios/plugins/check_pop -H '$address$'"

}

object CheckCommand "check_smtp" {

import "plugin-check-command"

command = "/usr/lib/nagios/plugins/check_smtp -H '$address$'"

}
```

Cuadro 5.3.2.6: Comandos para monitoreo definidos en Icinga

- **eventCommandTrafficManager.conf:** Contiene la definición del comando que se utilizará en la herramienta para invocar al módulo administrador. En icinga se lo conoce como un "comando de evento", que es invocado cada vez que se produce un cambio en el estado de un chequeo. Su contenido es:

```
object EventCommand "notify_traffic_manager" {

import "plugin-event-command"

command = "/opt/trafficmanager/nagiosplugins/notify_traffic_manager.php $host.name$ $host.vars.domain$
$host.vars.dnsaddress$ $host.vars.failoverdnsaddress$ $host.state$ $host.state_type$
$host.vars.isfailoverhost$ $host.last_state$ "

}
```

Cuadro 5.3.2.7: Comando de Evento (EventCommand) definidos en Icinga

5.3.3 Módulo de resolución de nombres

Para la implementación del módulo de resolución de nombres, se instaló el software BIND desarrollado por el InterNet Software Consortium. BIND (Berkeley Internet Domain Name) es una aplicación de código abierto que implementa el protocolo de resolución de nombres DNS. Se ha elegido esta implementación, debido a que es una de las más estables y más utilizadas en InterNet. Adicionalmente corre sobre el mismo sistema operativo en el que se implementa la solución.

La solución completa utiliza 2 (dos) servidores DNS con el fin de mantener la alta disponibilidad. La instalación del sistema operativo y el software en ambos casos se realizó siguiendo el mismo procedimiento. La diferencia entre ambos servidores reside en la configuración del módulo de resolución de nombres (software BIND). El servidor de nombres instalado en el mismo equipo en el que corre el software de monitoreo y la interfaz de administración, adoptará el rol de servidor de nombres primario; y es en el cual se realizarán las modificaciones de zonas, a través de la herramienta nsupdate. El segundo servidor de nombres adoptará el rol de esclavo o secundario. Cada vez que se dé de alta una zona en el servidor primario, la misma deberá ser configurada en el segundo servidor con el fin de que la información referida a los registros de resolución pueda ser actualizada desde el primer servidor a través del concepto de transferencia de zonas descripto en la RFC 1035. Al momento de realizar la instalación del software, se optó como en los casos anteriores por la opción paquetizada que provee Debian. En ambos servidores se ejecutó el siguiente comando:

```
# apt-get install bind9
```

Cuadro 5.3.3.1: Instalación de software de resolución de nombres BIND

Con el fin de que el servidor DNS primario pueda ser actualizado sin necesidad de modificar manualmente sus archivos de zonas, se utilizó la herramienta **nsupdate**, la cual ha sido explicada en

capítulos anteriores. A través de la utilización de dicha herramienta, los módulos interfaz web y administrador pueden interactuar con el módulo de resolución de nombres, invocando comandos estándares y bien documentados en la RFC 2136. Para establecer un mecanismo de autenticación, de modo tal que las actualizaciones puedan realizarse de modo seguro se utilizaron llaves creadas con la herramienta **dnssec-keygen**. Para la implementación se utilizaron llaves de 512 bits generadas con el algoritmo HMAC-MD5. Para ello, se ejecutaron los siguientes comandos:

```
# cd /etc/bind

# dnssec-keygen -a HMAC-MD5 -b 512 -n HOST trafficManager
```

Cuadro 5.3.3.2: Generación de llaves con la herramienta **dnssec-keygen**

El parámetro **–a** determina el algoritmo criptográfico que se utiliza para la generación de la llave. El parámetro **–b** indica la cantidad de bits que serán utilizados para crear la clave y por último el parámetro **–n** indica quién será el dueño de la clave. En este caso se trata de un equipo (HOST) identificado como **trafficManager**. El resultado de la ejecución de dicho comando será reflejado en la creación de 2 archivos dentro del directorio **/etc/bind**, tal como lo muestra la siguiente figura:

```
-rw------- 1 root bind  123 mar 17 15:47 Ktrafficmanager.+157+01187.key

-rw------- 1 root bind  229 mar 17 15:47 Ktrafficmanager.+157+01187.private
```

Cuadro 5.3.3.3: Llaves creadas con la herramienta **dnssec-keygen**

El archivo **Ktrafficmanager.+157+01187.key** corresponde a la clave pública que será utilizada para validar las zonas de DNS. Para el caso de la autenticación, la clave que se utilizará es la privada, que se encuentra contenida en el archivo **Ktrafficmanager.+157+01187.private**. Dentro del archivo de configuración del servidor DNS, **named.conf** que se encuentra en el directorio **/etc/bind**/, se deberá crear la sección key con el nombre que se utilizó para identificar al host al ejecutar el comando **dnssec-keygen**, y con la información de la clave generada, tal como se muestra en el cuadro 5.3.3.4:

```
key "trafficManager" {
   algorithm HMAC-MD5;
   secret " FdrDDkrD/2Cel5c8IJFdEd9N71mvXY6lH3slqe7PPvpXme70XM2V3rclvbBdIwvvv2wnrLkKUzhQCflJ4UBz1Q== ";
};
```

Cuadro 5.3.3.4: Definición de claves en servicio DNS para el servidor BIND

Por último, para cada zona que sea gestionada a través de la herramienta Traffic Manager, es importante que al momento de su definición se haga referencia a través del comando **allow-update** a la llave creada. El cuadro 5.3.3.5 muestra un ejemplo de configuración que debería utilizarse para las zonas gestionadas:

```
zone "ejemplo.com.ar" in {
type master;
file "/etc/bind/zonas/ejemplo.com.ar.conf";
allow-update  { key trafficmanager };
};
```

Cuadro 5.3.3.5: Definición de zona de DNS en el servidor BIND gestionada por la herramienta Traffic Manager

Con el fin de que el servidor primario pueda intercambiar información con el secundario, se debe definir un mecanismo de autorización entre ambos. El método que comúnmente se utiliza, consisten en habilitar la dirección IP del servidor secundario en el primario, para que el mismo pueda realizar transferencias de zonas. Este método resulta poco seguro, ya que permite que la autenticación se realice únicamente con direcciones IP, lo cual puede ser vulnerado por un atacante de modo sencillo. El mejor mecanismo reside en la utilización de llaves generadas con dnssec-keygen; y debido a que es el mecanismo que ya se viene utilizando para la actualización de registros en el DNS primario, es el que se opta para autenticar la transferencia de información entre ambos servidores. Para realizar la configuración en el DNS primario, se agregó la sentencia allow-transfer en la definición de zona, modificando lo expuesto en el cuadro **5.3.3.5**. La definición completa de la zona se muestra en el siguiente cuadro:

```
zone "ejemplo.com.ar" in {
type master;
file "/etc/bind/zonas/ejemplo.com.ar.conf";
allow-update  { key trafficmanager };
allow-transfer  { key trafficmanager };
}
```

Cuadro 5.3.3.6: Definición zona de DNS en el servidor BIND con comando **allow-transfer**

En el servidor DNS secundario será necesario definir la configuración de la llave que se utilizará para realizar transferencias de zonas contra el primario. Esta configuración comprende la definición de la llave a utilizar a través del a sentencia **key** y el servidor sobre el cual se utilizará esa llave a través de la sentencia **server**. Ambas se pueden realizar en el archivo de configuración **named.conf** ubicado dentro del directorio **/etc/bind/**, tal como se muestra en el cuadro 5.3.3.7 (notar que la dirección IP 192.168.1.1 denota al servidor DNS primario):

```
key "trafficmanager" {

algorithm HMAC-MD5;

secret "FdrDDkrD/2Cel5c8IJFdEd9N71mvXY6lH3slqe7PPvpXme70XM2V3rclvbBdIwvvv2wnrLkKUzhQCflJ4UBz1Q==";

};

server 192.168.1.1 {

keys { trafficmanager; };

};
```

Cuadro 5.3.3.7: Definición de clave y asociación al servidor primario en el servidor DNS secundario

Por último, para cada zona en la que se defina que el servidor secundario tiene que actuar como esclavo, se deberá agregar la siguiente configuración para dar de alta la zona en el mismo e indicar que el mismo está autorizado a dar respuestas autoritativas:

