Estudio y Simulación de redes Border Gateway Protocol

Proyecto Fin de Carrera

Francisco Huertas Ferrer

Proyecto de Sistemas Informáticos
Facultad de Informática
Universidad Complutense de Madrid

Septiembre 2010

Documento maquetado con TEXIS v.1.0+.

Estudio y Simulación de redes Border Gateway Protocol

Memoria Fin de Proyecto

Francisco Huertas Ferrer

Dirigida por

Dr. Juan Carlos Fabero Jiménez
Pedro A. Aranda Gutiérrez

Proyecto de Sistemas Informáticos
Facultad de Informática
Universidad Complutense de Madrid

Septiembre 2010

Autorizo a la Universidad Complutense a difundir y utilizar con fines académicos, no comerciales y mencionando expresamente a sus autores, tanto la propia memoria, como el código, la documentación y/o el prototipo desarrollado

Fdo: Francisco Huertas Ferrer

Agradecimientos

En primer lugar me gustaría dar gracias a mi jefe de TID, Francisco Javier, por darme la oportunidad de formar parte de mi grupo de trabajo y, con ello, llevar a cabo este proyecto. Gracias a mi tutor, Pedro, por su apoyo y por todos los conocimientos que me ha dado y que han hecho posible este proyecto. También me gustaría darle las gracias a Quique por ofrecerme la oportunidad de hacerme valer.

Por parte de la universidad gracias a Juan Carlos por todo la ayuda y apoyo prestados.

Son muchas las personas; amigos, compañeros de universidad y compañeros de TID; con quien he compartido momentos especiales durante los últimos años, algunas se han quedado atrás y otras siguen siendo mis amigos. Gracias a todas ellas por darme su apoyo y aguantarme en los momentos difíciles. En especial gracias a Raúl, Manuel y Cristina porque sé que siempre puedo contar con ellos.

Por último gracias a mis padres y mis hermanos. Siempre han estado cuando les he necesitado, nunca me han defraudado y a ellos les debo todo lo que soy.

Gracias a todos, Paco

Resumen

El Border Gateway Protocol (BGP-4)[1] es un protocolo de encaminamiento entre sistemas autónomos.

La función principal del sistema se comunica con BGP-4 es el intercambio de información de accesibilidad de la red con otros sistemas BGP-4. Esta información de accesibilidad incluye información sobre la lista de Sistemas Autónomos que la información ha atravesado. Esta información es suficiente para construir el grafo que de acceso a la red, podar bucles de encaminamiento y aplicar políticas en el proceso de decisión a nivel de sistema autónomo.

BGP-4 es el protocolo de apoyo del proceso de decisión de encaminamiento de Internet y, como la mayoría de los proveedores de servicios de Internet deben utilizar BGP-4 para establecer rutas entre unos y otros, es uno de los protocolos más importantes de Internet.

Actualmente el estudio de BGP-4 está muy limitado debido a que las herramientas que existen para analizar el protocolo son escasas y tienen problemas de escalabilidad, no están completamente implementadas o no permiten profundizar más en el comportamiento interno del protocolo debido a que son de software propietario.

Los objetivos de este proyecto son:

- Investigar y mejorar las herramientas disponibles para el estudio de BGP-4.
- Estudiar BGP-4 con las herramientas investigadas.

Palabras Claves: BGP, BGP-4, J-Sim, Wedgie, Oscilacion, Proceso de Decisión, Simulador, Simulación

Abstract

The Border Gateway Protocol (BGP-4)[1] is an inter-Autonomous System routing protocol.

The primary function of a BGP-4 speaking system is to exchange network reachability information with other BGP-4 systems. This network reachability information includes information on the list of Autonomous Systems that reachability information traverses. This information is sufficient for constructing a graph of AS connectivity for this reachability, from which routing loops may be pruned and, at the AS level, some policy decisions may be enforced.

BGP-4 is the protocol backing the core routing decisions on the Internet and, since most Internet service providers must use BGP-4 to establish routing between one another, it is one of the most important protocols of the Internet.

Nowadays, the study of BGP-4 is very limited because the tools available to analyze the protocol are limited and have scalability issues, are not fully implemented or do not allow to go in depth into the internal behavior of the protocol because they are proprietary software.

The objectives of this project are:

- Research and improve the tools available for studying BGP-4.
- Study the BGP-4 protocol with the researched tools.

Keywords: BGP, BGP-4, J-Sim, Wedgie, Oscilation, Decision Process, Simulator, Simulation

Índice

Agradecimientos vii

Resumen ix

Abstract xi

1. Introducción 1
- 1.1. Qué es BGP-4 . . . 1
- 1.2. La red BGP-4 . . . 1
- 1.3. Requisitos para el estudio de BGP-4 . . . 2
- 1.4. Alternativas para el estudio de BGP-4 . . . 2

2. JavaSim 5
- 2.1. Arquitectura de JavaSim (J-Sim) . . . 5
  - 2.1.1. Arquitectura de componentes autónomos . . . 7
  - 2.1.2. Tcl y su entorno gráfico Tcl/Tk . . . 8
  - 2.1.3. Arquitectura de las mejoras . . . 8
- 2.2. Desarrollo . . . 9
  - 2.2.1. Rutinas de automatización para la configuración de topologías en Tcl . . . 9
  - 2.2.2. Desarrollo del protocolo BGP-4 . . . 9
  - 2.2.3. Abstracción de Tcl de la configuración de topologías y gestión de eventos . . . 11
  - 2.2.4. Protocolos y herramientas creadas para el J-Sim . . . 12
  - 2.2.5. Correcciones del simulador . . . 13
  - 2.2.6. Fallos conocidos que no han sido corregidos . . . 13

3. Validación de BGP-4 15
- 3.1. Validación del simulador y BGP . . . 15
  - 3.1.1. Escenario básico BGP-4 . . . 15
  - 3.1.2. Escenario complejo BGP-4 . . . 16
  - 3.1.3. Escenarios Extensos BGP-4 . . . 16

4. Estudio de BGP-4 **19**
4.1. Necesidad de estudio 19
4.2. Proceso de decisión 19
4.2.1. Oscilación 21
4.2.2. Wedgies 28
4.2.3. Proceso de decisión: Conclusión 34

5. Conclusión **37**
5.1. Simulador 37
5.2. Estudio BGP-4 37

A. Apendice: Uso del simulador **39**
A.1. Ejecución del simulador 39
A.2. Ejecución de simulaciones 39
A.3. Estructura de los ficheros XML 39

B. Apendice: Configuraciones **41**
B.1. Escenarios BGP-4 41
B.1.1. Escenario básico 41
B.1.2. Escenario avanzado 45
B.1.3. Escenario extenso 1 48
B.1.4. Escenario extenso 2 50