```
zone "ejemplo.com.ar" in {
type slave;
file "/etc/bind/zonas/ejemplo.com.ar.sec";
masters { 192.168.1.1; };
};
```

Cuadro 5.3.3.8: Definición de zona de DNS en servidor secundario

Con las configuraciones mencionadas, ambos servidores podrán intercambiar información acerca de las definiciones realizadas en las zonas de resolución. De este modo, cada vez que se produzca un cambio en la zona de DNS del servidor primario, el mismo notificará al servidor secundario sobre el cambio, para que este último actualice su configuración de resolución.

5.3.4 Módulo administrador

El módulo administrador toma las decisiones de convergencia en base a la información provista por el módulo de monitoreo y la configuración realizada desde la interfaz gráfica de administración. Con el fin de unificar el lenguaje de programación utilizado en toda la solución y reutilizar código, se optó por programar el módulo administrador en lenguaje PHP. De este modo, es posible utilizar las mismas clases programadas para la interacción con la base de datos y la configuración del módulo de monitoreo desde el módulo administrador y la interfaz gráfica.

Cada vez que se da de alta un servicio para ser gestionado por la herramienta Traffic Manager, el módulo de monitoreo comienza a realizar chequeos sobre dicho servicio y la dirección IP secundaria de monitoreo. Ante un cambio en el resultado del chequeo del servicio, el módulo de monitoreo invoca al módulo de administración con la información correspondiente al servicio y su dirección IP de failover. Es en este punto donde la lógica del módulo de monitoreo determinará si es necesario disparar la convergencia cambiando los registros de resolución del servicio DNS (tarea que se realiza invocando al módulo de resolución), o bien no es necesario realizar ninguna tarea.

El módulo de monitoreo invoca al módulo administrador utilizando el concepto conocido en Icinga 2 como **EventCommand**. El mismo define un comando que se ejecuta al momento en que se detecte un cambio en el resultado del chequeo realizado a un servicio u objeto del tipo host. Este último comando se corresponde con la aplicación PHP definida que contiene la lógica para determinar si es necesario disparar la convergencia o no. Un **EventCommand** en Icinga puede recibir diversos datos como parámetro, los cuáles son accedidos desde el programa invocado. La definición del comando de evento definido, se muestra en el cuadro 5.3.2.7 de la sección 5.3.2 el presente capitulo. Los parámetros que recibe el módulo administrador por parte del módulo de monitoreo son:

- **$host.name$:** Contiene el nombre de DNS (nombre de host y dominio) del servicio que provoca el evento.
- **$host.vars.domain$:** Contiene el nombre de dominio al cual pertenece el servicio que provoca el evento
- **$host.vars.dnsaddress$:** Contiene la dirección IP a la que se debe hacer failover en el servicio de DNS. Notar que esta dirección solo se utiliza a los efectos de realizar la resolución de nombres. La dirección a la que se realizan los chequeos del servicio configurado se encuentra representada por la variable **$host.address$**.
- **$host.vars.failoverdnsaddress$:** Contiene la dirección IP a la que se deberá hacer failover en el servicio de DNS. Del mismo modo que ocurre con la variable anterior, solo se utiliza a los efectos de resolución de nombres.
- **$host.state$:** Contiene el estado en el que se encuentra el servicio al momento de invocar al comando. El estado puede ser UP O DOWN, dependiendo si el servicio se encuentra operando normalmente, o está presentando inconvenientes que hacen que el mismo no pueda atender a los requerimientos de los usuarios.
- **$host.state_type$:** Cuando el módulo de monitoreo detecta un problema en un chequeo, el mismo realiza nuevos chequeos para re verificar el estado real del servicio. Cuando se confirma que el servicio no se puede alcanzar, el valor de esta variable será HARD. Previo a este estado, esta variable contendrá la palabra SOFT.

- **$host.vars.isfailoverhost$:** Si el host con el que se invoca al módulo administrador se corresponde al primario, el valor de esta variable será 0 (cero), caso contrario contendrá el valor 1 (uno). El módulo de monitoreo realiza chequeos a la IP primaria y a la IP de failover de cada servicio gestionado. Cualquiera de ellas podría producir un evento de monitoreo e invocar al módulo administrador.
- **$host.last_state$:** Contiene el estado anterior en el que se encontraba un servicio. Con este valor, el módulo administrador se asegura de disparar la convergencia cuando realmente se produzca un cambio en un servicio.

Cada vez que el módulo de monitoreo detecte un cambio en un objeto del tipo host, ejecutará desde la línea de comandos el archivo **notify_traffic_manager.php** que se encuentra ubicado en el directorio **/opt/trafficmanager/nagiosplugins/** con los parámetros indicados anteriormente. El mismo determinará la necesidad de realizar la convergencia en base a la lógica programada. Adicionalmente registrará su comportamiento de modo tal que pueda ser consultado a través de la interfaz gráfica.

5.3.5 Módulo de Interfaz gráfica

El módulo de Interfaz Gráfica, representa el punto de partida para comenzar a operar y utilizar la aplicación. Es a través del cual se registran las zonas de DNS que se van a gestionar y se dan de alta los dispositivos. La misma permite definir los chequeos que se realizan sobre los equipos administrados y provee una interfaz de visualización de logs y eventos.

Con el fin de respetar los estándares actuales de programación, se optó por realizar un desarrollo modular en capas. Este esquema de desarrollo permite que las modificaciones y actualizaciones a la aplicación puedan realizarse de modo más sencillo. Para utilizar este esquema de desarrollo, se optó por la utilización del motor de plantillas Smarty. Esta herramienta facilita la manera de separar la aplicación lógica y el contenido en la presentación. La interfaz que se muestra a los usuarios, es codificada utilizando el concepto de plantilla. Cada plantilla contiene el código HTML y el conjunto de validaciones que se desea realizar.

Todo el código de programación del módulo interfaz gráfica fue ubicado dentro del directorio raíz del servidor web, que se encuentra en **/var/www**/ del servidor Traffic Manager. Los archivos han sido separados en diversos directorios con el fin de agrupar su funcionalidad. Dentro de los más importantes, se destacan:

- **bootstrap:** Contiene los scripts y las hojas de estilo utilizados en la aplicación que permiten formatear los datos que se presentan a los usuarios.
- **configs:** Contiene el archivo de configuración global de la aplicación. En el mismo se definen las credenciales de acceso a la base de datos y las llaves que se deben utilizar para interactuar con el módulo de resolución de nombres y de monitoreo.
- **libs:** Contiene las librerías utilizadas con la lógica de la aplicación. Las mismas permiten interactuar con la base de datos, el servicio DNS y el módulo de monitoreo.
- **modules:** En este directorio se encuentran cada uno de los módulos de funcionalidad que componen la interfaz de administración. Cada módulo está ordenado en un subdirectorio independiente. La información que se encuentra en cada subdirectorio es similar en todos los casos. La lógica codificada en cada caso permite realizar la interacción con el modelo de datos (invocación de clases PHP generalmente representadas en los archivos ubicados en el directorio **libs** del punto anterior) y la interfaz gráfica. Cada vez que es invocada la funcionalidad desde la herramienta, se consultan los datos necesarios para presentar en la interfaz de usuario y se instancian las plantillas de Smarty que proveen la visualización y el formato final de los datos. Las módulos implementados son:
    - **dashboard:** Provee información general acerca del estado de la infraestructura en una única pantalla. Para visualizar información estadística en formato gráfico se utilizó le herramienta Google Charts. Se trata de una herramienta gratuita provista por le empresa Google que permite realizar distintos tipos de gráficas invocando a los servidores de Google.
    - **dns:** Permite realizar operaciones sobre las zonas de DNS en el servidor primario y provee información acerca de la configuración a realizar en los servidores secundarios.
    - **host:** Permite manipular los objetos que gestiona la aplicación.
    - **log:** Provee información acerca de los cambios de estado de los equipos gestionados y auditoría de usuarios.
    - **user:** Permite definir los usuarios que podrán utilizar el sistema.
- **templates:** Contiene las plantillas que se utilizan para la visualización de datos. En este directorio se encuentran una serie de archivos con terminación **.tpl** en su nombre que proveen el código HTML que será mostrado al usuario. Dentro del código se incrustan las variables que deben ser reemplazadas al momento de instanciar la plantilla por los datos que deba presentar la aplicación. Adicionalmente en este directorio se encuentran las plantillas que son utilizadas para escribir los archivos de configuración del servidor DNS y el servidor de monitoreo. La idea de utilizar plantillas para la escritura permite que ante un cambio en el formato de archivo de configuración, pueda modificarse únicamente la plantilla para adaptar el sistema.

La aplicación utiliza como soporte para su funcionamiento el motor de base de datos MySQL, en el cual almacena la configuración de sus objetos e información de logueo. Al momento de optar por un motor de base de datos a utilizar se eligió MySQL, debido a que es el que utiliza Iginga (módulo de monitoreo). La unificación de los motores de base de datos, permite que la toma de backups y mantenimiento de la solución pueda realizarse de modo más sencillo. El esquema de base de datos utilizado se muestra en la figura **5.3.5.1**:

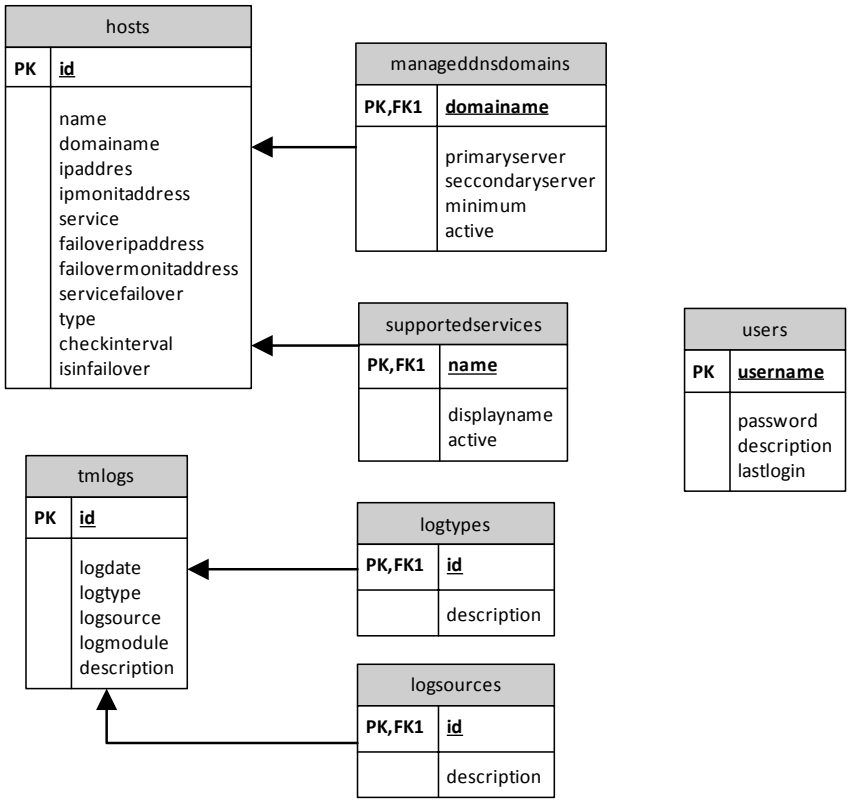


Figura 5.3.5.1: Esquema de base de datos utilizado por la herramienta Traffic Manager

Toda la funcionalidad de acceso a la base de datos fue encapsulada en una clase PHP que permite la interacción con la misma. Dicha funcionalidad ha sido implementada en el archivo **database.class.php** que se encuentra ubicado en el subdirectorio **libs**. Esto permite que en caso de ser necesaria la utilización de otro motor de base de datos, se pueda adaptar la funcionalidad de la aplicación simplemente modificando este último archivo con las funciones propias del nuevo motor.

5.3.6 Funcionalidad de la Interfaz Gráfica

Las pruebas funcionales de la aplicación implementada fueron realizadas con el navegador Google Chrome bajo el Sistema Operativo Windows 8. Debido a que se utilizaron estándares de desarrollo que permiten compatibilizar el acceso desde diversos navegadores, la aplicación debería funcionar correctamente con cualquier navegador y bajo cualquier sistema operativo. Otra particularidad de la aplicación web es que puede ser accedida desde diversos dispositivos como por ejemplo tabletas y teléfonos inteligentes, adaptándose la interfaz al tamaño de pantalla en la que se está mostrando. La misma ha sido probada sobre un dispositivo Iphone 6 y un navegador Safari.

5.3.6.1 Interfaz de Login

La misma permite realizar la validación del usuario que accederá al sistema. Se ha creado un usuario inicial para el acceso a la interfaz, cuyos datos son:

- Usuario: admin
- Contraseña: Passw0rd

Figura 5.3.6.1.1: Pantalla de Login de la herramienta Traffic Manager

5.3.6.2 Dashboard Inicial (Estado General)

La siguiente figura muestra la primera pantalla que puede visualizar un usuario cuando inicia sesión en el sistema. Sobre la parte izquierda de la pantalla, se puede visualizar el menú de operaciones permitidas que estará presente en todas las pantallas que utilice el usuario. El mismo muestra el nombre del usuario

logueado y la fecha y hora del ultimo acceso del mismo al sistema. Sobre el centro y la parte derecha de la pantalla se puede visualizar la funcionalidad accedida. El dashboard inicial provee de una vista al usuario acerca del estado general del sistema. Los cuadros que se pueden visualizar son:

- **Estado del Sistema:** Muestra en formato gráfico la utilización de Memoria y CPU del servidor Traffic Manager al momento de cargar esta pantalla
- **Estado de los servicios:** Permite monitorear las componentes que forman parte de la solución con el fin de evaluar posibles fallas.
- **Estadística de Hosts:** Muestra en formato gráfico el estado de los dispositivos gestionados y sus servicios para realizar el failover.
- **Estado de los hosts:** Permite identificar en formato de tabla el estado de cada dispositivo gestionado, la fecha y hora en la que se realizó el último chequeo informado por el módulo de monitoreo; y en qué momento se realizará el próximo chequeo.

Figura 5.3.6.2.1: Pantalla Dashboard Inicia (estado general) de la herramienta Traffic Manager

5.3.6.3 Agregar Equipo

Esta opción permite agregar un dispositivo al sistema para que el mismo pueda ser gestionado a través de la aplicación. Como se puede observar, la interfaz realiza la validación de los datos a medida que el usuario los ingresa. Los campos que se deben completar son:

- **Nombre:** Corresponde al nombre del host, dentro de la zona de DNS. Debe respetar el juego de caracteres establecidos en la RFC 1035 y una longitud máxima de 63 caracteres.

- **Dominio:** Se debe seleccionar de la lista de dominios gestionados por la aplicación. Sumado al campo nombre, conformará el FQDN del dispositivo administrado.
- **Tipo de registro de DNS:** Define el modo en el que será dado de alta el registro en el servidor DNS. Las opciones actualmente soportadas son:
    - **A:** Tipo de registro que se utiliza para traducir un nombre a una dirección IPv4.
    - **MX prioridad 10:** Tipo de registro que define un servidor de intercambio de correo para el dominio con prioridad 10, de acuerdo a lo explicado en el capítulo 3 del presente trabajo.
    - **MX prioridad 20:** Tipo de registro que define un servidor de intercambio de correo para el dominio con prioridad 20, de acuerdo a lo explicado en el capítulo 3 del presente trabajo.
- **IP primaria para DNS:** Es la dirección IP que será utilizada para brindar el servicio gestionado. Esta dirección se utiliza para la resolución de nombres
- **IP primaria para monitoreo:** Es la dirección que será utilizada para realizar el monitoreo al servicio gestionado a través del enlace primario de InterNet. Esta dirección puede coincidir con la anterior, o bien ser distinta en caso de querer realizar el monitoreo a otro equipo de red.
- **Servicio para monitoreo IP primaria:** Permite seleccionar el servicio a través del cual se realizará el chequeo a la IP primaria para monitoreo. En caso de detectarse una falla en este servicio, se gestionará la convergencia a la dirección IP de failover.
- **IP de failover para DNS:** Es la dirección IP que será utilizada para resolver el nombre del host en caso de detectarse una falla en el servicio primario.
- **IP de failover para monitoreo:** Es la dirección que será utilizada para realizar el monitoreo al servicio gestionado a través de la IP de failover.
- **Servicio para monitoreo IP de Failover:** Permite seleccionar el servicio a través del cual se realizará el chequeo a la IP para monitoreo de failover.
- **Intervalo entre chequeos:** Determina el tiempo que transcurre entre cada chequeo que se realiza al dispositivo gestionado. Cuánto menor es el intervalo, más rápido se realizará la convergencia.

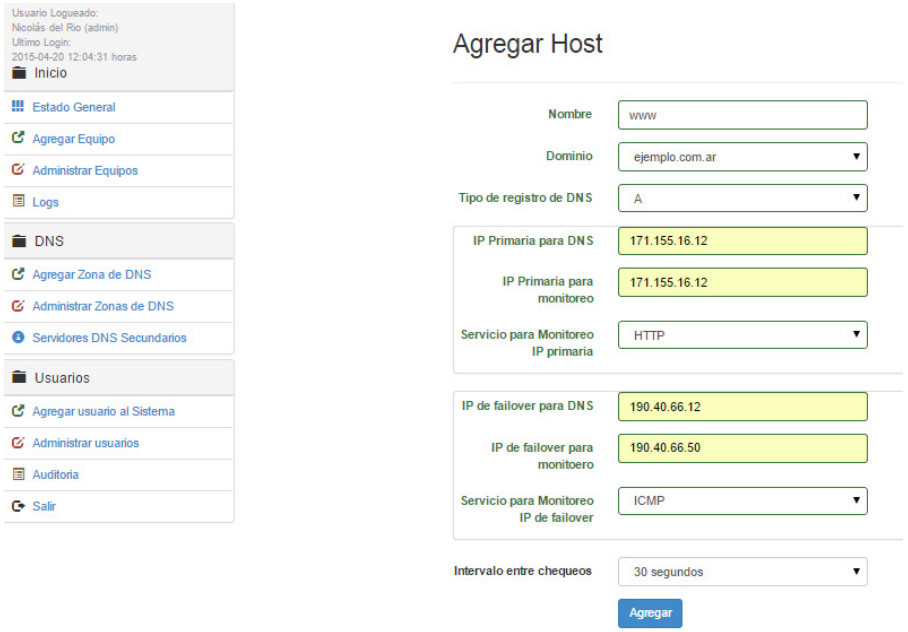


Figura 5.3.6.3.1: Pantalla para Agregar Host en la herramienta Traffic Manager

5.3.6.4 Administrar Equipos

Esta opción muestra los dispositivos administrados por la herramienta. Permite acceder a la interfaz de edición de atributos (similar a la interfaz para agregar host) y eliminar un dispositivo:

Figura 5.3.6.4.1: Pantalla de Administración de Hosts de la herramienta Traffic Manager

5.3.6.5 Logs

A través de esta opción se puede acceder al registro de eventos del módulo de monitoreo. Esta interfaz permite determinar los eventos ocurridos sobre los dispositivos administrados con un nivel de detalle acerca de la falla registrada por el módulo de monitoreo y el momento de su ocurrencia. También permite realizar un filtro por rango de fechas y horas para acotar las búsquedas:

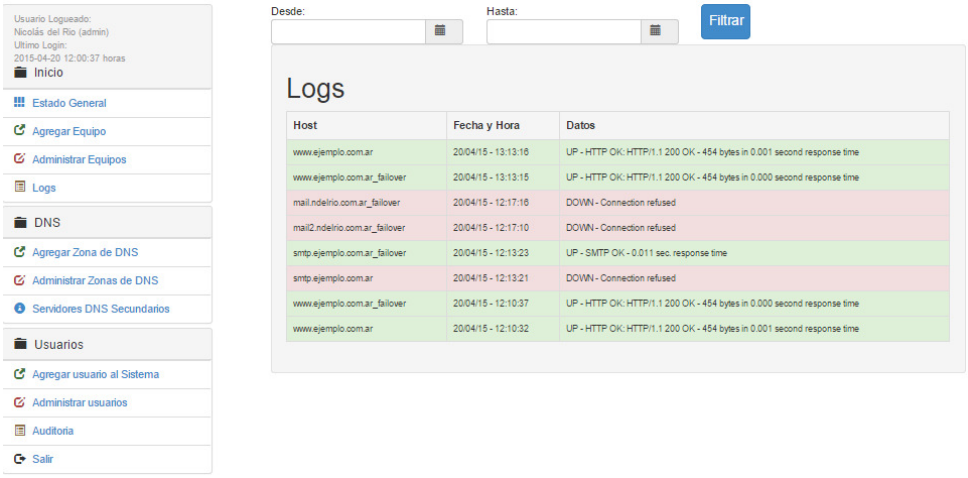


Figura 5.3.6.5.1: Pantalla de visualización de Logs de la herramienta Traffic Manager

5.3.6.6 DNS

En esta sección se podrá acceder a la funcionalidad referida al módulo de resolución de nombres. La lógica de las interfaces es similar a las de agregado y gestión de hosts, razón por la cual no se explicarán detalladamente las opciones sino que se presentará una figura abreviada con la información más relevante de cada pantalla. La opción Servidores DNS secundarios, provee un instructivo acerca de cómo deben ser configurados los servidores DNS esclavos con el fin de que puedan interactuar a través de transferencias de zonas con el servidor primario que es donde reside la función principal de la herramienta Traffic Manager. Las configuraciones indicadas en el instructivo, permitirán realizar la configuración sobre cualquier servidor BIND, aunque utilizando los datos indicados, podría adaptarse la configuración para cualquier otro tipo de servidor DNS que sea compatible con la RFC 1035.

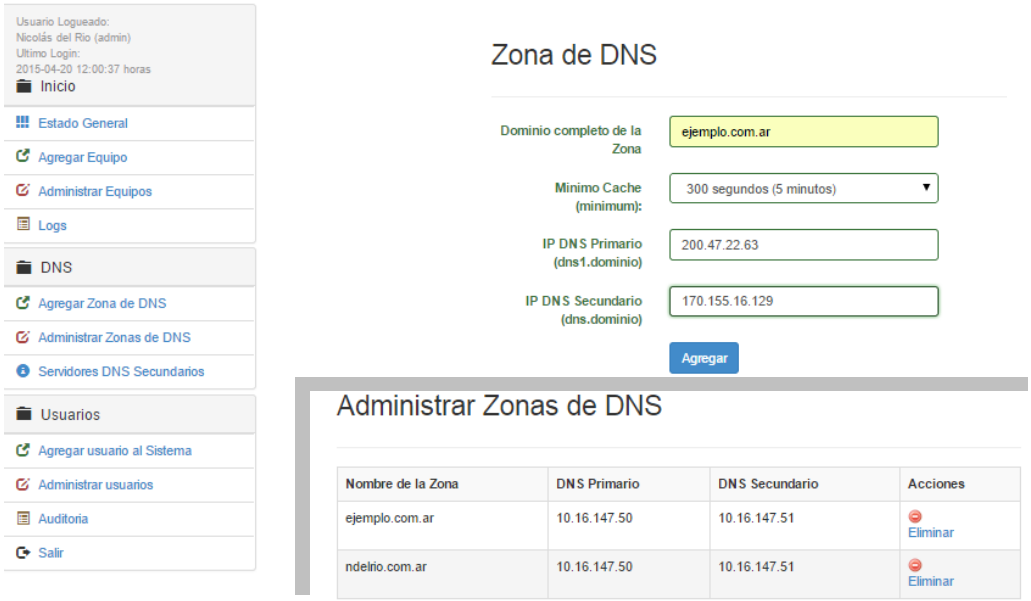


Figura 5.3.6.6.1: Pantallas para la gestión de zonas de DNS en la herramienta Traffic Manager

5.3.6.7 Usuarios

Siguiendo el mismo criterio utilizado con el servicio DNS, el menú usuarios permite realizar acciones sobre los usuarios que tienen permitido acceder al sistema para operar su funcionalidad. Le herramienta permite dar de alta usuarios, así como también editar sus datos personales, contraseña y eliminarlos del sistema. También provee un registro de auditoría, que permite visualizar las acciones realizadas por cada usuario en el sistema, así como las acciones automáticas que realiza la herramienta ante la ocurrencia de distintos eventos. Esta funcionalidad es de vital importancia al momento de verificar el funcionamiento del sistema y de las operaciones realizadas por la misma. Las siguientes figuras muestran las pantallas para agregar un usuario al sistema y gestionarlo; así como también la auditoría obtenida de los últimos eventos:

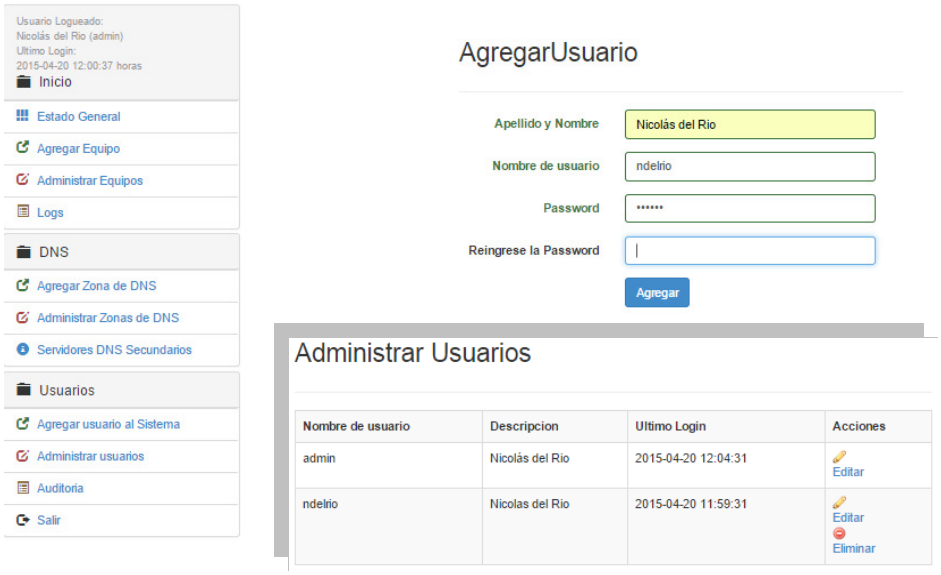


Figura 5.3.6.7.1: Pantallas para la gestión de usuarios en la herramienta Traffic Manager

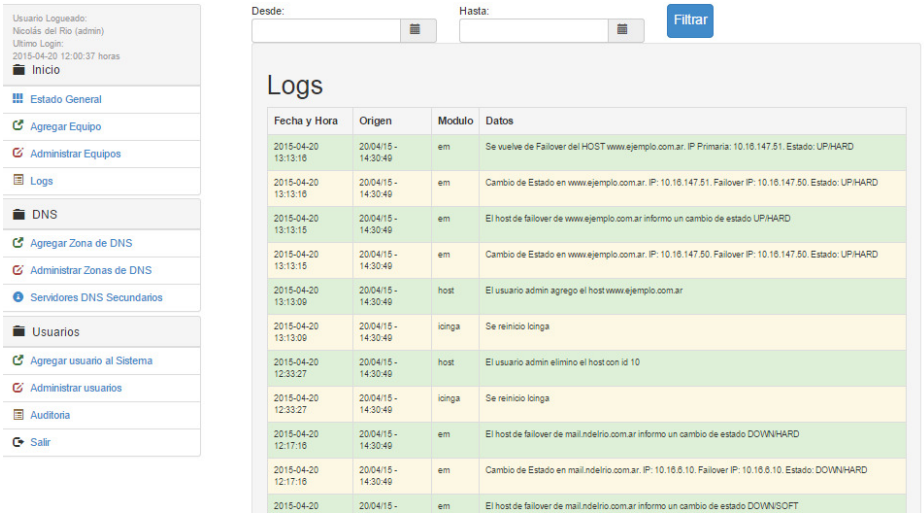


Figura 5.3.6.7.2: Pantallas para la visualización de Logs de Auditoría en la herramienta Traffic Manager

5.3.7 Interacción de componentes

Como se indicó anteriormente, la solución ha sido desarrollada bajo un esquema modular, de modo tal que permita su escalabilidad y adaptabilidad. A continuación se documentará través de diagramas de flujo la interacción entre las componentes y los eventos que la disparan.

5.3.7.1 Agregar un host a través de la interfaz gráfica

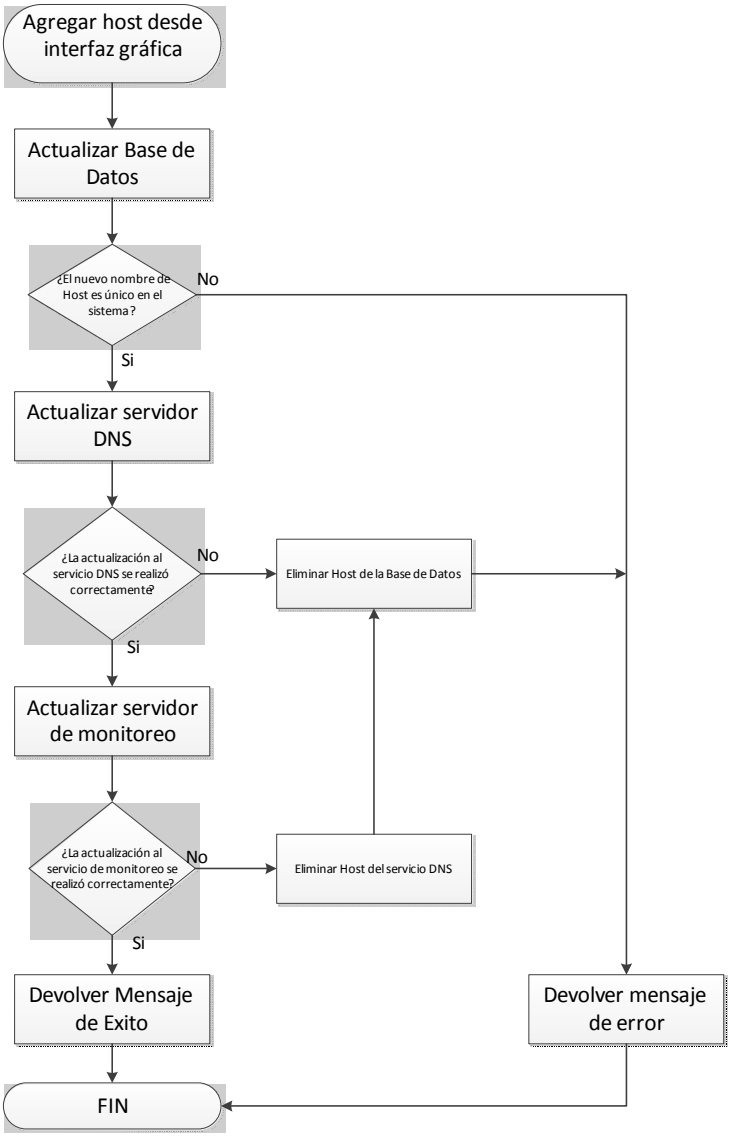


Figura 5.3.7.1.1: Diagrama de Flujo para el agregado de un Host en la herramienta Traffic Manager

5.3.7.2 Editar/Eliminar un host a través de la interfaz gráfica

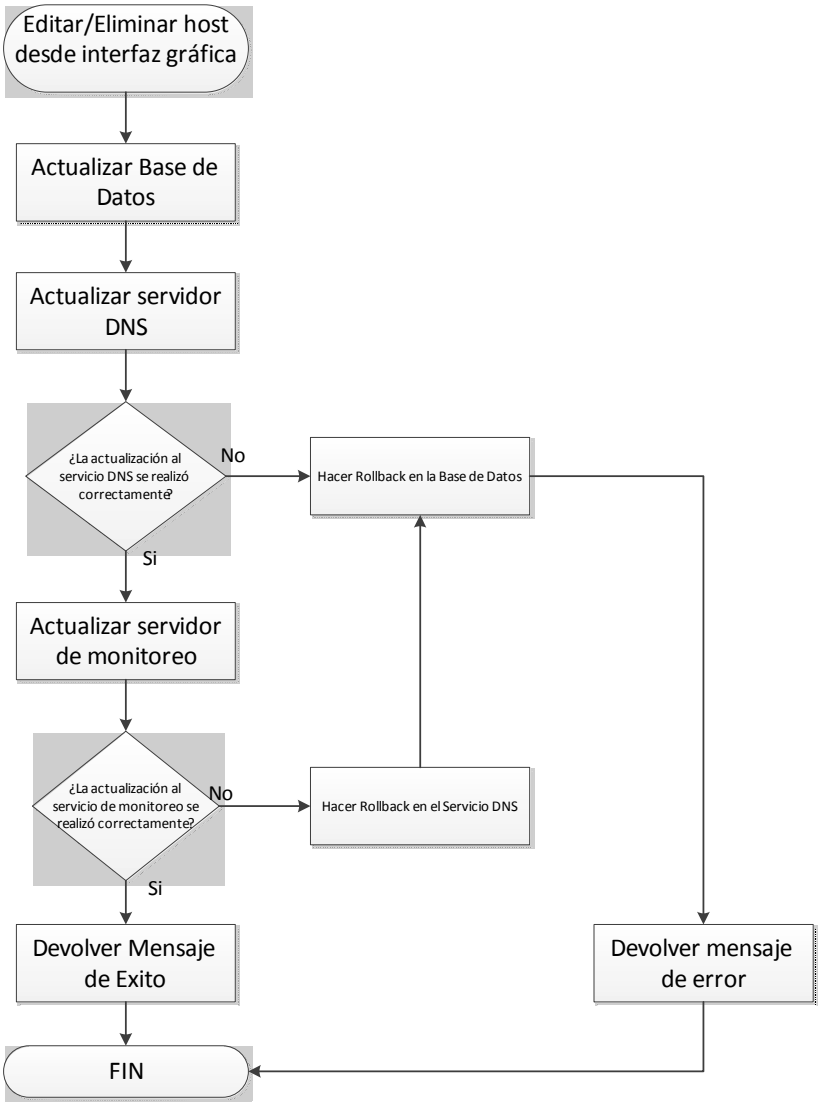


Figura 5.3.7.2.1: Diagrama de Flujo para Editar/Eliminar un Host en la herramienta Traffic Manager

5.3.7.3 Agregar una zona de DNS a través de la interfaz gráfica

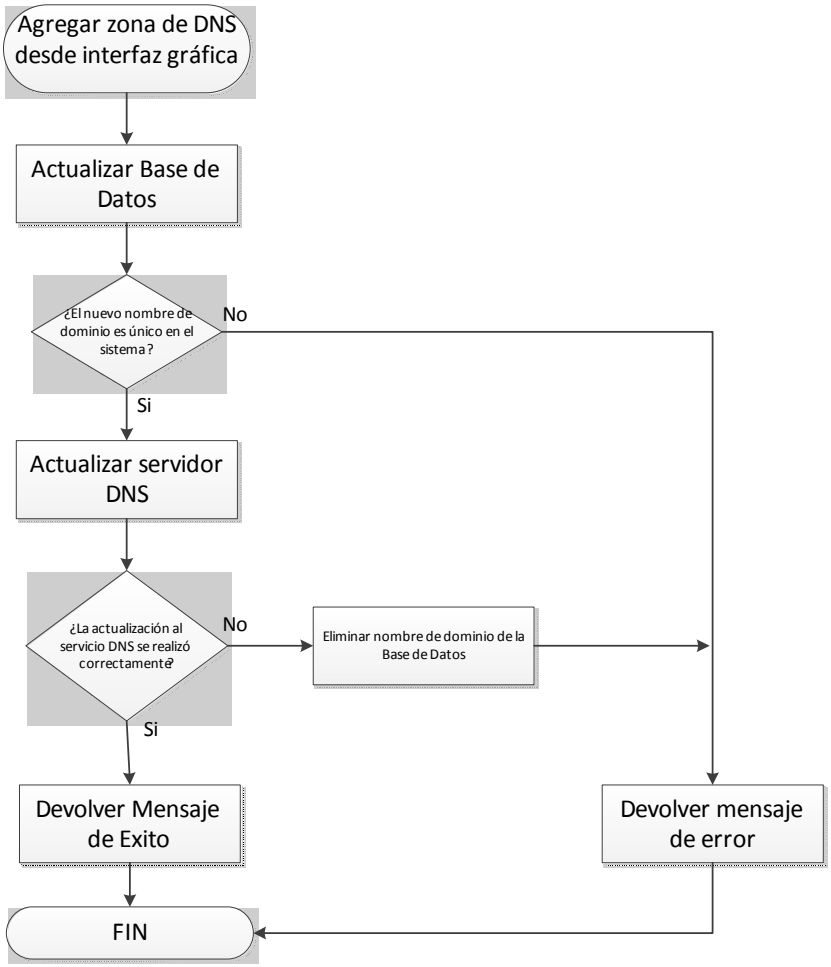


Figura 5.3.7.3.1: Diagrama de Flujo para el agregado de una zona de DNS en la herramienta Traffic Manager

5.3.7.4 Eliminar una zona de DNS a través de la interfaz gráfica

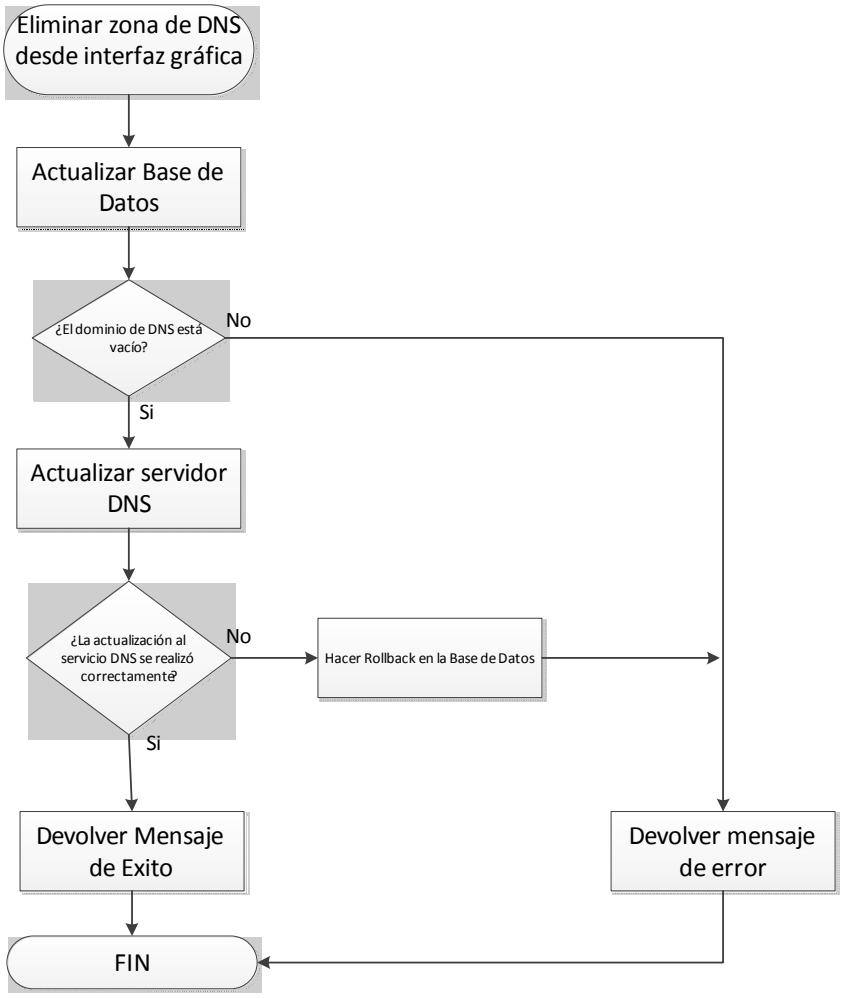


Figura 5.3.7.4.1: Diagrama de Flujo para eliminar una zona de DNS en la herramienta Traffic Manager

5.3.7.5 Detección de la caída/recuperación de un servicio desde el módulo administrador

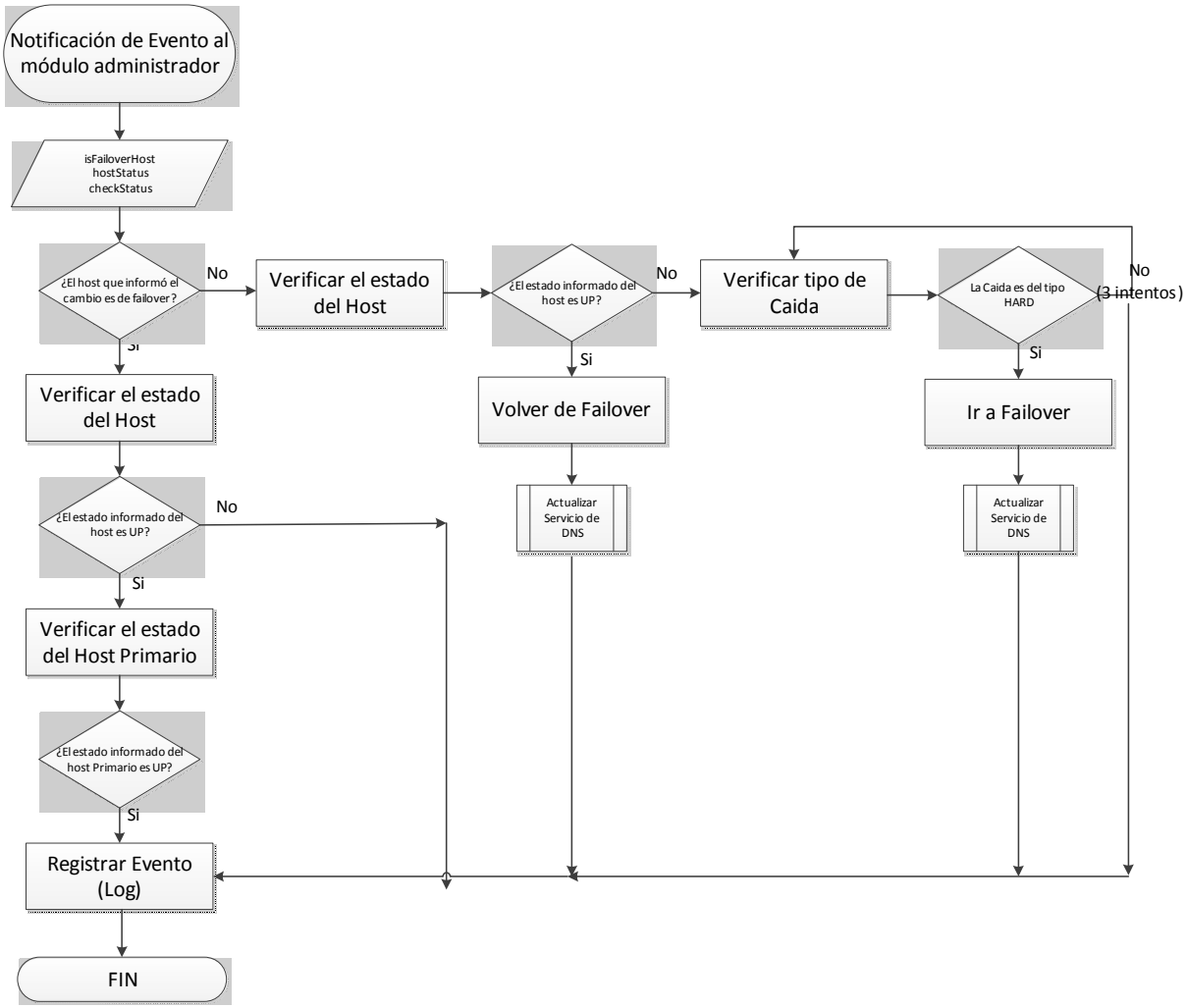


Figura 5.3.7.5.1: Diagrama de Flujo para la detección/recuperación de un Host en la herramienta Traffic Manager

5.3.8 Distribución de la solución Traffic Manager

Si bien el producto final del presente trabajo se encuentra representado a través de 2 máquinas virtuales que pueden ser instaladas en cualquier hipervisor, el código fuente de la aplicación ha sido publicado en el siguiente sitio:

https://github.com/ndelrio2000/trafficManager

El sitio representa un repositorio de GitHub, el cual ha sido utilizado como versionador del software durante el período de desarrollo. El software se encuentra publicado bajo el esquema abierto, lo que significa que cualquier usuario puede acceder al código para realizar descargas del mismo e inclusive clonarlo en nuevos proyectos que permitan ampliar la funcionalidad del mismo.

Para la obtención del código fuente, es posible clonar el repositorio donde se encuentra publicado el software, a través de la utilización del comando `git`. Para ello, utilizando cualquier distribución GNU/Linux y posicionado sobre un directorio en el cual el usuario posea permisos de escritura, se puede ejecutar el siguiente comando:

```
git clone https://github.com/ndelrio2000/trafficManager trafficManager
```

El comando anterior permitirá realizar la descarga del software, el cual deberá ser ubicado en los directorios correspondientes dentro de la estructura de directorios, de acuerdo a lo documentado en capítulos anteriores.

Las máquinas virtuales han sido copiadas en un DVD el cuál se adjunta al presente trabajo. En el directorio **MaquinasVirtuales**, se pueden encontrar 2 archivos, los cuales representan discos virtuales en formato VMDK que pueden ser importados en cualquier hipervisor. Al momento de crear las máquinas virtuales es importante destacar que las mismas deben ser configuradas seleccionando una arquitectura de 64 bits y con una configuración mínima que incluya un adaptador de red. Los archivos contenidos en el directorio son:

- **trafficManager.vmdk:** Representa a la máquina virtual que contiene toda la lógica de monitoreo e interfaz de administración desarrollada en el presente trabajo.
- **dnsSecundario.vmdk:** Representa a la máquina virtual que posee el servidor DNS secundario instalado con el cual se comunicará la máquina trafficManager.

En ambos casos, es necesario configurar los parámetros de red acorde a la infraestructura en la que se desee instalar el software. Para ello se deberá ingresar a las máquinas utilizando las credenciales de usuario documentadas en el cuadro **5.3.1.2**. La configuración de red deberá realizarse sobre el archivo

`/etc/network/interfaces`. Una vez realizada la configuración re red, se podrá acceder a la herramienta utilizando cualquier navegador web y apuntando a la dirección IP configurada en la máquina virtual **trafficManager.vmdk**.

5.4 Conclusiones finales del capítulo

Le herramienta desarrollada permite la realización de chequeos sobre los recursos gestionados y la asignación de nombres a través de la gestión del servicio DNS. La utilización de software libre que provea funciones, permitió que el proceso de desarrollo sea más ágil debido a la reutilización de funciones. Este mismo esquema se adoptó para los desarrollos que el tesista tuvo que realizar, proveyendo grandes ventajas al momento de la corrección de fallas. El hecho de encapsular funcionalidad y definir roles en cada módulo que forma parte de la solución, permitió que cada uno actúe hacia los demás como una caja negra, abstrayendo la lógica programada del resto de los módulos. Esta metodología de trabajo permite que futuras mejoras u agregados al software puedan realizarse de modo sencillo.

Capítulo 6

Validación de la Solución y Esquema de Pruebas

6.1 Introducción

El proceso de desarrollo de software, ha sido un importante tema de estudio desde su comienzo hasta la actualidad. Se han estudiado y desarrollado diversas técnicas y modelos, las cuales permiten dividir las tareas en etapas con el fin de documentar los procesos y actividades que deben desarrollarse en cada una de ellas. De las distintas técnicas o modelos de desarrollo, ha surgido por ejemplo el modelo en cascada, que define un enfoque de tareas las cuales se desarrollan siguiendo un estricto flujo secuencial. Otro enfoque es el basado en prototipos, que prevé desarrollos parciales que luego se adaptarán a lo largo del tiempo con el fin de brindar la solución final. En la actualidad es muy común escuchar hablar acerca de metodologías ágiles para el desarrollo. Sin duda alguna, todas las técnicas prevén como actividad la validación y prueba del desarrollo producido.

Este capítulo tiene como finalidad validar la solución planteada teniendo en cuenta su propio funcionamiento y comparándola con otras soluciones con el fin de valorar su desempeño. Para la realización de las pruebas que permitan determinar la calidad de la solución implementada, se propondrán diversos esquemas de red sobre los cuales se realizarán ensayos planteando distintos escenarios.

6.2 Software utilizado para la realización de pruebas

Para la realización de las pruebas se optó por utilizar el simulador de redes GNS3. Graphical Network Simulator versión 3 (GNS3 por sus siglas), es una herramienta de código abierto que actúa como interfaz gráfica para la simulación de redes reales. Dentro de sus características principales, permite simular el sistema operativo IOS de los routers Cisco, y acceder a máquinas virtuales de VirtualBox. La herramienta que hace posible la simulación, en realidad es Dynamips que consiste en un emulador de routers Cisco que permite correr un conjunto de imágenes estándares de diversos modelos de routers, switches y dispositivos de seguridad de la empresa. Dynamips no posee una interfaz gráfica integrada, sino que debe ser administrado utilizando la línea de comandos. Es por esta razón que GNS3 representa una componente fundamental de la solución.

La razón por la cual se optó por este simulador y no por otros como Boson Network Simulator[25] o Common Open Research Emulator (CORE)[26], reside en que Dynamips permite correr imágenes nativas del sistema operativo de los routers Cisco. Es en este último sistema en el cual el autor de presente trabajo posee experiencia, y adicionalmente es uno de los más utilizados por los enrutadores en InterNet para llevar a cabo el proceso de ruteo BGP. A través de la utilización del simulador se pretenden lograr métricas de performance lo más cercanas a la realidad posible, montando escenarios que utilicen los mismos dispositivos que se utilizan en el mundo real.

La versión GNS3 utilizada para la realización de las pruebas es 1.3.1 que incluye la instalación de Dynamips durante el mismo proceso de instalación de software. Adicionalmente se optó por la utilización de imágenes virtuales de routers Cisco modelo 3725 con IOS versión 12.4(25d) que poseen la capacidad de correr el protocolo BGP. Para la obtención de estadísticas de red se utilizaron la herramienta de debug incluida en los router y la herramienta Wireshark.

6.3 Primer escenario de Pruebas: Tiempo de convergencia de BGP

Para el primer escenario de pruebas, se montó una infraestructura de red basada en routers modelo 3725 de la línea Cisco y máquinas virtuales de Virtual Box. Sobre la topología se realizó una configuración básica de BGP con el fin de poder publicar las redes hacia todos los routers que forman parte de la solución. La siguiente figura muestra el ejemplo de Topología utilizada:

[25] Boson Network Simulator http://www.boson.com/
[26] Core http://www.nrl.navy.mil/itd/ncs/products/core

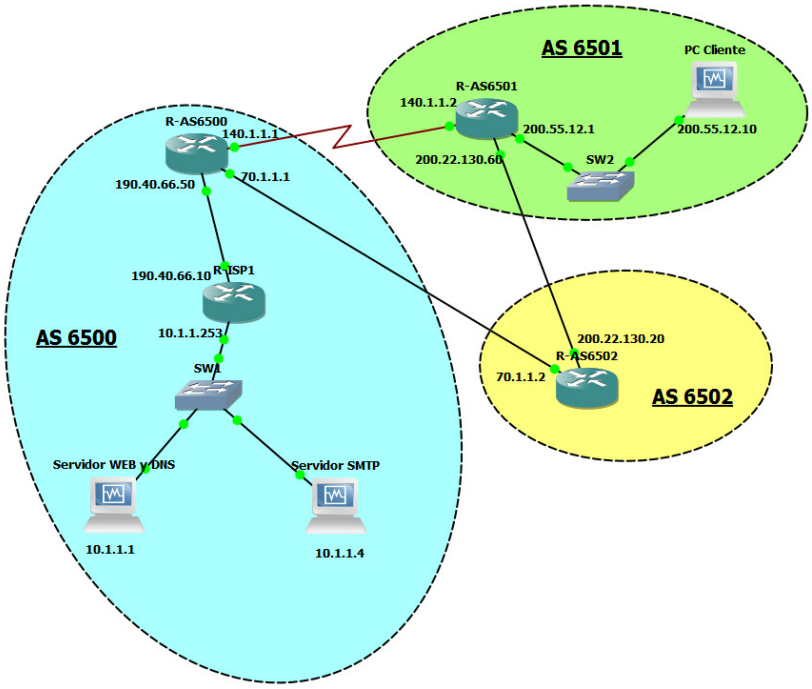


Figura 6.3.1: Topología de red utilizada en el primer escenario de prueba con BGP

La zona de color celeste denota al sistema autónomo de BGP identificado con el número 6500. Se trata de una red que utiliza direccionamiento privado para su red interna y el concepto de NAT para la publicación de servicios. La red cuenta con el espacio de direcciones públicas **190.40.66.0/24** provisto por su proveedor y es a través de este último que publica sus servicios para que sean accedidos desde InterNet. El router con nombre **R-ISP1** posee en su interfaz externa configuradas diversas direcciones IP. Las mismas son utilizadas para trasladar los requerimientos realizados por los usuarios externos hacia los servidores internos de la red con direccionamiento privado. Las traslaciones o NAT configurados son:

IP Externa (R-ISP1)	IP servidor Interno	Protocolos
190.40.66.11	10.1.1.1	Todos
190.40.66.14	10.1.1.4	Todos

Cuadro 6.3.1: Traslaciones de red (NAT) configurados para el primer escenario de pruebas

Adicionalmente, cada cliente de la red interna (Servidor WEB y DNS, Servidor SMTP) tiene configurada su ruta por defecto hacia el router **R-ISP1**, cuya dirección IP en la red interna es **10.1.1.253**. El router **R-ISP1** tiene configurada su ruta por defecto hacia el router de borde del proveedor de servicios de InterNet, que dialoga BGP con routers de otros sistemas autónomos y es quien se encarga de publicar el

direccionamiento IP que se utiliza para publicar servicios y redireccionar los requerimientos hacia la red interna. La configuración del router **R-ISP1** se muestra en el siguiente cuadro:

```
hostname R-ISP1

interface FastEthernet0/0

description ###Hacia la WAN para publicar servicios

ip address 190.40.66.11 255.255.255.0 secondary

ip address 190.40.66.14 255.255.255.0 secondary

ip address 190.40.66.10 255.255.255.0

ip nat outside

interface FastEthernet0/1

description ###Hacia la LAN

ip address 10.1.1.253 255.255.255.0

ip nat inside

duplex auto

speed auto

ip route 0.0.0.0 0.0.0.0 190.40.66.50

ip nat inside source static 10.1.1.1 190.40.66.11

ip nat inside source static 10.1.1.4 190.40.66.14
```

Cuadro 6.3.2: Configuración del router **R-ISP1** para el primer escenario de pruebas

El router **R-AS6500** se encuentra interconectado a través de 2 enlaces con los sistemas autónomos denotados por el número 6501 (área de color verde en la gráfica de la figura **6.3.1**) y 6502 (área de color amarillo en la topología de red. Los 3 routers de la topología cuyo nombre posee el prefijo "**AS**", dialogan BGP e intercambian información de ruteo sobre las rutas que cada uno tiene directamente conectadas.

En el sistema autónomo 6501, se instaló una PC denotada por el nombre "PC Cliente", que posee instalado un sistema operativo Windows 8, y es la que será utilizada para la realización de pruebas en la topología.

La configuración realizada sobre el router **R-AS6500** se muestra en el siguiente cuadro:

```
hostname R-AS6500

interface FastEthernet0/0

description ###Hacia AS6502

ip address 70.1.1.1 255.255.255.252

duplex auto

speed auto

interface FastEthernet0/1

description ###Hacia R-ISP1

ip address 190.40.66.50 255.255.255.0

duplex auto

speed auto

interface Serial1/0

description ###Hacia AS6501

ip address 140.1.1.1 255.255.255.252

serial restart-delay 0

router bgp 6500

no synchronization

bgp log-neighbor-changes

redistribute connected

neighbor 70.1.1.2 remote-as 6502

neighbor 140.1.1.2 remote-as 6501

no auto-summary
```

Cuadro 6.3.3: Configuración del router **R-AS6500** para el primer escenario de pruebas

La configuración realizada sobre el router R-AS6501 se muestra en el siguiente cuadro:

```
hostname R-AS6501

interface FastEthernet0/0

description ###Hacia AS6502

ip address 200.22.130.60 255.255.255.0

duplex auto

speed auto

interface FastEthernet0/1

description ###Hacia la Red Interna

ip address 200.55.12.1 255.255.255.0

duplex auto

speed auto

interface Serial1/0

description ### Hacia AS6500

ip address 140.1.1.2 255.255.255.252

serial restart-delay 0

router bgp 6501

no synchronization

bgp log-neighbor-changes

redistribute connected

neighbor 140.1.1.1 remote-as 6500

neighbor 200.22.130.20 remote-as 6502

no auto-summary
```

Cuadro 6.3.4: Configuración del router **R-AS6501** para el primer escenario de pruebas

La configuración realizada sobre el router R-AS6502 se muestra en el siguiente cuadro:

```
hostname R-AS6502

interface FastEthernet0/0

description ###Hacia AS6500

ip address 70.1.1.2 255.255.255.252

duplex auto

speed auto

interface FastEthernet1/0

description ###Hacia AS6501

ip address 200.22.130.20 255.255.255.0

duplex auto

speed auto

router bgp 6502

no synchronization

bgp log-neighbor-changes

redistribute connected

neighbor 70.1.1.1 remote-as 6500

neighbor 200.22.130.60 remote-as 6501

no auto-summary
```

Cuadro 6.3.5: Configuración del router **R-AS6502** para el primer escenario de pruebas

Como se puede observar, en todos los routers se ha configurado el comando **log-neighbor-changes**, que permite registrar en consola las notificaciones de BGP que envían los routers a sus vecinos. Adicionalmente se habilitó el debug de BGP en la consola de todos los routers y se configuraron los relojes de los mismos para que todos tengan la misma hora. Para ello, se configuró el servidor "Servidor Web y DNS" cuya IP es **10.1.1.1** como servidor de hora de toda la topología. El mismo es accedido por todos los routers a través de la redirección de puertos configurada en el router **R-ISP1** que permite el acceso al puerto **UDP/123** que es utilizado por el servidor de hora para la publicación de su servicio. Esta configuración

permite que los mensajes de debug de los distintos routers tengan sincronizados sus horas de modo tal de poder sacar comparativas de tiempos.

Para la realización de la primera prueba en la topología planteada, se utilizará el protocolo ICMP. Desde la "PC Cliente" en el sistema autónomo 6501 se enviará tráfico de red hacia la dirección IP 190.40.66.11. La misma es una dirección IP secundaria del router R-ISP1 perteneciente al sistema autónomo 6500 que redirige todo el tráfico hacia la IP 10.1.1.1 que se corresponde con el "Servidor Web y DNS".

La siguiente figura muestra la ejecución del comando Ping hacia el nombre de DNS tm.ejemplo.com.ar que resuelve a la 190.40.66.11 desde la PC Cliente, donde se puede visualizar que los paquetes llegan a destino y son correctamente respondidos por el mismo. La PC cliente posee configurado su servidor DNS hacia la IP 190.40.66.11 que es el servidor de nombres autoritativo para la zona de DNS mencionada:

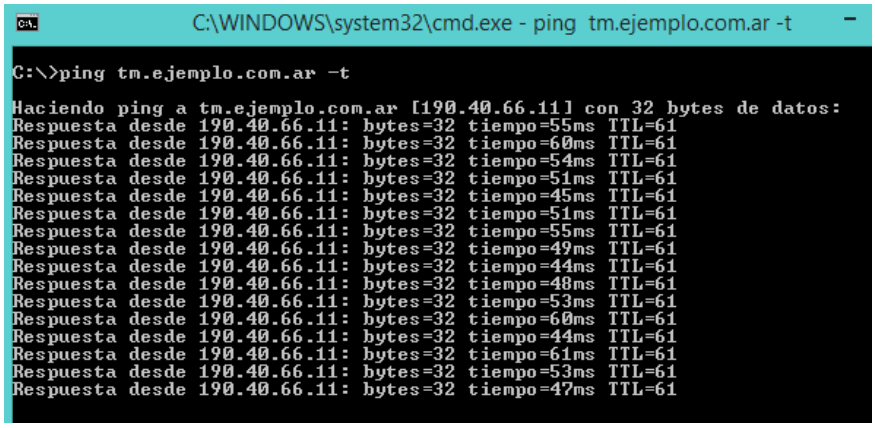


Figura 6.3.2: Prueba de ICMP desde la PC Cliente hacia la IP 190.40.66.11

Del análisis de la topología planteada, se desprende que la información que la PC Cliente envía al Servidor WEB y DNS, puede seguir alguno de los siguientes caminos:

1. PC Cliente → R-AS6501 → R-AS6500 → R-ISP1 → Servidor WEB y DNS
2. PC Cliente → R-AS6501 → R-AS6502 → R-AS6500 → R-ISP1 → Servidor WEB y DNS

La decisión acerca del modo de encaminar los datos, es tomada por el router **R-AS6501** en base a la información topológica aprendida a través del proceso de ruteo BGP. La siguiente figura muestra la tabla de ruteo del router **R-AS6501**, donde puede observarse que la ruta seleccionada para alcanzar el destino se corresponde con la primera opción. Este es un proceso lógico de BGP, ya que como se ha mencionado anteriormente, el protocolo elige como ruta preferida a la que posee el camino más corto hacia el destino.

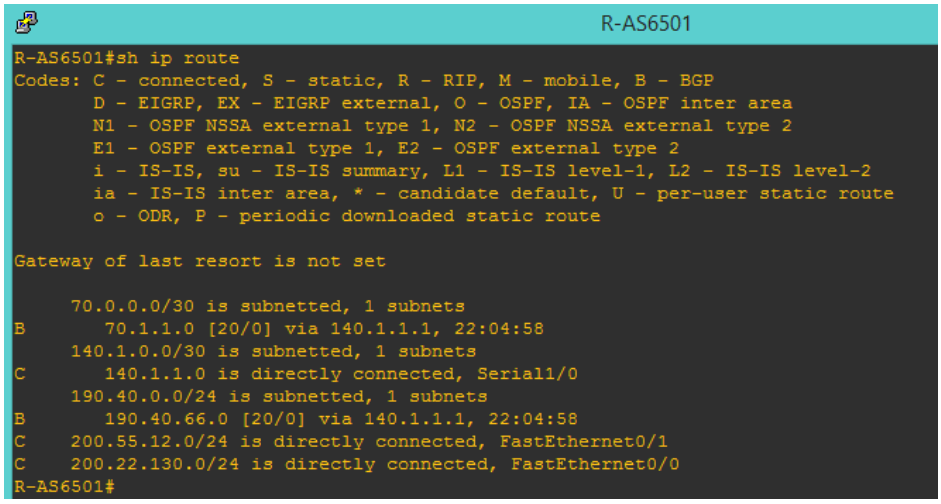


Figura 6.3.3: Tabla de ruteo del router **R-AS6501**

La primera falla a inyectar en la red, será provocar un corte intencional del vínculo que comunica al router **R-AS6501** con el router **R-AS6500**. De este modo, entrará en funcionamiento el proceso de convergencia de BGP, el cual causará que la nueva ruta elegida por el router **R-AS5601** sea el camino 2, es decir PC Cliente → R-AS6501 → R-AS6502 → R-AS6500 → R-ISP1 → Servidor WEB y DNS. Para ello, en el router **R-AS6500** se procederá a bajar la interfaz **Serial1/0** a través del comando **shutdown**.

Como se puede observar en la siguiente gráfica, la PC Cliente dejó de recibir respuesta del Servidor Web y DNS durante un período de tiempo, causando esto la pérdida de 19 paquetes ICMP, lo cual representa el 61% del tráfico total de 31 paquetes enviados.

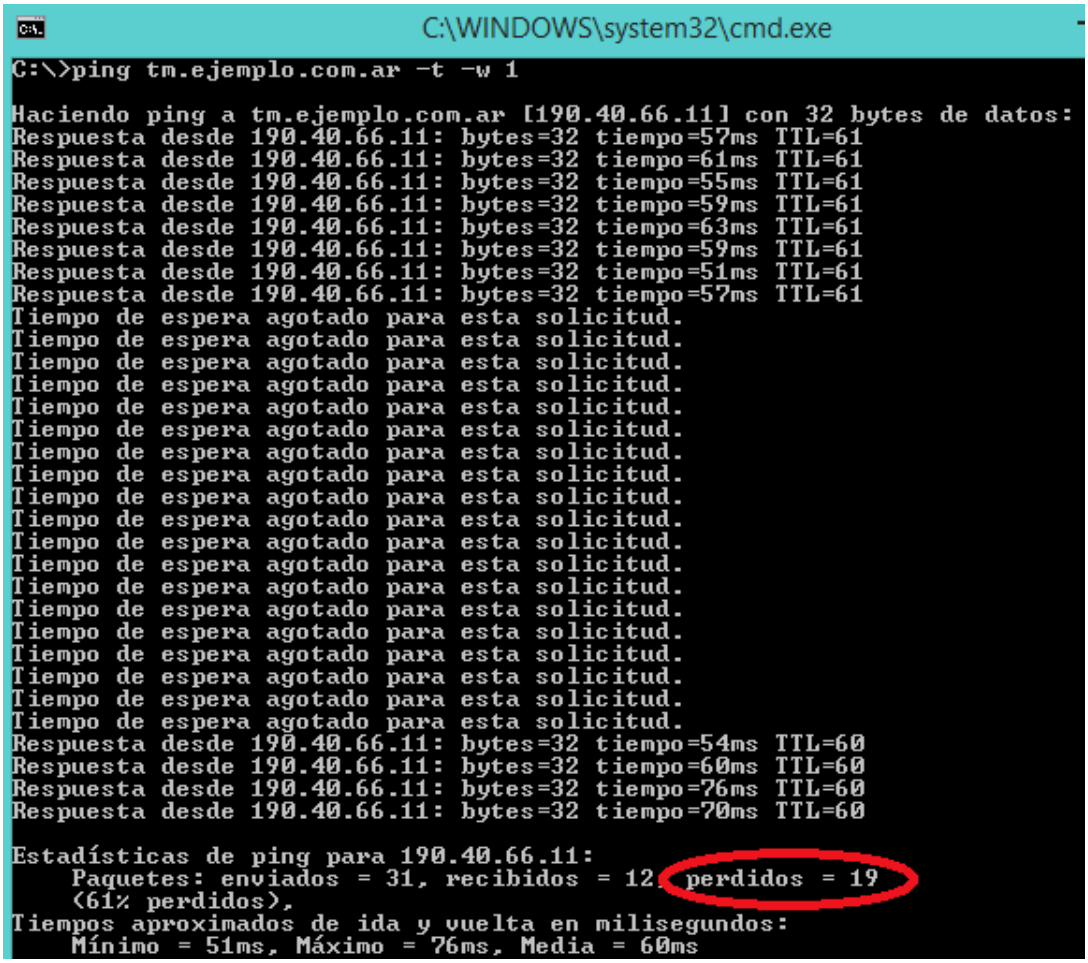


Figura 6.3.4: Tráfico ICMP desde la PC Cliente hacia la IP 19040.66.11 durante convergencia de BGP

El tiempo de convergencia, puede contabilizarse con la información de debug de las consolas de los routers **R-AS6500** y **R-AS6501** que se muestran en las siguientes figuras. Como puede visualizarse, a las 16 horas, 51minutos, 51 segundos y 134 milisegundos, el proceso BGP del router **R-AS6500** detecta la caída de la interfaz del router que lo comunica directamente con el sistema autónomo 6501. Esta caída, causa que el router **R-AS6500** modifique su tabla de ruteo, inyectando a la misma información que determine que para alcanzar a la red **200.55.12.0/24**, ahora debe encaminar al tráfico a través del sistema autónomo 6502.

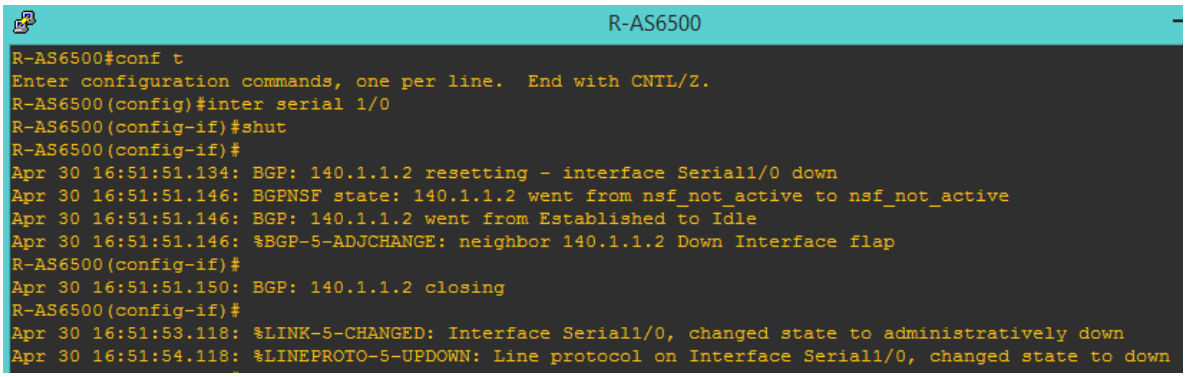


Figura 6.3.5: Detección de caída de vínculo por el proceso BGP en router **R-AS6500**

El router **R-AS6501**, detecta la caída del router **R-AS6500**, luego de no recibir información del mismo durante el período de tiempo establecido en la configuración por defecto de BGP. En ese momento, determina que la red **190.40.66.0/24** que anteriormente se alcanzaba directamente a través del vínculo con el sistema autónomo 6500, ahora se debe alcanzar a través del sistema autónomo 6502. En la siguiente figura puede observarse que el router **R-AS6501** detecta la caída y toma la decisión de convergencia a las 16 horas, 52 minutos, 19 segundos y 404 milisegundos:

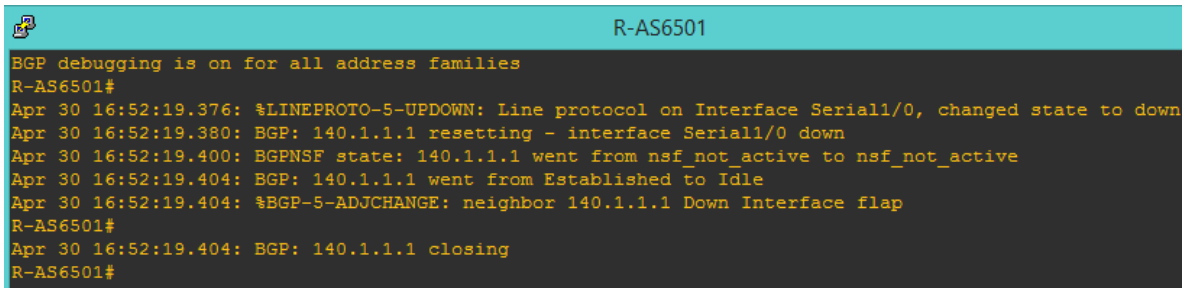


Figura 6.3.6: Detección de caída de vínculo por el proceso BGP en router **R-AS6501**

La nueva tabla de ruteo del router **R-AS6501** se muestra en la siguiente figura, donde puede verse la diferencia en la tabla de ruteo generada por la caída del vínculo, en la que la red **190.40.66.0/24** ahora es alcanzada a través del router con IP 200.22.130.20 correspondiente al sistema autónomo 6502:

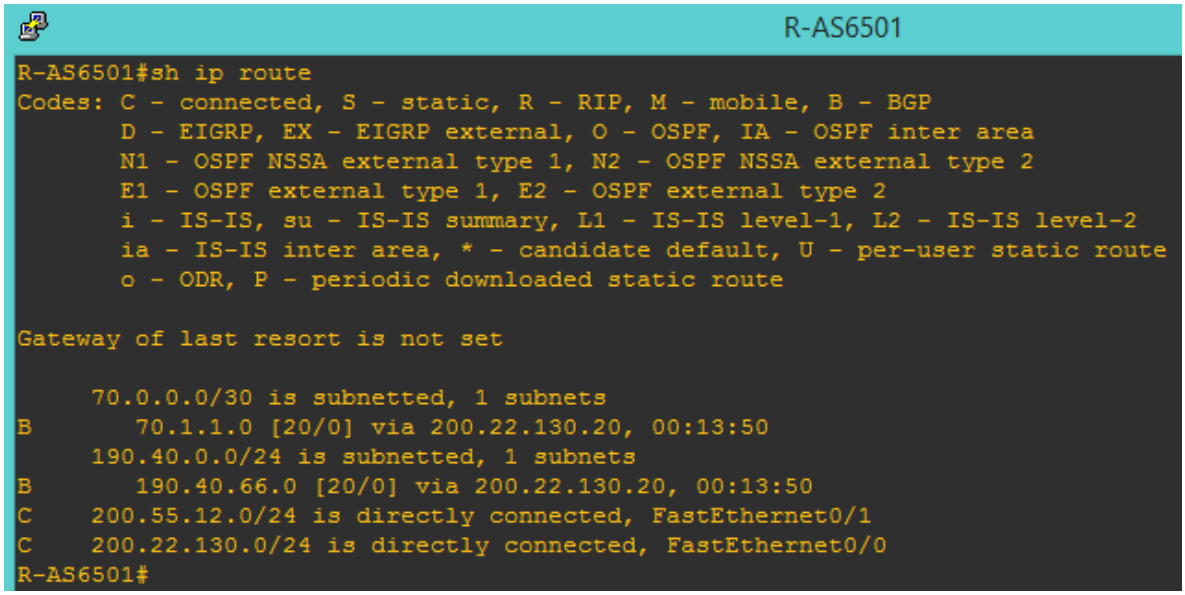


Figura 6.3.7: Tabla de ruteo del router **R-AS6501** luego de la convergencia de BGP

*La conclusión final de esta prueba demuestra que la caída de un vínculo productivo a través del cual se está traficando información genera la pérdida de 19 paquetes ICMP y tarda en converger **28 segundos y 270 milisegundos**. Este tiempo es suficiente para que la mayoría de las sesiones de tráfico establecidas finalice por timeout y tenga que volver a establecerse.*

6.4 Segundo escenario de Pruebas: Tiempo de convergencia de BGP con penalización

El procesamiento de BGP resulta una tarea muy costosa para los routers en concepto de utilización de CPU y memoria. Es por esta razón, que los ISP utilizan técnicas de penalización, también conocidas como dampening, con el fin de evitar que enlaces inestables generen repetidas actualizaciones en el proceso de BGP. Para la siguiente prueba, se procederá a habilitar la funcionalidad de penalización en el router **R-AS6501** y se simularán reiteradas caídas del vínculo que comunica el router **R-ISP1** con el router **R-AS6500**. Es de esperar que ante cada caída del vínculo el tráfico ICMP que se envía desde la PC Cliente hacia el Servidor WEB y DNS deje de responder. Lo que se pretende demostrar es que la funcionalidad de dampening puede dejar una red completamente aislada del resto por un período de tiempo muy superior al tiempo de convergencia estándar de BGP, una vez que se detectó la inestabilidad de la red por parte de los routers. La siguiente figura muestra la configuración realizada en el router **R-AS6501** y los parámetros por defecto de configuración que se utilizarán para la penalización:

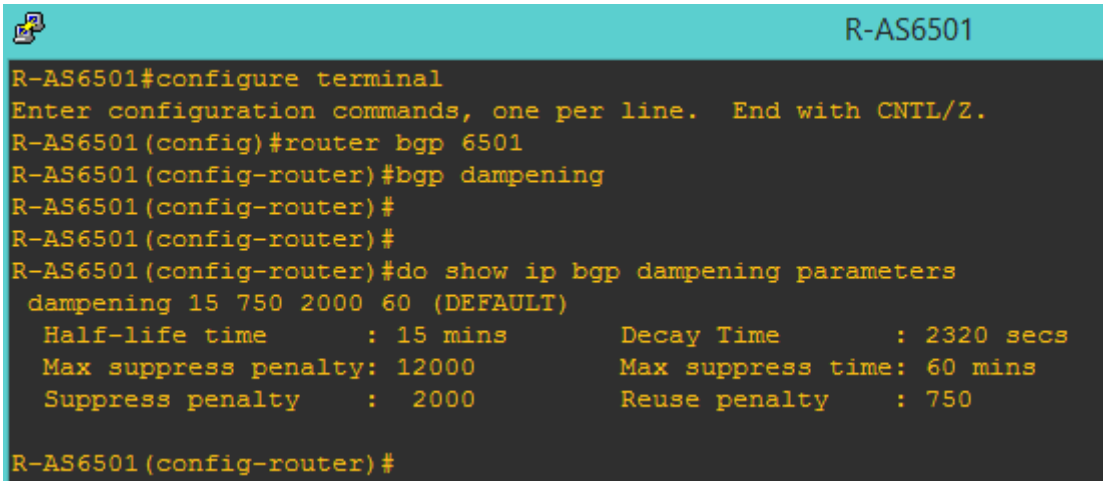


Figura 6.4.1: configuración de penalizaciones de BGP en router **R-AS6501**

La configuración de los routers Cisco, una vez que se habilita la funcionalidad de dampening, determina que cada vez que se anuncian cambios (flapping) sobre una ruta se debe aplicar una penalización. Una ruta en estado de flapping es aquella que aparece con estado de UP o DOWN, o que se modifica un atributo BGP de la misma, como por ejemplo el AS_PATH. Los valores en la configuración por defecto que utilizan los routers cisco son:

- **Penalización:** En cada flapeo, se incrementa en 1000. Si hay un cambio en un atributo, el incremento es de 500.
- **Umbral de supresión (Supress penalty):** cuando la penalización alcanza el valor de 2000
- **Umbral para rehusar (Rehuse penalty):** cuando la penalización alcanza el valor 750
- **Tiempo medio (Half-life time):** 15 minutos. Durante este tiempo, la ruta no será suprimida, aunque puede ser penalizada si flapea (lo cual puede ocurrir cada 5 segundos).
- **Tiempo máximo de supresión:** 4 veces el tiempo medio. Cada 10 segundo se revisan las rutas con el fin de verificar si alguna de ellas puede pasar de suprimida a anunciarse. Esto se cumplirá cuando su número de penalizaciones se haya reducido por debajo del límite de rehuso. En el caso de que pase el tiempo máximo de supresión y no se haya reducido suficientemente el número de penalizaciones, la ruta dejará de estar suprimida para poder ser anunciada.

Para la simulación de la prueba, se comenzará con la generación de trafico ICMP desde la “PC Cliente” hacia el “Servidor Web y DNS”. Adicionalmente, y luego de haber configurado la funcionalidad de dampening en el router **R-AS6501** como se mencionó anteriormente, se simularán caídas del vínculo entre el router **R-ISP1** y **R-AS6500**, de modo tal que el ultimo router envíe actualizaciones a través de BGP a sus vecinos indicando que la red **190.40.66.10/24** ya no es alcanzable. Debido a que el ultimo router se encuentra interconectado contra los sistemas autónomos 6501 y 6502, los mensajes UPDATE de BGP se

propagarán hacia los 2 vecinos que el miso posee configurados. Solo el router **R-AS6501** tiene la función de dampening habilitada, pero en caso de que **R-AS6502** también la tuviera, el mismo también calculará la penalización. Es importante destacar que si bien solo **R-AS6501** tiene el dampening habilitado, al estar interconectado a los 2 sistemas autónomos y a su vez alcanzar a la red **190.40.66.0/24** a través de ellos, el router calculará la penalización a través de ambos caminos. La siguiente figura muestra la tabla BGP del router **R-AS6501**, así como también que no existe ninguna ruta penalizada al momento de iniciar la prueba. Como se puede observar, la ruta hacia la red **190.40.66.0/24** es a través de los 2 (dos) caminos por los que puede ser alcanzada. El camino preferido es a través del router **R-AS6500** que se encuentra directamente conectado:

R-AS6501

```
R-AS6501#sh ip bgp
BGP table version is 37, local router ID is 200.55.12.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*  70.1.1.0/30      140.1.1.1                0             0 6500 ?
*>                  200.22.130.20            0             0 6502 ?
*  140.1.1.0/30     140.1.1.1                0             0 6500 ?
*>                  0.0.0.0                  0         32768 ?
*                   200.22.130.20                          0 6502 6500 ?
*> 190.40.66.0/24   140.1.1.1                0             0 6500 ?
*                   200.22.130.20                          0 6502 6500 ?
*  200.22.130.0     140.1.1.1                              0 6500 6502 ?
*                   200.22.130.20            0             0 6502 ?
*>                  0.0.0.0                  0         32768 ?
*> 200.55.12.0      0.0.0.0                  0         32768 ?
R-AS6501#
R-AS6501#
R-AS6501#sh ip bgp dampening damp
R-AS6501#sh ip bgp dampening dampened-paths

R-AS6501#
```

Figura 6.4.2: Tabla de ruteo de BGP en router **R-AS6501**

Con el fin de simular un flapeo de la red **190.40.66.0/24**, se bajará la interfaz **FastEthernet0/1** del router **R-AS6500** que es la que tiene directamente conectada a dicha red. Para ello, en modo configuración de interfaz se ejecutará el comando **shutdown**. Como puede verse en la siguiente figura, el proceso de ruteo BGP en el router **R-AS6501**, ha sido notificado del cambio en la topología y marcó la ruta bajo un estado **h** (history), que significa que no hay un camino mejor hacia la ruta indicada, pero que la información de flapping (penalidades) aún se conserva.

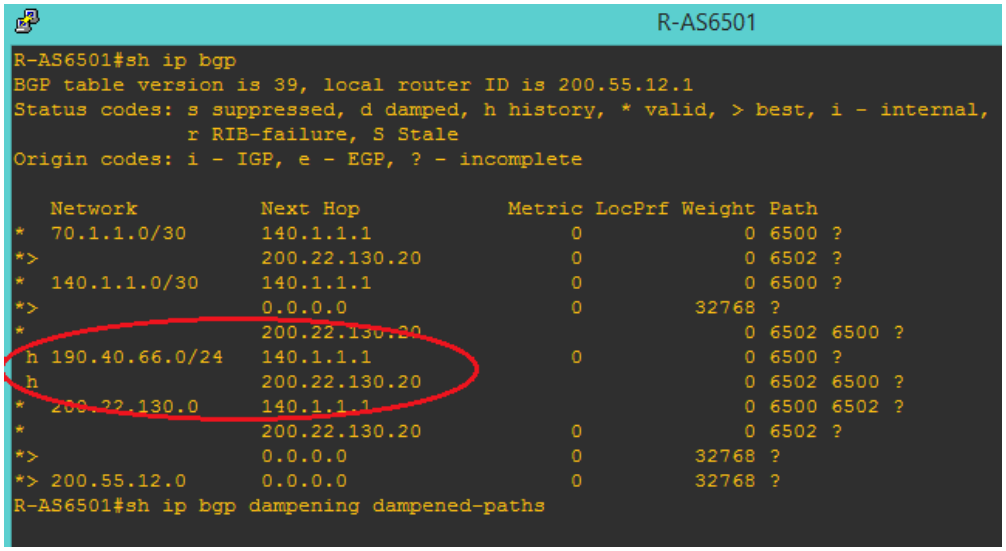


Figura 6.4.3: Tabla de ruteo de BGP en router **R-AS6501** luego de detectar caída de red 190.40.66.0/24

Adicionalmente y con el fin de verificar el estado de la penalización, puede ejecutarse el comando **show ip bgp 190.40.66.0**, el cual brindará información detallada sobre el destino. La siguiente figura muestra el resultado de ejecutar dicho comando en el router **R-AS6501**:

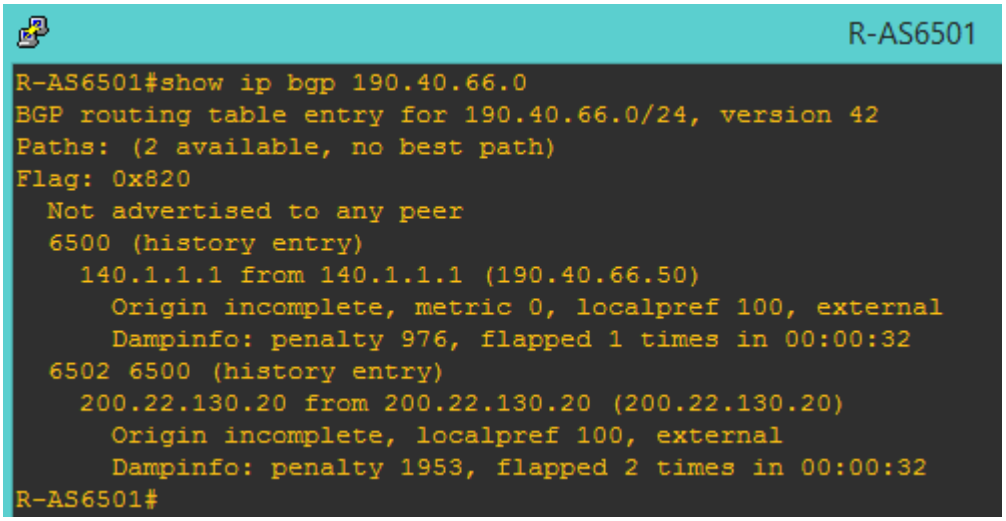


Figura 6.4.4: Análisis de penalidad para la red 190.40.66.0 en **R-AS6501**

Como puede observarse, el valor de penalización para la ruta, a través de los 2 caminos posibles que puede ser alcanzada, no ha llegado al valor de 2000 que es el valor fijado para el límite de supresión (valor por defecto configurado en routers cisco), de manera que la ruta no ha sido aún suprimida y sigue figurando en la tabla de ruteo del router **R-AS6501**. Es importante destacar que la penalización se calcula por camino y no por destino, con lo cual podría ser penalizada por un camino y seguir siendo alcanzable y listada en la tabla de ruteo, a través de un camino alternativo. Si el enlace siguiera flapeando repetidas

veces, la red **190.40.66.0/24** podría verse penalizada a través de los 2 caminos por los que la puede alcanzar el router **R-AS6501** y marcarse en la tabla de ruteo de modo tal que no sea más accedida. La siguiente figura muestra que luego de causar 1 (una) caída más sobre el enlace, el mismo ya ha sido penalizado por el camino que lo interconecta con el sistema autónomo 6502:

```
R-AS6501#show ip bgp 190.40.66.0
BGP routing table entry for 190.40.66.0/24, version 54
Paths: (2 available, best #2, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1
  6502 6500, (suppressed due to dampening)
    200.22.130.20 from 200.22.130.20 (200.22.130.20)
      Origin incomplete, localpref 100, valid, external
      Dampinfo: penalty 3649, flapped 4 times in 00:02:41, reuse in 00:01:23
  6500
    140.1.1.1 from 140.1.1.1 (190.40.66.50)
      Origin incomplete, metric 0, localpref 100, valid, external, best
      Dampinfo: penalty 1822, flapped 2 times in 00:02:41
```

Figura 6.4.5: Verificación de penalización de BGP por un camino para la red 190.40.66.0 en **R-AS6501**

De generarse nuevos flapeos sobre el enlace, se podrá observar que cuando el valor de penalización alcance el valor de supresión sobre el único enlace disponible, la ruta volverá a ser penalizada:

```
R-AS6501#show ip bgp 190.40.66.0
BGP routing table entry for 190.40.66.0/24, version 55
Paths: (2 available, no best path)
Flag: 0x820
  Not advertised to any peer
  6502 6500, (suppressed due to dampening)
    200.22.130.20 from 200.22.130.20 (200.22.130.20)
      Origin incomplete, localpref 100, valid, external
      Dampinfo: penalty 5921, flapped 7 times in 00:06:05, reuse in 00:00:58
  6500, (suppressed due to dampening)
    140.1.1.1 from 140.1.1.1 (190.40.66.50)
      Origin incomplete, metric 0, localpref 100, valid, external
      Dampinfo: penalty 3430, flapped 4 times in 00:06:05, reuse in 00:00:03
```

Figura 6.4.6: Penalización completa de BGP para la red 190.40.66.0 en **R-AS6501**

Al observar la tabla de BGP del router, se puede ver también que la ruta ha sido marcada como penalizada, y por último al ver la tabla global de ruteo del equipo se puede verificar que la ruta hacia la red destino ha sido suprimida:

```
R-AS6501#sh ip bgp
BGP table version is 55, local router ID is 200.55.12.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*  70.1.1.0/30      140.1.1.1                0             0 6500 ?
*>                  200.22.130.20            0             0 6502 ?
*  140.1.1.0/30     140.1.1.1                0             0 6500 ?
*>                  0.0.0.0                  0         32768 ?
*                   200.22.130.20                          0 6502 6500 ?
*d 190.40.66.0/24   200.22.130.20                          0 6502 6500 ?
*d                  140.1.1.1                0             0 6500 ?
*  200.22.130.0     140.1.1.1                              0 6500 6502 ?
*                   200.22.130.20            0             0 6502 ?
*>                  0.0.0.0                  0         32768 ?
*> 200.55.12.0      0.0.0.0                  0         32768 ?
R-AS6501#
R-AS6501#sh ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

     70.0.0.0/30 is subnetted, 1 subnets
B       70.1.1.0 [20/0] via 200.22.130.20, 2d22h
     140.1.0.0/30 is subnetted, 1 subnets
C       140.1.1.0 is directly connected, Serial1/0
C    200.55.12.0/24 is directly connected, FastEthernet0/1
C    200.22.130.0/24 is directly connected, FastEthernet0/0
R-AS6501#
```

La Ruta no figura en la tabla de ruteo global

Figura 6.4.7: Supresión en la tabla de ruteo global de la ruta penalizada por BGP en **R-AS6501**

Por último y con el fin de verificar el estado de la conectividad, se puede observar que la PC Cliente ya no posee conexión hacia el Servidor WEB y DNS, debido a que el router **R-AS6501** ha suprimido la ruta hacia el destino debido al flapeo:

```
C:\WINDOWS\system32\cmd.exe - ping  tm.ejemplo.com.ar -t -w 1
Respuesta desde 200.55.12.1: Host de destino inaccesible.
Respuesta desde 200.55.12.1: Host de destino inaccesible.
Respuesta desde 200.55.12.1: Host de destino inaccesible.
Respuesta desde 200.55.12.1: Host de destino inaccesible.
Respuesta desde 200.55.12.1: Host de destino inaccesible.
Respuesta desde 200.55.12.1: Host de destino inaccesible.
Respuesta desde 200.55.12.1: Host de destino inaccesible.
Respuesta desde 200.55.12.1: Host de destino inaccesible.
Respuesta desde 200.55.12.1: Host de destino inaccesible.
Respuesta desde 200.55.12.1: Host de destino inaccesible.
Respuesta desde 200.55.12.1: Host de destino inaccesible.
Respuesta desde 200.55.12.1: Host de destino inaccesible.
Respuesta desde 200.55.12.1: Host de destino inaccesible.
Respuesta desde 200.55.12.1: Host de destino inaccesible.
Respuesta desde 200.55.12.1: Host de destino inaccesible.
```

Figura 6.4.8: Verificación de pérdida de conectividad desde la PC Cliente hacia la IP 190.40.66.11

Como se mencionó anteriormente, esta prueba generó diversos cortes de red, debido a que el enlace que presentó inestabilidad era el mismo sobre el cual se publican los servicios que se intentan alcanzar. Independientemente de esta situación, lo que se pretende mostrar es que aun cuando el enlace haya estado en un estado de flapping y se haya recuperado; producto de la penalización recibida por los routers, la red no será alcanzable por un largo período de tiempo, generando esto que los servicios no puedan ser accedidos inclusive cuando los problemas de red hayan sido solucionados. En el ejemplo planteado, puede verse en la figura **6.4.6**, que la penalización obtenida para la ruta utilizada a través del camino que la vincula directamente con el sistema autónomo 6500 posee un valor de 3430. La siguiente figura muestra el instante en el que el router determinó que la ruta debe dejar de ser penalizada y se vuelve a inyectar en la tabla de ruteo de modo tal que el cliente pueda alcanzar nuevamente al destino:

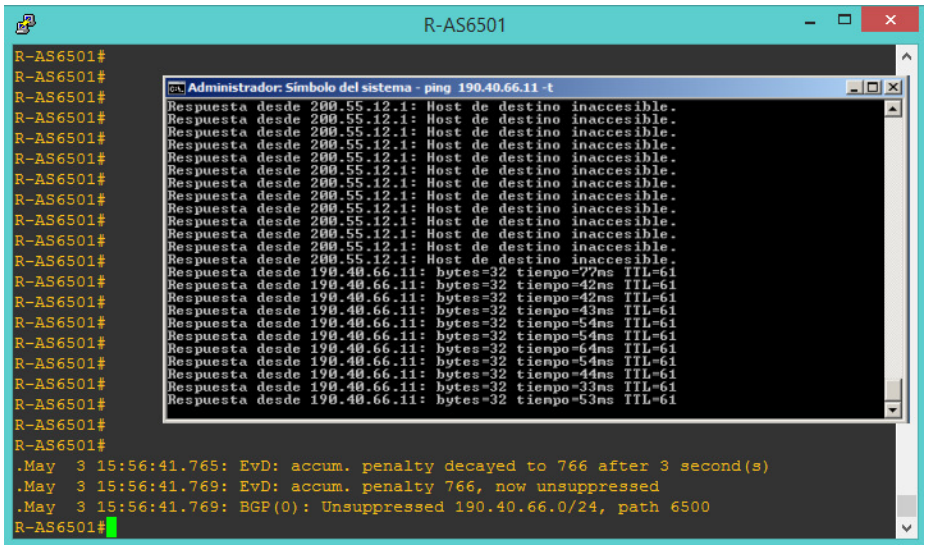


Figura 6.4.9: Despenalización de ruta en R-AS6501

Como se puede observar esto ocurre cuando la penalización llega a un valor de 766. Teniendo en cuenta que el índice de penalización está dado en segundos, calculando la diferencia entre ambos valores se puede determinar que la red ha quedado inaccesible por al menos 2664 segundos, lo que equivale a 44 minutos y 24 segundos.

La conclusión final de esta prueba demuestra que la inestabilidad de un vínculo suma a los problemas demostrados en la prueba anterior, los inherentes a la penalización. El uso de BGP en ciertas situaciones puede causar que determinada red pueda quedar inaccesible por un tiempo superior a los ***44 minutos.***

6.5 Tercer escenario de Pruebas: Administración de tráfico de red utilizando DNS y chequeos de disponibilidad

El presente escenario de pruebas, pretende demostrar que es posible tomar decisiones de convergencia de tráfico sin utilizar protocolos de ruteo del modo en que han sido empleados en las pruebas anteriores. Para la siguiente prueba, se utilizarán chequeos de disponibilidad con el fin de determinar el estado de la red y los servicios; y el protocolo DNS para cambiar los puntos de referencia que permiten localizar recursos en la red. En conjunto, ambas tecnologías interactuarán con el fin de brindar una solución automatizada de convergencia de tráfico de modo similar a lo que ocurre con el protocolo de ruteo BGP. Para realizar la funcionalidad de chequeos y resolución de nombres, se utilizará la herramienta desarrollada durante el presente trabajo, a la cual se le dio el nombre de "Traffic Manager".

Con el fin de poder comparar métricas bajo situaciones similares, se reutilizó la topología de los ejemplos anteriores y se realizaron ciertos cambios. La topología modificada puede apreciarse en la siguiente figura:

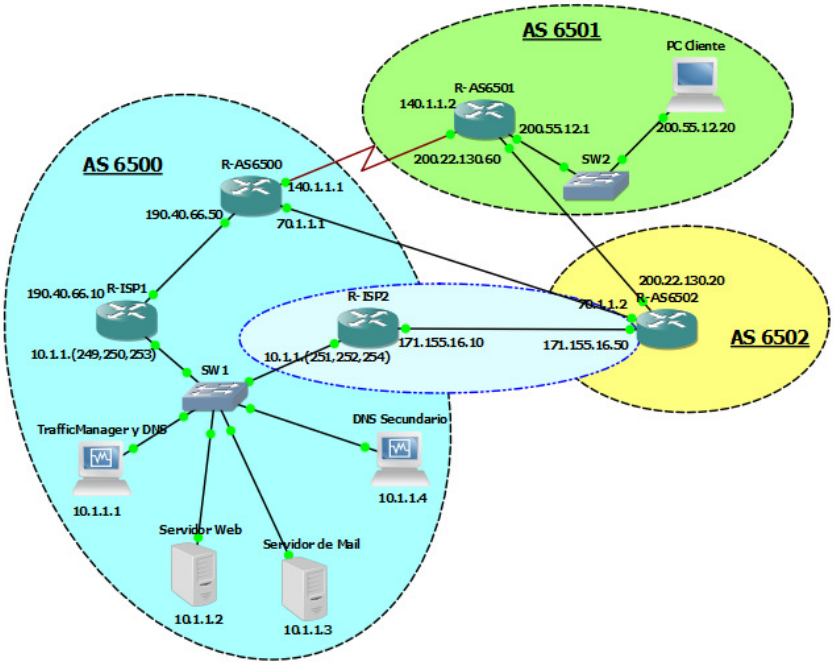


Figura 6.5.1: Topología de red utilizada en el tercer escenario de prueba con la herramienta Traffic Manager

En la gráfica de la figura **6.5.1** puede observarse que a la red **10.1.1.0/24** que hospeda los servidores utilizados para la realización de las pruebas, se agregó un nuevo router con nombre **R-ISP2** (gráfica coloreada con celeste claro y puntos azules). El mismo permite rutear el tráfico que le llega, a través del enlace que lo conecta al sistema autónomo 6502. Si bien la red de servidores forma parte del sistema autónomo 6500, el diálogo de BGP solo lo realiza el router de borde de la red denotado bajo el nombre **R-AS6500**. El presente ejemplo sigue el mismo esquema, en el cual los routers **R-ISP1** y **R-ISP2** no dialogan

bajo ningún protocolo de ruteo con otros routers de la topología planteada. Ambos equipos proveen conectividad de red y tienen configuradas rutas estáticas por defecto hacia **R-AS6500** y **R-AS6502** respectivamente.

De modo similar a la configuración realizada en el router **R-ISP1**, el nuevo router ha sido configurado con la opción de NAT para que redirecciones las conexiones recibidas en sus direcciones IP públicas hacia los servidores internos. Las traducciones configuradas son:

IP Externa (R-ISP2)	IP servidor Interno	Protocolos
171.155.16.11	10.1.1.1	Todos
171.155.16.12	10.1.1.2	Todos
171.155.16.13	10.1.1.3	Todos
171.155.16.14	10.1.1.4	Todos

Cuadro 6.5.1: Traslaciones de red (NAT) configurados para el tercer escenario de pruebas en router **R-ISP2**

Adicionalmente y con el fin de mostrar servicios de ejemplo se agregaron a la topología de red 2 (dos) servidores. Los mismos están representados por las direcciones IP **10.1.1.2** y **10.1.1.3** en la red interna y prestarán el servicio **HTTP** y **SMTP** respectivamente.

La primera particularidad que presenta este tipo de topología es el encaminamiento de datos desde la red de servidores hacia otras redes. En la lógica de ruteo, cuando un equipo necesita comunicarse con otro que no se encuentra en la misma red, debe enviar el paquete de datos a través del equipo intermedio que determine su tabla de ruteo. Este último equipo se conoce como router y generalmente se condice con la puerta de enlace por defecto que tiene configurada el cliente. La principal función de los routers es interconectar redes entre sí. Por esta razón es que generalmente poseen más de una interfaz conectadas a distintas redes. En la presente topología los equipos de la red **10.1.1.0/24** pueden canalizar sus requerimientos a otras redes a través del router **R-ISP1** o **R-ISP2**, ya que ambos poseen conexiones con otras redes y aceptarán el encaminamiento de los datos. Esta decisión dependerá de la configuración realizada en cada equipo dentro de su tabla de ruteo. Par el caso de prueba planteado, todos los equipos de la red poseen configurada su ruta por defecto a través del nuevo router agregado a la topología **R-ISP2**, cuya dirección IP es **10.1.1.254**. Solo el equipo "**TrafficManager y DNS**" cuya dirección IP es **10.1.1.1** tiene configurada su ruta por defecto a través del router **R-ISP1** con IP **10.1.1.253**.

Del mismo modo que los servidores pueden acceder a otras redes a través de ambos routers, ocurre que los clientes de otras redes pueden acceder ahora a los servidores a través de **R-ISP1** o **R-ISP2**. Esto último ocurre debido a que en ambos routers se encuentran configuradas las traducciones de red (NAT), que permiten que desde una red externa (por ejemplo **200.55.12.0/24** perteneciente al sistema autónomo 6501) se pueda acceder a un servidor de la red interna utilizando las IP públicas configuradas para tal fin. Es así por ejemplo que cuando el cliente en el sistema autónomo 6501 con IP **200.55.12.20** desee conectarse con el servidor web cuya IP privada es **10.1.1.2**, podrá utilizar la redirección configurada en el router **R-ISP2** en la IP **171.155.16.12**. Cuando el router **R-ISP2** reciba un paquete cuya dirección destino sea **171.155.16.12**, verificará su tabla de traducciones y determinará que debe conmutar el paquete hacia la IP **10.1.1.2** cambiando la dirección de destino a esta última. Cuando el paquete llegue al equipo de destino, será procesado y posteriormente conmutado de vuelta al destino en base a la información que se encuentre

en la tabla de ruteo del servidor. La misma lógica ocurrirá en el caso que el cliente con IP **200.55.12.20** quiera conectarse con el servidor web, pero esta vez utilizando la traducción configurada en el router **R-ISP1**. Para este caso, el cliente generará un paquete cuyo destino será la IP **190.40.66.12**, el cual será aceptado por el router **R-ISP1** y conmutado al servidor de destino. El inconveniente que ocurrirá en este caso es que como el plano de traducción de direcciones opera separado del plano de ruteo, cuando el cliente responda el requerimiento, lo hará a través de su ruta por defecto. Esta configuración determinará que la conexión que fue recibida a través del router **R-ISP1**, será respondida a través del router **R-ISP2**. En muchas configuraciones esta comunicación funcionará sin ningún tipo de inconvenientes. Ante la eventual caída del vínculo de comunicaciones del router **R-ISP2**, ocurrirá que los paquetes desde el cliente hacia el servidor llegarán correctamente, pero las respuestas no debido a que siempre son canalizadas a través del router **R-ISP2**. Esta configuración no permitirá que puedan utilizarse ambos vínculos de datos uno como respaldo del otro.

Con el fin de subsanar el inconveniente anterior, existen diversas herramientas y alternativas. La utilización de cada una dependerá de los dispositivos de comunicación que se utilicen y sus capacidades. Una posible solución podría ser la utilización de ruteo dinámico basado en origen. Para ello, el switch que interconecta los routers con los servidores podría aportar la inteligencia para determinar que cierto flujo de datos siempre sea respondido utilizando el mismo camino por el que arribó. Esta configuración puede resultar compleja y no todos los switches la soportan, debido a los requerimientos de procesamiento y memoria que implican mantener actualizadas las tablas de estado para las conversaciones. Con el fin de adoptar una tecnología que esté presente en la mayoría de los dispositivos, se optó la técnica de traducción de direcciones (NAT). Esta misma técnica se utilizó para la publicación de servicios de la red interna hacia otras redes. En esta nueva configuración, cuando un paquete atraviesa el router hacia un determinado servidor, además de cambiar la dirección IP de destino, el router también cambiará la dirección IP origen por una dirección dentro de la misma red en la que se encuentra el servidor. De este modo, cuando el servidor tenga que devolver su respuesta, lo hará a esta última IP correspondiente al router por el cual ingresó el requerimiento. Cuando el router reciba la respuesta, cambiará las direcciones IP de origen y destino por la dirección pública utilizada para acceder al router y la del cliente. La siguiente figura muestra la variación de las direcciones IP origen y destino en cada salto de la red tanto para la consulta como para la respuesta en una comunicación típica entre la PC con IP **200.55.12.20** y el servidor con IP **10.1.1.2** accesible a través de la IP pública del router **R-ISP1**:

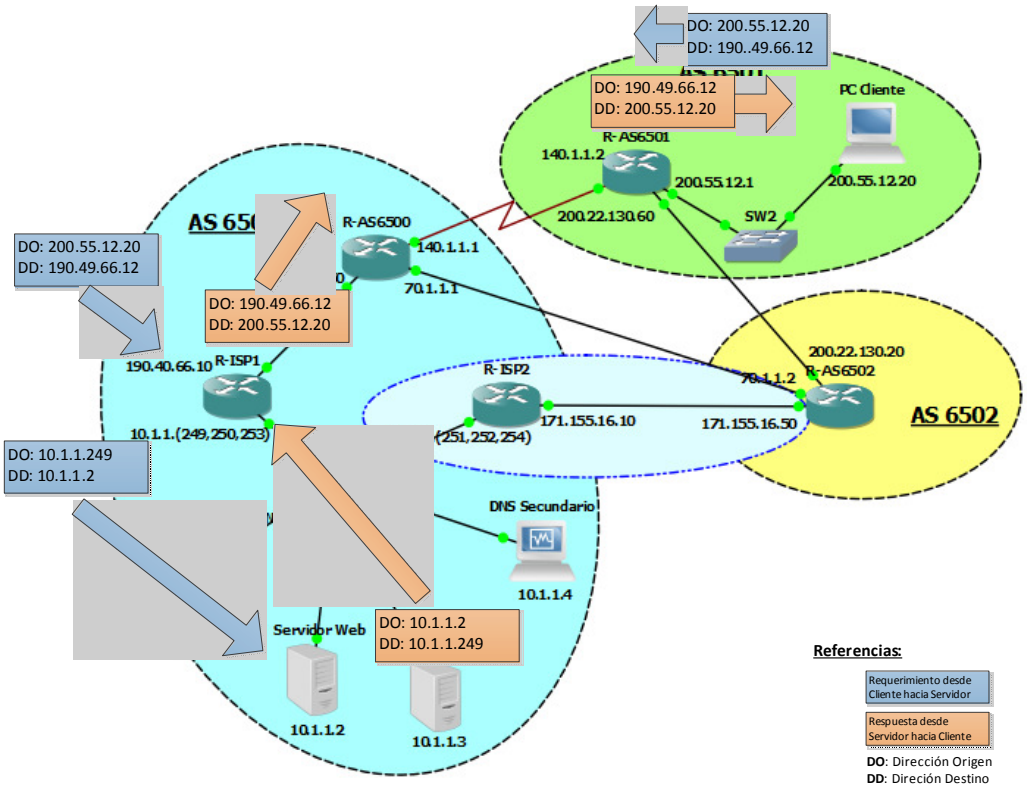


Figura 6.5.2: Acceso desde la PC Cliente al servidor Web a través de R-ISP1. Direccionamiento afectado por el NAT

Es importante destacar que esta última configuración, provoca que desde el servidor de destino no se pueda conocer la dirección IP original con la que el cliente realizó el requerimiento. Bajo esta metodología de acceso, el servidor siempre verá los requerimientos que a él lleguen a través del router **R-ISP1**, con IP origen **10.1.1.249** o **10.1.1.250**. Debido a esta situación será deseable que los requerimientos ingresen a través del router **R-ISP2**. Esto se debe a que es el dispositivo configurado como ruta por defecto en los servidores y por consiguiente no es necesario enmascarar la dirección IP origen de los paquetes, ya que las respuestas serán canalizadas naturalmente a través de él. Esta premisa determina que el servidor "TrafficManager y DNS", deben ser ubicados estratégicamente sobre el vínculo de contingencia o secundario. Adicionalmente, los servidores que brindan servicios hacia los clientes deben ser configurados con su puerta de enlace apuntando hacia el router **R-ISP2**. Esta última configuración permitirá evitar que los accesos lleguen hacia el servidor con la dirección IP origen cambiada. De respetar esta premisa, no existirá la necesidad de configurar un NAT reverso cambiando la dirección IP origen para todos los requerimientos que lleguen a la red interna, ya que siempre serán respondidos a través de este último equipo. La siguiente tabla muestra las traducciones reversas que ha sido necesario configurar en cada router:

Router	Dirección ip externa del router	NAT reverso
R-ISP1	190.40.66.12	10.1.1.249 o 10.1.1.250
R-ISP1	190.40.66.13	10.1.1.249 o 10.1.1.250
R-ISP1	190.40.66.14	10.1.1.249 o 10.1.1.250
R-ISP2	171.155.16.11	10.1.1.251 o 10.1.1.252

Cuadro 6.5.2: Traslaciones de red (NAT) reversos configurados para el tercer escenario de pruebas en los router **R-ISP1** y **R-ISP2**

A continuación se transcribe la configuración realizada sobre los routers **R-ISP1** y **R-ISP2**. Solo se transcribe la configuración de estos últimos debido a que el resto del equipamiento utilizado mantiene las configuraciones de las pruebas anteriores. El equipo **R-AS6502** también tuvo que ser reconfigurado con el fin de interconectarlo con el equipo **R-ISP2**. Para ello solo se agregó y configuró una nueva interfaz de red con el direccionamiento **171.155.16.50** y máscara de red **255.255.255.0**. Debido a que esta configuración ya ha sido mostrada en ejemplos anteriores, no se transcribe nuevamente. La configuración realizada sobre el equipo **R-ISP1** se muestra en el siguiente cuadro:

```
hostname R-ISP1

interface FastEthernet0/0

description ###Hacia la WAN para publicar servicios

ip address 190.40.66.11 255.255.255.0 secondary

ip address 190.40.66.14 255.255.255.0 secondary

ip address 190.40.66.12 255.255.255.0 secondary

ip address 190.40.66.13 255.255.255.0 secondary

ip address 190.40.66.10 255.255.255.0

ip nat outside

duplex auto

speed auto

interface FastEthernet0/1

description ###Hacia la LAN

ip address 10.1.1.253 255.255.255.0

ip nat inside

duplex auto

speed auto

ip route 0.0.0.0 0.0.0.0 190.40.66.50

ip route 10.1.1.249 255.255.255.255 FastEthernet0/0

ip route 10.1.1.250 255.255.255.255 FastEthernet0/0

ip nat pool local-249-250 10.1.1.249 10.1.1.250 netmask 255.255.255.0

ip nat inside source static 10.1.1.1 190.40.66.11 no-payload
```

```
ip nat inside source static 10.1.1.2 190.40.66.12

ip nat inside source static 10.1.1.3 190.40.66.13

ip nat inside source static 10.1.1.4 190.40.66.14 no-payload

ip nat outside source list 101 pool local-249-250

access-list 101 permit ip any host 190.40.66.14

access-list 101 permit ip any host 190.40.66.12

access-list 101 permit ip any host 190.40.66.13
```

Cuadro 6.5.3: Configuración del router **R-ISP1** para el tercer escenario de pruebas

La configuración realizada sobre el router R-ISP2 se muestra en el siguiente cuadro:

```
hostname R-ISP2

interface FastEthernet0/0

description ###Hacia R-AS6502

ip address 171.155.16.11 255.255.255.0 secondary

ip address 171.155.16.12 255.255.255.0 secondary

ip address 171.155.16.13 255.255.255.0 secondary

ip address 171.155.16.14 255.255.255.0 secondary

ip address 171.155.16.10 255.255.255.0

ip nat outside

duplex auto

speed auto

interface FastEthernet0/1

description ###Hacia la Lan

ip address 10.1.1.254 255.255.255.0

ip nat inside

duplex auto

speed auto
```

```
ip route 0.0.0.0 0.0.0.0 171.155.16.50

ip route 10.1.1.251 255.255.255.255 FastEthernet0/0

ip route 10.1.1.252 255.255.255.255 FastEthernet0/0

ip nat pool local-251-252 10.1.1.251 10.1.1.252 netmask 255.255.255.0

ip nat inside source static 10.1.1.1 171.155.16.11 no-payload

ip nat inside source static 10.1.1.2 171.155.16.12

ip nat inside source static 10.1.1.3 171.155.16.13

ip nat inside source static 10.1.1.4 171.155.16.14 no-payload

ip nat outside source list 101 pool local-251-252

access-list 101 permit ip any host 171.155.16.11
```

Cuadro 6.5.4: Configuración del router **R-ISP2** para el tercer escenario de pruebas

Para la realización de la primera prueba en la topología planteada, y con el fin de poder utilizar los mismos parámetros que en las pruebas anteriores, se utilizará el protocolo ICMP. Desde la "PC Cliente" en el sistema autónomo 6501 se realizarán consultas hacia el equipo "Servidor Web" que se encuentra en la red privada del sistema autónomo 6500. De modo similar a las pruebas anteriores, se utilizarán nombres de DNS con el fin de referenciar al servidor web; ya que lo que se pretende mostrar es que es posible cambiar un registro de resolución de nombres en breves lapsos de tiempo. A diferencia de las pruebas anteriores, se lanzarán reiteradas pruebas de ICMP enviando un único paquete en cada una y a intervalos de 1 segundo. El cambio en el mecanismo, se debe a que cuando se realizan pruebas utilizando ICMP, lo primero que hace el equipo cliente es traducir el nombre indicado a determinada dirección IP y el proceso de resolución no vuelve a ocurrir hasta que se realice una nueva prueba de ICMP. Por esta última razón, la prueba de ICMP podría correr indefinidamente, aun cuando la solución ya haya convergido hacia la nueva dirección IP. La instanciación de reiteradas pruebas causará que la resolución de nombres pueda volver a realizarse cuando el registro resuelto expire, y por tal motivo se pueda resolver a una nueva dirección IP.

Como en los casos anteriores, se ha utilizado el dominio **ejemplo.com.ar**, el cual ha sido configurado en los servidores DNS de la red privada de la topología. El servidor con dirección IP **10.1.1.1** es el servidor DNS primario y autoritativo para el dominio **ejemplo.com.ar**. El mismo es accesible a través de la traslación de direcciones configurada en el router **R-ISP1**. La IP pública que se utiliza para alcanzar al servidor DNS desde otras redes es **190.40.66.11**. El servidor secundario y autoritativo para el dominio de DNS que se utilizará, también se encuentra en la red interna y su dirección IP es **10.1.1.4**. El mismo es

accesible a través de la traslación de direcciones configurada en el router **R-ISP2**. La dirección IP que se utiliza para alcanzar al servidor desde otras redes es **171.155.16.14**. El hecho de que los servidores sean accesibles a través de enlaces de InterNet distintos, permite que ante la falla en uno de ellos, el otro pueda seguir respondiendo a los requerimientos que realicen los clientes externos.

El equipo "Servidor web", se dio de alta en la herramienta de administración de tráfico basada en DNS y chequeos de disponibilidad, bajo el nombre **www**, dentro del dominio **ejemplo.com.ar**. Se define como enlace primario por el cual operará el servidor en condiciones normales, al provisto a través del equipo **R-ISP2**. Esto significa que el registro de DNS **www.ejemplo.com.ar** resolverá la dirección IP **171.155.16.12** en condiciones normales. Ante una condición de convergencia, se esperará que la herramienta Traffic Manager cambie el registro de resolución de DNS a la dirección IP **190.40.66.12** correspondiente a la redirección configurada en **R-ISP1**. La siguiente figura muestra los datos configurados en la herramienta para el alta del Servidor Web:

Figura 6.5.3: Alta de Host www.ejemplo.com.ar en la herramienta Traffic Manager

Como se puede observar en la figura, la dirección IP que será utilizada para resolver el nombre **www.ejemplo.com.ar** es **171.155.16.12**. El servicio que será utilizado para realizar el monitoreo por parte de la herramienta Traffic Manager es HTTP hacia la misma dirección IP. El monitoreo se realizará desde el servidor "TrafficManager y DNS" de la topología, utilizando el vínculo de datos externo. Esto implica que los chequeos atravesarán los sistemas autónomos 6501 y/o 6502 desde la dirección IP configurada como NAT para el servidor (**190.40.66.11**) y hasta la IP de destino. De este modo, se podrán detectar fallas en el servicio primario del mismo modo que afectarían a un cliente externo del mismo. Los distintos escenarios de fallas que es capaz de controlar la herramienta se discuten más adelante en la sección 6.5 del presente capítulo. Adicionalmente, el servidor Traffic Manager monitoreará la dirección IP secundaria o de contingencia (**190.40.66.50**) utilizando el protocolo ICMP. En caso de detectar una caída en el monitoreo a la dirección IP primaria, se realizará la convergencia hacia la dirección IP secundaria para DNS (**190.40.66.12**), siempre y cuando el chequeo del servicio de contingencia se encuentre operativo. Es importante destacar que las direcciones IP que se utilizan para la realización de los chequeos pueden diferir de las direcciones IP que se utilizan para realizar la resolución de nombres de registros DNS. Esto es debido a que en ciertas situaciones puede resultar preferible realizar un chequeo a determinado punto de la red con el fin de obtener datos más reales acerca del estado de la infraestructura.

Para poder analizar los tiempos de convergencia se creó un programa sencillo, que a intervalos de 1 segundo realiza una prueba utilizando el protocolo ICMP hacia el servidor registrado bajo el nombre www.ejemplo.com.ar desde la "PC Cliente". Adicionalmente, el programa imprime en pantalla la hora actual de ejecución para cada instancia de prueba ICMP, con el fin de determinar a partir de qué momento el servicio dejó de responder y a partir de que instante vuelve a responder el mismo. Las siguientes pruebas utilizarán como referencia la hora que se imprime en cada registro solo a fines de poder medir intervalos de tiempo. Previo a la realización de las pruebas, todos los relojes fueron sincronizados con el servidor "TrafficManager y DNS" que actúa como servidor horario de la solución. Esto último permitirá comparar las horas de los distintos componentes que forman parte de la topología, sabiendo que no existe variación horaria. El siguiente cuadro muestra el programa por lotes escrito para ser ejecutado bajo la línea de comandos del sistema operativo Windows:

```
@echo off

setlocal enabledelayedexpansion

:loop

set TIMESTAMP=%TIME:~0,2%_%TIME:~3,2%_%TIME:~6,2%

echo The local time is %TIMESTAMP%

ping –n 1 –w 1 www.ejemplo.com.ar

timeout /t 1

goto loop
```

Cuadro 6.5.5: Programa para la realización de chequeos en Windows para el tercer escenario de pruebas

En la siguiente figura pueden visualizarse solo algunas instancias de la ejecución del programa. Por cuestiones de espacio y para no reiterar información, los resultados de las ejecuciones intermedias han sido suprimidos. En la misma se puede verificar que el servidor **www.ejemplo.com.ar** responde correctamente a los requerimientos ICMP utilizando su IP primaria (**170.155.17.12**) a la hora **12:25:08** de la ejecución. Posteriormente puede visualizarse el momento en el que se pierde conexión con el servidor de destino, luego de inyectar una falla en la red simulando la caída del vínculo de comunicaciones (hora **12:25:09**). Para la generación de la falla, se procedió a bajar la interfaz **FastEthernet0/0** del router **R-ISP2**, a través del comando **shutdown**. En la parte inferior de la gráfica, puede visualizarse que aún en el instante de la hora **12:26:06**, la comunicación no se ha reestablecido. Posteriormente se puede apreciar a la hora **12:26:07** que la resolución del nombre www.ejemplo.com.ar responde utilizando la nueva dirección IP (**190.40.66.12**), lo cual permite que los requerimientos enviados desde el cliente vuelvan a tener respuesta por parte del servidor:

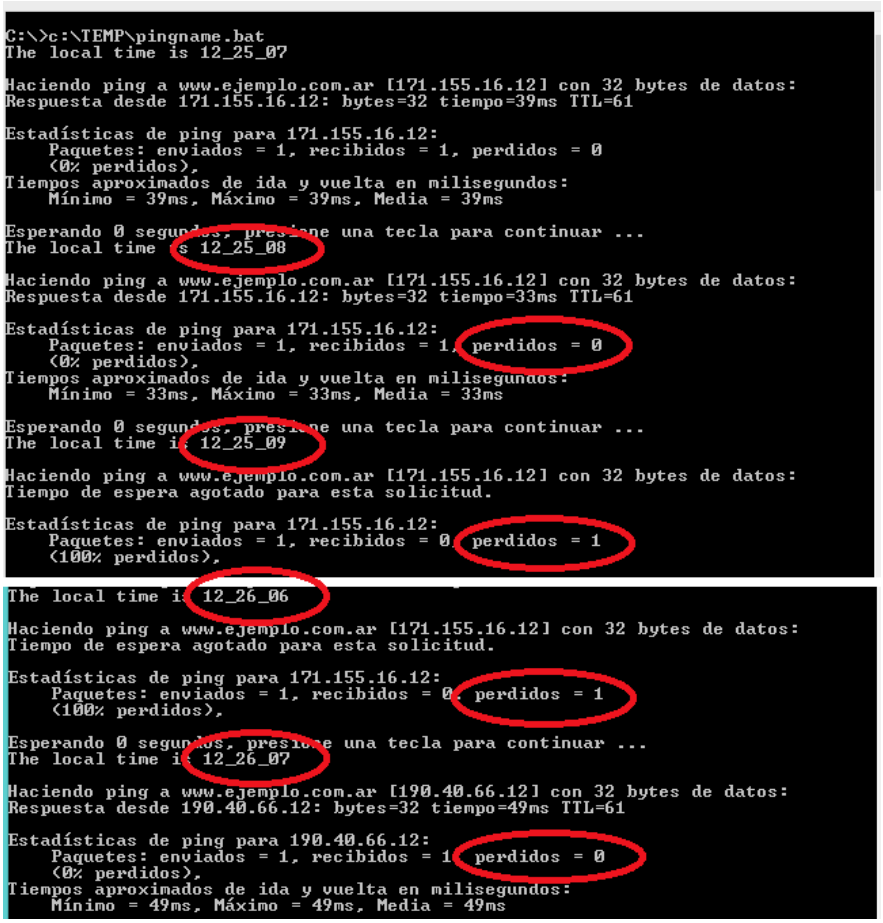
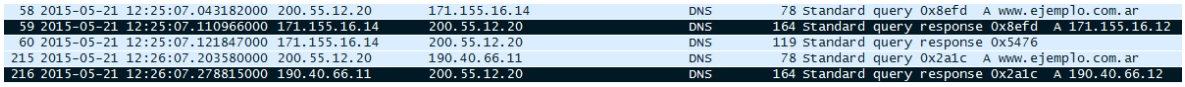


Figura 6.5.4: Resultado de la ejecución del programa creado para el tercer escenario de pruebas

Previo a la invocación del programa de prueba utilizado, se inició una captura de tráfico sobre la interfaz de red de la PC Cliente con el fin de visualizar las consultas de DNS realizadas. A continuación se muestra un extracto de la captura del tráfico, donde pueden visualizarse las consultas de DNS realizadas por parte del cliente con el fin de resolver el nombre **www.ejemplo.com.ar**. Se puede apreciar que en una primera instancia, la respuesta devuelta por parte del servidor primario para el dominio indica que la dirección a la que resuelve dicho nombre es **171.155.16.12**. Posteriormente y luego de 60 segundos, una nueva consulta al servidor autoritativo devuelve la nueva IP **190.40.66.12**. El hecho de que la segunda consulta se realice luego de 60 segundos, responde a que el tiempo de vida (TTL) configurado para los registros de DNS gestionados a través de la herramienta Traffic Manager se ha establecido en un valor de 60 segundos. Al detectar la expiración del registro, el cliente vuelve a realizar una nueva consulta al servidor DNS para el nombre **www.ejemplo.com.ar**:



58	2015-05-21 12:25:07.043182000	200.55.12.20	171.155.16.14	DNS	78 Standard query 0x8efd A www.ejemplo.com.ar
59	2015-05-21 12:25:07.110966000	171.155.16.14	200.55.12.20	DNS	164 Standard query response 0x8efd A 171.155.16.12
60	2015-05-21 12:25:07.121847000	171.155.16.14	200.55.12.20	DNS	119 Standard query response 0x5476
215	2015-05-21 12:26:07.203580000	200.55.12.20	190.40.66.11	DNS	78 Standard query 0x2a1c A www.ejemplo.com.ar
216	2015-05-21 12:26:07.278815000	190.40.66.11	200.55.12.20	DNS	164 Standard query response 0x2a1c A 190.40.66.12

Figura 6.5.5: Captura de tráfico realizada para el tercer escenario de pruebas

La responsabilidad de la detección y cambio en los registros de resolución, es de la herramienta Traffic Manager, la cual detecta la caída de servicio sobre la IP primaria para monitoreo y dispara la convergencia. La información en la herramienta, es registrada para los administradores con referencias horarias y estado de los chequeos realizados como puede verse en la siguiente figura:

Logs

Fecha y Hora	Origen	Modulo	Datos
21/05/15 - 12:25:47	cli	em	Se va a Failover en el HOST www.ejemplo.com.ar. IP de Failover: 190.40.66.12. Estado de HOST principal: DOWN/HARD
21/05/15 - 12:25:47	cli	em	Cambio de Estado en www.ejemplo.com.ar. IP: 171.155.16.12. Failover IP: 190.40.66.12. Estado: DOWN/HARD
21/05/15 - 12:25:27	cli	em	Se detecto una caida en el HOST www.ejemplo.com.ar, del tipo SOFT. Se continuan los chequeos para determinar si es definitiva. No se toma ninguina accion.
21/05/15 - 12:25:27	cli	em	Cambio de Estado en www.ejemplo.com.ar. IP: 171.155.16.12. Failover IP: 190.40.66.12. Estado: DOWN/SOFT

Figura 6.5.6: Visualización de Logs en la herramienta Traffic Manager para el tercer escenario de pruebas

Es importante notar que la caída fue detectada por la herramienta de monitoreo a las **12:25:27** horas, mientras que la PC Cliente perdió conectividad de red a las **12:25:09**. Esta primera diferencia de tiempo implica una demora en la detección de la falla de 18 segundos. Cuando se dan de alta los servicios en la herramienta Traffic Manager se debe determinar el intervalo entre los chequeos. Cuanto menor sea este valor, más rápida será la detección de las fallas y por consiguiente la diferencia de tiempo anterior tenderá a reducirse en la mayoría de los casos. El hecho de decrementar el intervalo entre los chequeos, implica también que el tráfico de red sea mayor, ya que los mismos se realizarán a intervalos menores. Como puede visualizarse en la figura 6.5.6, la decisión de convergencia se toma a las **12:25:47**, luego de haber confirmado en 2 (dos) instancias que el servicio se encuentra caído. Recién en ese instante, la herramienta toma la decisión de cambiar el registro **www** en el servidor DNS, de modo tal que resuelva con la IP de failover. Debido a que la PC Cliente ha realizado la resolución de nombres a las **12:25:07**, de acuerdo a la figura 6.4.5, y que el tiempo de vida del registro es de 60 segundos, la PC no volverá a realizar una nueva consulta hasta la hora **12:26:07**. Es en ese instante en el cual resuelve el nombre con la nueva dirección IP y la comunicación se reestablece.

Del análisis de los resultados, surge que desde el instante en el que se detectó la primera caída del servicio por parte de la PC cliente a las **12:25:09** horas, hasta el momento en que la herramienta tomó la

decisión de convergencia a las **12:25:47** horas, transcurrieron solo 38 segundos. Cualquier cliente que no hubiera resuelto el nombre **www** en el dominio **ejemplo.com.ar**, luego de 38 segundos, accederá directamente a la nueva dirección IP. Esto último no ocurre con la PC cliente, debido a que el registro de DNS aún se encuentra activo en la cache del sistema operativo de la PC cliente.

Otra situación que podría ocurrir es aquella en que el servicio presente una falla justo en el instante en el que la herramienta Traffic Manager debe realizar el chequeo. Bajo esta situación, solo transcurrirán 20 segundos hasta que la herramienta confirme la caída del servicio y converja a la nueva IP. De este modo, la convergencia solo tardaría 20 segundos para aquel cliente que no haya realizado la resolución del nombre de DNS. En el peor de los casos para el cliente que si haya realizado la resolución, la convergencia podría tardar 110 segundos (30 segundos de intervalo entre los chequeos, más 20 segundos de la detección de la falla más 60 segundos hasta la expiración del registro de DNS recientemente resuelto).

Como podrá apreciarse, existen diversos factores que pueden verse involucrados en el cálculo del tiempo, ya que a diferencia de las pruebas anteriores en las que la decisión era tomada en base a la información que se intercambiaba entre los routers; aquí entra en juego la información que posean los clientes. Lo importante a destacar es que desde el momento en que la herramienta Traffic Manager confirma la caída del servicio y notifica al módulo de resolución de nombre, la convergencia se realizará paulatinamente en base a la información que cada cliente posea en su propia base de resolución.

La misma prueba ha sido realizada bajo un sistema operativo Debian GNU/Linux con Kernel 3.12, arrojando los mismos resultados. Esta última prueba permitió determinar que el sistema operativo respeta el tiempo de vida establecido en la repuesta del servidor DNS para los registros DNS del mismo modo que lo hace el sistema operativo Windows. El script ejecutado fue escrito en bash y se muestra en el siguiente cuadro:

```
#!/bin/bash

while true;

do

date

ping -c 1 -w 1 www.ejemplo.com.ar

sleep 1

done
```

Cuadro 6.5.6: Programa para la realización de chequeos en GNU/Linux para el tercer escenario de pruebas

Otro aspecto a tener en cuenta como se mencionó en capítulos anteriores, es que las distintas aplicaciones pueden tener su propia cache para la resolución de nombres con el fin de acelerar su funcionamiento. Cuando una aplicación necesita realizar una resolución de nombres, solicitan dicho servicio al sistema operativo. El sistema operativo primero verifica si el registro solicitado se encuentra su propia cache de resolución y posee un tiempo de vida válido. Cuando esto último ocurre, el sistema operativo devuelve directamente la respuesta a la aplicación sin la necesidad de realizar la consulta a los servidores DNS correspondientes. Esta última acción permite acelerar el proceso de resolución de nombres, dado que no es necesario esperar la respuesta de un servidor externo. Las aplicaciones también pueden poseer su propia base de datos de resolución, donde almacenan las resoluciones realizadas recientemente. Las aplicaciones que necesitan brindar altos tiempos de respuesta al usuario, utilizan esa técnica. Los navegadores web, generalmente implementan su propia cache de resolución con el fin de no realizar solicitudes de resolución reiteradas al sistema operativo que pudieran incurrir en un tiempo extra para brindar los resultados al usuario. En las pruebas planteadas, se ha podido comprobar que los sistemas operativos Windows y GNU/Linux respetan el tiempo de vida de los registros resueltos. A continuación se realizará una prueba similar con distintos navegadores web, que permitan determinar cuál es el período de vida que los mismos utilizan para almacenar los registros resueltos en sus propias cache de resolución. En el capítulo 3.8 del presente trabajo, se estudió que las aplicaciones utilizan funciones para comunicarse con el sistema operativo y solicitar resultados al proceso de resolución de nombres. En la misma sección se observó que las funciones utilizadas no exponen a las aplicaciones el tiempo de vida de los registros de DNS. Por esta misma razón, las aplicaciones que implementen el concepto de caching deben seleccionar un valor arbitrario de tiempo para almacenar los registros en sus propias bases de datos temporales.

Para esta última comprobación se realizaran requerimientos HTTP al sitio **www.ejemplo.com.ar** desde distintos navegadores, mientras se realizan capturas de tráfico de red con el fin de comprobar las consultas DNS realizadas por parte de la PC Cliente hacia los servidores autoritativos para el dominio. Los servidores DNS han sido configurados para devolver sus respuestas con un tiempo de vida de 60 segundos. En base a dicha información y las capturas de tráfico realizadas se intentará determinar cuál es el lapso de tiempo por el que los navegadores almacenan los registros resueltos en sus caches de resolución.

Con el fin de verificar el comportamiento del navegador web **Google Chrome**, se realizaron sucesivos requerimientos al servidor **www.ejemplo.com.ar** durante 6 minutos. La versión del navegador utilizada para la prueba es 40.0 ejecutándose sobre un sistema operativo **Windows 8**.

4	2015-05-22 16:08:43.461840000	200.55.12.20	171.155.16.14	DNS	78	Standard query 0xa4e5 A www.ejemplo.com.ar
5	2015-05-22 16:08:43.524399000	171.155.16.14	200.55.12.20	DNS	164	Standard query response 0xa4e5 A 171.155.16.12
118	2015-05-22 16:09:44.816795000	200.55.12.20	171.155.16.14	DNS	78	Standard query 0xe437 A www.ejemplo.com.ar
120	2015-05-22 16:09:44.878892000	171.155.16.14	200.55.12.20	DNS	164	Standard query response 0xe437 A 171.155.16.12
202	2015-05-22 16:10:46.939641000	200.55.12.20	171.155.16.14	DNS	78	Standard query 0x5e1a A www.ejemplo.com.ar
204	2015-05-22 16:10:47.000691000	171.155.16.14	200.55.12.20	DNS	164	Standard query response 0x5e1a A 171.155.16.12
325	2015-05-22 16:11:48.939642000	200.55.12.20	171.155.16.14	DNS	78	Standard query 0x5234 A www.ejemplo.com.ar
327	2015-05-22 16:11:49.002070000	171.155.16.14	200.55.12.20	DNS	164	Standard query response 0x5234 A 171.155.16.12
402	2015-05-22 16:13:09.600825000	200.55.12.20	171.155.16.14	DNS	78	Standard query 0x99e6 A www.ejemplo.com.ar
403	2015-05-22 16:13:10.601060000	200.55.12.20	190.40.66.11	DNS	78	Standard query 0x99e6 A www.ejemplo.com.ar
405	2015-05-22 16:13:10.674271000	190.40.66.11	200.55.12.20	DNS	164	Standard query response 0x99e6 A 171.155.16.12
460	2015-05-22 16:14:10.263551000	200.55.12.20	190.40.66.11	DNS	78	Standard query 0x10db A www.ejemplo.com.ar
461	2015-05-22 16:14:10.322131000	190.40.66.11	200.55.12.20	DNS	164	Standard query response 0x10db A 190.40.66.12

Figura 6.5.7: Captura de tráfico DNS para el navegador Google Chrome

En la figura puede apreciarse de acuerdo al horario en el que se realizan las consultas, que Google Chrome utiliza un valor de 60 segundos para almacenar los registros resueltos. Inclusive, en la última respuesta puede visualizarse como luego de inyectar una falla en la red del mismo modo que se realizó en los casos anteriores, la solución ha convergido y la respuesta de DNS es la dirección IP de failover a la cual realiza la conexión el navegador. Adicionalmente puede verse en la cache interna de resolución del navegador, que ante cada consulta realizada el tiempo de expiración del registro se define a 60 segundos posteriores a haber realizado la consulta. La siguiente gráfica muestra los extractos de la consulta a la cache de DNS de Google Chrome:

Hostname	Family	Addresses	Expires
www.ejemplo.com.ar	IPV4	171.155.16.12	2015-05-22 16:09:43.246

Hostname	Family	Addresses	Expires
www.ejemplo.com.ar	IPV4	171.155.16.12	2015-05-22 16:10:44.601

Hostname	Family	Addresses	Expires
www.ejemplo.com.ar	IPV4	171.155.16.12	2015-05-22 16:11:46.723

Hostname	Family	Addresses	Expires
www.ejemplo.com.ar	IPV4	171.155.16.12	2015-05-22 16:12:48.724

Hostname	Family	Addresses	Expires
www.ejemplo.com.ar	IPV4	190.40.66.12	2015-05-22 16:15:10.044

Figura 6.5.8: Cache de resolución de registros DNS del navegador Google Chrome (**chrome://net-internals#dns**)

De acuerdo a la información publicada por W3Counter en su sitio de estadísticas en la URL http://www.w3counter.com/globalstats.php, el segundo navegador más utilizado globalmente para la navegación de sitios web es Internet Explorer de la empresa Microsoft. A continuación se puede visualizar la captura de tráfico DNS para requerimientos sucesivos al servidor www.ejemplo.com.ar durante 42 minutos utilizando el navegador **Internet Explorer versión 11.0 en sistema operativo Windows 8**:

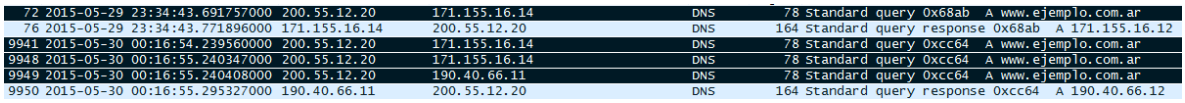


Figura 6.5.9: Captura de tráfico DNS para el navegador Internet Explorer

En la figura puede apreciarse que la primera consulta al servidor DNS fue realizada a las **23:34:43** horas. A partir de ese instante, se realizaron repetidas actualizaciones del sitio, con el fin de verificar si luego de transcurridos 60 segundos se volvía a realizar una nueva consulta. Esta acción se llevó a cabo durante 41 minutos aproximadamente, corroborándose que el navegador no lanzaba una nueva consulta de DNS. Esta prueba permite suponer que InterNet Explorer cachea las resoluciones de DNS por tiempos mayores a los 30 minutos documentados por el fabricante.

Luego de 42 minutos se optó por inyectar una falla en la red con el fin de verificar el comportamiento del navegador. Para ello se procedió a bajar la interfaz **FastEthernet0/0** del router **R-ISP2**. Luego de realizar dicha acción, se observó que al refrescar nuevamente el contenido de la página, el navegador realizó una nueva consulta de DNS al no poder establecer la conexión hacia la IP **171.155.16.12** que se encontraba en su propia cache de resolución. La última consulta se realizó a las **00:15:55** la cual fue respondida por el único servidor DNS accesible en la red, quien devolvió como resultado la IP de failover **190.40.66.12**. El navegador en ningún momento arrojó errores acerca de la falta de conectividad.

Luego de transcurrir 10 minutos más se procedió a levantar la interfaz del router **R-ISP2** y posteriormente bajar la interfaz **FastEthernet0/0** del router **R-ISP1**. Esta última acción causó que la página web de ejemplo deje de mostrarse. Transcurridos 20 minutos, se pudo visualizar la página nuevamente tras una nueva consulta al servidor DNS quien arrojó como resultado para la consulta al nombre **www.ejemplo.com.ar**, la dirección IP **171.155.16.12**.

Como conclusión y basándose en la documentación publicada por el fabricante, Internet Explorer mantiene su propia cache de resolución respetando el valor configurado de cache de 30 minutos para los registros resueltos. Mientras el sitio siga respondiendo a los requerimientos, el navegador no realizará una nueva consulta de DNS y utilizará la IP que posee almacenada. De producirse un error de conexión transcurridos los 30 minutos, el navegador realizará una nueva consulta y almacenará el resultado por 30 minutos más. Si volvieran a ocurrir errores, dentro del periodo de los 30 minutos de haber resuelto el nombre de DNS, no se volverá a lanzar una nueva consulta, a menos que se cierre el navegador y se vuelva a abrir, lo cual causa el vaciamiento de la cache en todos los casos.

Por último, se realizaron pruebas utilizando el navegador **Mozilla Firefox versión 38.01** en sistema operativo **Windows 8**. Los resultados de la captura de tráfico DNS para requerimientos sucesivos al servidor www.ejemplo.com.ar durante 3 (tres) minutos pueden verse en la siguiente figura:

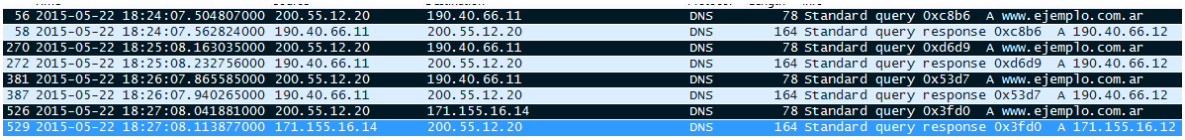


Figura 6.5.10: Captura de tráfico DNS para el navegador Mozilla Firefox

La captura de datos realizada, demuestra que el navegador define el tiempo de vida de un registro de DNS resuelto en 60 segundos. De hecho, el equipo de desarrollo ha implementado en el navegador una opción configurable que determina cuál es el tiempo de vida por el que debe almacenarse un registro de DNS en la cache local del navegador. El valor configurado por defecto en la opción **Network.dnsCacheExpiration** está fijado a 60 segundos. El siguiente artículo define el concepto: http://kb.mozillazine.org/Network.dnsCacheExpiration. Una nueva prueba modificando dicho valor a 120 segundos, demostró que el valor configurable se sigue respetando.

Las pruebas bajo el sistema operativo GNU/Linux con Kernel 3.12, han arrojado los mismos resultados para los navegadores Google Chrome y Mozilla Firefox. Por cuestiones de compatibilidad, no se pudieron realizar pruebas con el navegador Internet Explorer en esta última plataforma.

Teniendo en cuenta que el tiempo de vida configurado en el servidor DNS para el registro **www** en la zona **ejemplo.com.ar** fue de 60 segundos; y que los navegadores Google Chrome y Mozilla Firefox utilizan el mismo tiempo para definir el tiempo de vida de los registros resueltos, se optó por realizar una nueva prueba. Esta vez, el servidor DNS fue configurado para devolver los resultados de las consultas realizadas con un valor de TTL de 20 segundos. Luego de repetir las capturas de tráfico para los 3 navegadores web, se pudo observar que el comportamiento era exactamente el mismo. Debido a que los navegadores no expiran los registros resueltos dentro de sus caches temporales hasta que no transcurra el período configurado (60 segundos en Google Chrome y Mozilla Firefox; y 30 minutos en Internet Explorer en caso de producirse un error en la conexión), no se realizan nuevas consultas a través del sistema operativo antes del tiempo configurado. Esta última prueba permitió confirmar los resultados a los que se arribaron en las pruebas anteriores.

La conclusión final de esta prueba determina que el tiempo que se tarda en converger el tráfico de red de un vínculo hacia otro es variable y depende en muchos casos del comportamiento de los clientes y el tipo de aplicación que estén utilizando. El tiempo de detección de falla y convergencia podría tardar entre 20 y 50 segundos, mientras que el tiempo que tarda un cliente en anoticiarse acerca del cambio puede verse afectado por diversos factores. Si el cliente no había realizado la resolución de nombres previo al cambio, la convergencia será automática. Si el cliente estaba accediendo al servicio mientras se producía la falla, el comportamiento dependerá de la aplicación. Si la aplicación no utiliza mecanismos de cache, el tiempo de convergencia podrá variar entre 1 y 60 segundos. Si la aplicación utiliza caché de resolución, el tiempo podrá variar desde 1 segundo hasta el valor configurado o los 30 minutos impuestos por Internet Explorer.

Un último aspecto importante a destacar de la solución, es que como la convergencia la realiza cambiando la dirección IP en un registro de resolución de DNS, aquellas aplicaciones que se encontraban transmitiendo datos sobre una sesión establecida, deberán volver a realizar la conexión para poder continuar con la transferencia.

6.6 Escenarios de Fallas controlados por la herramienta

La arquitectura de red planteada, permite la detección de fallas en diversos escenarios. Con el fin de abarcar la mayor cantidad de casos, es importante respetar la ubicación de las componentes, teniendo en cuenta que el servidor DNS primario y el que realiza los chequeos de disponibilidad debe ser ubicado en el enlace de contingencia o de respaldo, mientras que el servidor DNS secundario debe publicarse sobre el enlace principal.

Las posibles fallas que el sistema puede detectar y sobre las que puede actuar son:

- **Problema en el vínculo de comunicación del ISP 2:** Al realizar el chequeo de disponibilidad desde el ISP 1, el módulo de monitoreo detectará que el servicio publicado con la IP del ISP 2 no es accesible, con lo cual disparará la convergencia actualizando la zona del servidor DNS primario con la IP del ISP 1 para los servicios **www** y **mail**. Como el vínculo con ISP 2 no se encuentra disponible, la actualización de zona no se verá reflejada en el DNS secundario. Al encontrarse caído el vínculo de ISP 2, los requerimientos de resolución de nombres, solo serán atendidos por el servidor DNS 1 que es el único accesible en la red. El mimo responderá con la dirección IP de failover configurada para el servicio.
- **Problema en un nodo intermedio de la red:** Al realizar el chequeo de disponibilidad desde el ISP 1, el módulo de monitoreo detectará que el servicio publicado con la IP del ISP 2 no es accesible. Esta situación podría deberse a un inconveniente de comunicación entre ISP 1 e ISP 2. ICINGA detectará que el servicio publicado con la IP del ISP 2 no es accesible, con lo cual disparará la convergencia del mismo modo que ocurrió en el caso anterior, salvo que esta vez, el problema no se encuentra en el vínculo final del ISP 2, sino que el problema reside en un nodo intermedio que causa que la red InterNet se encuentra parcialmente fragmentada. Bajo esta situación, las configuraciones de los servidores DNS no se encontrarán sincronizadas, y los clientes que puedan acceder al DNS publicado a través del ISP 1, resolverán la dirección IP de este proveedor para los servicios **www** y **mail**; mientras que los clientes que puedan acceder al DNS publicado a través del ISP 2 resolverán la dirección IP de este último y accederán al servicio por este enlace.
- **Problema en el vínculo de comunicación del ISP 1:** Al realizar el chequeo de disponibilidad desde el ISP 1, el módulo de monitoreo detectará que el servicio publicado con la IP del ISP 2 no es accesible, con lo cual disparará la convergencia actualizando la zona del servidor DNS primario con la IP del ISP 2 para los servicios **www** y **mail.** Como el vínculo con ISP 2 no se encuentra accesible desde InterNet, la actualización de zona no se verá reflejada en el DNS secundario. Al encontrarse caído el vínculo de ISP 1, los requerimientos de resolución de nombres, solo serán atendidos por el servidor DNS 2 que es el único accesible en la red.

6.7 Conclusiones finales del capítulo

El presente capítulo, permitió probar el funcionamiento completo de la herramienta desarrollada y validar su desempeño. A través de las pruebas realizadas, pudieron detectarse fallas mínimas en la lógica de programación e interfaz, las cuales pudieron ser corregidas sin necesidad de cambiar el enfoque del desarrollo. Adicionalmente pudieron plasmarse en topologías emuladas los conceptos estudiados acerca del protocolo BGP. Esta última tarea involucró la instalación del software necesario con el fin de poder simular los casos de estudio, dentro de los cuales se encuentran el emulador GNS3, software Apache, servidor NTP y otros.

Las distintas pruebas fueron realizadas utilizando una topología base e intentando respetar las mismas condiciones y configuraciones. Ante la necesidad de realizar modificaciones a la topología o esquema de pruebas, se intentó que las mismas sean mínimas con el fin de no alterar los resultados. Luego de realizar cada prueba se documentaron los tiempos y métricas utilizados para llegar a la conclusión final de cada una.

Los resultados obtenidos permitieron determinar que el protocolo BGP provee altos tiempos de convergencia. Ante una falla de red en la topología planteada, solo tardó 28 segundos en converger el tráfico desde un enlace hacia otro alternativo. Una vez realizada la convergencia, no resultó necesario realizar ningún cambio ni esperar tiempo adicional. El principal problema de BGP reside las configuraciones que pudieran realizar terceros con el fin de evitar la propagación de cambios ante situaciones de enlaces inestables. La funcionalidad conocida como dampening puede brindar grandes beneficios a terceros en lo que a disminución de procesamiento innecesario respecta, pero puede generar que una red quede demasiado tiempo fuera de servicio ya sea por errores de configuración o enlaces inestables. En las pruebas realizadas se pudo demostrar que una red podría quedar hasta 44 minutos fuera de la topología en caso de ser penalizada.

La herramienta Traffic Manager, provee una performance aceptable en lo que a tiempos de convergencia se refiere. Las pruebas permitieron comprobar que en la media de los casos, la convergencia puede demorar entre 30 y 60 segundos. Dependiendo de la aplicación, y en particular se ha podido analizar en el caso de Internet Explorer, este tiempo podría llegar a demorar hasta 30 minutos. El principal problema que se presenta en la utilización de la herramienta, reside en la necesidad de inyectar tráfico de red para realizar los chequeos de disponibilidad; mientras que en BGP son los routers los que detectan las fallas sin necesidad de agregar una carga extra a la red. En contrapartida, la herramienta permite realizar la convergencia del tráfico destinado a una única dirección IP, mientras que BGP solo permite realizar la convergencia del segmento de red completo.

Las pruebas demostraron que la herramienta funciona de acuerdo a las expectativas y que los parámetros configurados sobre el intervalo entre chequeos, tiempo de vida de los registros de DNS, etc. son acordes a los parámetros utilizados por los desarrolladores de navegadores web. Estos últimos son quienes implementan las caches de resolución por aplicación y generan que el umbral máximo de convergencia medido en tiempo se eleve hasta los 30 minutos. De no existir este concepto, las métricas de convergencia rondarían entre los 30 y 60 segundos.

Capítulo 7

Conclusiones

7.1 Conclusiones finales

La utilización de tecnologías para brindar esquemas de alta disponibilidad en el acceso resulta necesaria para cualquier organización que provea servicios en InterNet.

El protocolo BGP permite conmutar el tráfico de red de un proveedor a otro fácilmente debido a su naturaleza. El principal problema de las soluciones basadas en BGP, radica en los requerimientos necesarios para obtener un bloque de direcciones IP propio, un número de sistema autónomo y la infraestructura de comunicaciones necesaria con el fin de procesar las tablas de ruteo, además del costo económico asociado. Otro factor importante a tener en cuenta en el uso de BGP es que la conmutación del tráfico no puede realizarse parcialmente en organizaciones pequeñas debido a los impedimentos planteados de no publicar redes con una máscara inferior a los 24 bits. De esto resulta que la conmutación en BGP debe realizarse para 254 direcciones IP en una misma operación como mínimo. Por último, es importante tener en cuenta que reiteradas actualizaciones de BGP causadas por un enlace de red inestable, pueden causar una penalización para las organizaciones debido al procesamiento que ello implica, lo que puede generar que la información no sea propagada por los vecinos durante largos periodos de tiempo.

La solución propuesta brinda alta disponibilidad en el acceso basada en el protocolo DNS, lo cual implica menores requerimientos al momento de realizar el registro de un dominio, menores costos de mantenimiento y una infraestructura de red más sencilla. La misma es más accesible a pequeñas organizaciones que no cuentan con la debida infraestructura para afrontar los requerimientos impuestos para la obtención de un bloque de direcciones IP propio. Adicionalmente, permite que la conmutación del tráfico pueda realizarse parcialmente para un único servicio, basándose en tantas métricas como chequeos puedan realizarse (disponibilidad, ancho de banda, horarios, etc.). El tiempo de propagación de cambios puede variar dependiendo del tipo de cliente que realice la resolución y las configuraciones locales definidas. De acuerdo a las pruebas realizadas, se ha demostrado que los tiempos de convergencia de la solución de direccionar tráfico utilizando el protocolo DNS son similares a los de BGP en el mejor de los casos y superadores ante la situación de que una ruta sea penalizada. Es importante publicar registros de DNS con tiempos de expiración cortos, lo que generará una mayor cantidad de consultas, pero permitirá actualizar las caches intermedias con mayor frecuencia. Otro factor importante a tener en cuenta es que los cambios de DNS no sufren de penalizaciones en caso de ser reiterados como en el caso de la actualización continua en BGP.

Por último es importante destacar que si bien la tecnología propuesta dota a las pequeñas organizaciones de una herramienta importante al momento de evaluar un esquema de alta disponibilidad en el acceso, la misma puede resultar muy útil en combinación con el protocolo BGP para las grandes organizaciones. Los grandes ISP pueden utilizar el protocolo BGP para converger tráfico de red de un vínculo

a otro ante una caída global, mientras que ante una situación puntual y de menor envergadura, puede resultar más útil converger utilizando el protocolo de resolución de nombres.

7.2 Trabajo a futuro

El conjunto de funciones implementadas en la herramienta abarca a la mayoría de las necesidades. No obstante existen diversos escenarios en infraestructuras de red grandes que no contempla la funcionalidad implementada en la herramienta. Por lo tanto, se propone implementar las siguientes funciones como trabajo a futuro:

- Capacidad de almacenar más de una dirección IP para la resolución de DNS, de modo de permitir la implementación de esquemas Round Robin de DNS realizando chequeos a las distintas direcciones IP que forman parte del pool y eliminando aquellas que presenten problemas de disponibilidad
- Implementar un proxy de DNS que intercepte las consulta realizadas por parte de lo cliente y permita dar respuestas inteligente basando las decisiones en diversos factores además de la disponibilidad. Estos factores incluyen el estado de los vínculos de re en cuanto a saturación se refiere, así como también la cantidad de saltos que deben atravesar las repuestas para llegar a los clientes.
- Implementar esquemas programados de downtime, de modo tal de programar caídas de vínculos o direccionar manualmente el tráfico hacia el vínculo de contingencia ante necesidades puntuales
- Implementar esquemas de notificaciones a través de distintos métodos como por ejemplo Mail, MS o Whatsapp

ANEXO A

Arquitectura REST

La Transferencia de Estado Representacional (REpresentation State Transfer) describe un estilo arquitectónico de sistemas en red como, por ejemplo, aplicaciones Web. El término fue utilizado por primera vez en el año 2000 durante una disertación doctoral por Roy Fielding, uno de los principales autores de la especificación HTTP. REST está comprendida por una serie de limitaciones y principios arquitectónicos. Si una aplicación o diseño cumple con esas limitaciones y principios, se considera RESTful. Uno de los principios REST de mayor importancia para las aplicaciones Web es que la interacción entre el cliente y el servidor no tiene estado entre solicitudes. Cada solicitud del cliente al servidor debe contener toda la información necesaria para comprender la solicitud. En el extremo del servidor, el estado y la funcionalidad de la aplicación se dividen en recursos. Un recurso es un elemento de interés, una identidad conceptual que se expone a los clientes. Algunos ejemplos de recursos son: objetos de aplicaciones, registros de bases de datos, algoritmos, etc. Cada recurso es de acceso único a través de una URI (Universal Resource Identifier). Todos los recursos comparten una interfaz uniforme para la transferencia de estados entre el cliente y el servidor. Se usan métodos estándar HTTP como *GET*, *PUT*, *POST* y *DELETE*. El motor del estado de la aplicación es Hypermedia y las representaciones de recursos se interconectan mediante hipervínculos. Otro principio REST importante es el de sistema por capas, el cual implica que un componente no puede ver más allá de la capa inmediata con la cual interactúa. Al restringir el conocimiento del sistema a una sola capa, se impone un límite en la complejidad del sistema en general, promoviendo así la independencia de las componentes.

Si bien el término REST se refería originalmente a un conjunto de principios de arquitectura, en la actualidad se usa en el sentido más amplio para describir cualquier interfaz entre sistemas que utilice directamente HTTP para obtener datos o indicar la ejecución de operaciones sobre los datos, en cualquier formato (XML, JSON, etc) sin las abstracciones adicionales de los protocolos basados en patrones de intercambio de mensajes, como por ejemplo SOAP.

El mejor modo de comprender la definición de la arquitectura REST, es compararlo con servicios web estándar que siguen el paradigma RPC. Hasta hace poco tiempo, los servicios web basados en SOAP construidos con arquitectura de estilo RPC representaban el enfoque más popular para la implementación de una Arquitectura orientada a servicios (SOA). En éste los clientes de un servicio web de estilo RPC envían paquetes con datos como información de métodos y argumentos al servidor utilizando el protocolo HTTP. El servidor abre el paquete y ejecuta los métodos nombrados con los argumentos transmitidos. Los resultados del método se encapsulan en un nuevo paquete y se devuelven al cliente como respuesta. Todo objeto tiene sus propios métodos y el servicio web de estilo RPC expone solamente una URI, la cual representa el único punto final. El enfoque RESTful de los servicios web surge como una alternativa popular por su naturaleza liviana y a la capacidad de transmitir datos directamente sobre HTTP. En un servicio web de estilo REST, cada recurso tiene una dirección. Los recursos en sí son los objetivos de las llamadas de los métodos y todos los recursos comparten una misma lista de métodos. Los métodos son estándar del protocolo HTTP.

Lo objetivos que busca la arquitectura REST son:

- **Escalabilidad de interacción con componentes:** La Web ha crecido exponencialmente sin degradar su rendimiento. Una prueba de ello es la variedad de clientes que pueden acceder a través de la Web: estaciones de trabajo, sistemas industriales, dispositivos móviles,…
- **Generalidad de interfaces:** Gracias al protocolo HTTP cualquier cliente puede interactuar con cualquier servidor HTTP sin ninguna configuración especial. Esto no es del todo cierto para otras alternativas, como SOAP para los Servicios Web.
- **Puesta en funcionamiento independiente:** Este hecho es una realidad que debe tratarse cuando se trabaja en Internet. Los clientes y servidores pueden ser puestas en funcionamiento durante años. Por tanto, los servidores antiguos deben ser capaces de entenderse con clientes actuales y viceversa. Diseñar un protocolo que permita este tipo de características resulta muy complicado. HTTP permite la extensibilidad mediante el uso de las cabeceras de las URIs, a través de la habilidad para crear nuevos métodos y tipos de contenido.
- **Compatibilidad con componentes intermedios:** Los más populares intermediaros en comunicaciones HTTP son los proxys. Algunos de ellos, las caches, se utilizan para mejorar el rendimiento. Otros permiten reforzar las políticas de seguridad como por ejemplo los firewalls. La compatibilidad con componentes intermedios permite reducir la latencia de interacción, reforzar la seguridad y encapsular otros sistemas.

REST logra satisfacer estos objetivos aplicando cuatro restricciones:

- Identificación de recursos y manipulación de ellos a través de representaciones. Esto se consigue mediante el uso de URIs. HTTP es un protocolo centrado en URIs. Los recursos son los objetos lógicos a los que se le envían mensajes. Los recursos no pueden ser directamente accedidos o modificados. Más bien se trabaja con representaciones de ellos. Cuando se utiliza un método PUT para enviar información, se coge como una representación de lo que nos gustaría que el estado del recurso fuera. Internamente el estado del recurso puede ser cualquier cosa desde una base de datos relacional a un fichero de texto.
- Mensajes autodescriptivos. REST dicta que los mensajes HTTP deberían ser tan descriptivos como sea posible. Esto hace posible que los intermediarios interpreten los mensajes y ejecuten servicios en nombre del usuario. Uno de los modos que HTTP logra esto es por medio del uso de varios métodos estándares, muchos encabezamientos y un mecanismo de direccionamiento. Por ejemplo, las cachés Web saben que por defecto el comando GET es cacheable (ya que es side-effect-free) en cambio POST no lo es. Además saben cómo consultar las cabeceras para controlar la caducidad de la información. HTTP es un protocolo sin estado y cuando se utiliza adecuadamente, es posible es posible interpretar cada mensaje sin ningún conocimiento de los mensajes precedentes. Por ejemplo, en vez de loguearse del modo que lo hace el protocolo FTP, HTTP envía esta información en cada mensaje
- Hipermedia como un mecanismo del estado de la aplicación. El estado actual de una aplicación Web debería ser capturada en uno o más documentos de hipertexto, residiendo tanto en el cliente como en el servidor. El servidor conoce sobre el estado de sus recursos, aunque no intenta seguirle la pista a las sesiones individuales de los clientes. Esta es la misión del navegador, él sabe cómo navegar de recurso a recurso, recogiendo información que el necesita o cambiar el estado que el necesita cambiar.

Bibliografía

[1] Deering, S.; Hinden, R. (1998) *Internet Protocol, Version 6 (IPv6).* RFC 2460: http://www.ietf.org/rfc/rfc2460.txt

[2] Darpa Internet Program. (1981) *Internet Protocol, Protocol Specification*. RFC 1981: http://www.ietf.org/rfc/rfc791.txt

[3] Malkin, G. (1994) *RIP Version 2*. RFC 1723: http://www.ietf.org/rfc/rfc723.txt

[4] Malkin, G. (1998) *RIP Version 2*. RFC 2453: http://www.ietf.org/rfc/rfc2453.txt

[5] Moy, J. (1998) *OSPF Version 2*. RFC 2328: http://www.ietf.org/rfc/rfc2328.txt

[6] Hinden, R.; Sheltzer, A. (1982) *The Darpa Internet Gateway*. RFC 823: http://www.ietf.org/rfc/rfc823.txt

[7] Rosen, E. (1982) *Exterior Gateway Protocol (EGP).* RFC 827: http://www.ietf.org/rfc/rfc827.txt

[8] Mills, D. (1984) *Exterior Gateway Protocol Formal Specification.* RFC 904: http://www.ietf.org/rfc/rfc904.txt

[9] Lougheed, J.; Rekhter, Y. (1989) *A border Gateway Protocol (BGP)*. RFC 1105: http://www.ietf.org/rfc/rfc1105.txt

[10] Li, T.; Rekheter, Y (1995) *A Border Gateway Protocol 4 (BGP-4). Version 4.* RFC 1771: http://www.ietf.org/rfc/rfc1771.txt

[11] Rekhter, Y.; Li, T.; Hares, S. (2006) *A Border Gateway Protocol 4 (BGP-4)*. RFC 4271: http://www.ietf.org/rfc/rfc4271.txt

[12] Van Beijnum, I. (2002) *BGP.* O'REILLY

[13] Sitio web de BGP: http://www.bgp4.as/. Consultado Marzo de 2013

[14] Zhang, R.; Bartell, M. (2003) *BGP Design and Implementation*. Cisco Press

[15] Teare, D. (2010) *Implementing Cisco IP* Routing (ROUTE). Cisco Press

[16] Huitema, C. (1995) *Routing in the Internet*. Prentice Hall

[17] Hawkinson, T.; Bates, T (1996) *Guidelines for creation, selection, and registration of an Autonomous System (AS).* RFC 1930http://www.ietf.org/rfc/rfc1930.txt

[18] Villamizar, C.; Chandra, R.; Govindan, R. (1998) *BGP Route Flap Damping*. RFC 2439: http://www.ietf.org/rfc/rfc2439.txt

[19] BGP: Frequently Asked Questions: http://www.cisco.com/en/US/tech/tk365/technologies_q_and_a_item09186a00800949e8.shtml y http://www.cisco.com/warp/public/459/bgpfaq_5816.shtml. Consultado Septiembre de 2014

[20]Fall Kevin R.; Stevens W. Richard (2012) *TCP/IP Illustrated, Volume 1. The Protocols, 2e.*

[21] Mockapetris, P. (1987) *Domain Names – Concepts and Facilities.* RFC 1034: http://tools.ietf.org/html/rfc1034.txt

[22] Mockapetris, P. (1987) *Domain Names – Implementation and Specification*. RFC 1035: http://tools.ietf.org/html/rfc1035.txt

[23] Liu, C.; Albits, P. (2006) *DNS and BIND 5e.* O'REILLY

[24]*Internet Corporation for Assigned Names and Numbres, Internationalized Domain Names (IDNs).* https://www.icann.org/resources/pages/idn-2012-02-25-en. Consultado Mayo de 2014

[25] Colouris G.; Dollimore, J.; Kindberg, T. (2000) *Distributed Systems Concepts and Design 3e*. Addison–Wesley

[26] Sitio web de BIND: http://www.isc.org/software/bind. Consultado Marzo de 2013

[27] Jeftovic, M. E. (2015) *Managing Mission-Critical Domains and DNS.* O'REILLY

[28] Tanenbaum, A.; Van Steen, M. (2007) *Distributed Systems: Principles and Paradigms, 2e*. Prentice Hall

[29] Thomson, S.; Rekhter, Y.; Bound, J. (1997) *Dynamic Updates in the Domain Name System (DNS UPDATE)*. RFC 2136: http://tools.ietf.org/html/rfc2136.txt

[30] Eastlake, M. (1999) *Domain Name System Security Extensions.* RFC 2535: http://tools.ietf.org/html/rfc2535

[31] Arends, R.; Austein, R.; Larson, M.; Massey, D.; Rose, S. (2005) *DNS Security Introduction and Requirements.* RFC 4033: http://tools.ietf.org/html/rfc4033.txt

[32] Wellington. B. (2000) *Secure Domain Name System (DNS) Dynamic Update.* RFC 3007: http://tools.ietf.org/html/rfc3007

[33] Status map of DNSSEC deployment in ccTLD and gTLD. http://www.ohmo.to/dnssec/maps/. Consultado Mayo de 2014

[34] Sitio web de Icinga: http://www.icinga.org/. Consultado Octubre de 2013

[35] Sitio web de Nagios: http://www.nagios.org/. Consultado Marzo de 2013

[36] Mehta, V. (2013) *Icinga Network Monitoring.* Packt

[37] Ryder T. (2013) *Nagios Core Administration Cookbook.* Packt Publishing

[38]Neculae, A. (2014) *Icinga Monitoring System Interface*. http://cds.cern.ch/record/1751464. Consultado Marzo de 2015.

[39] Sitio web de PHP: http://www.php.net/. Consultado Marzo de 2013

[40]Navarro Marset, R. (2006) *REST Vs Web Services*. http://users.dsic.upv.es/~rnavarro/NewWeb/docs/RestVsWebServices.pdf. Consultado Mayo de 2015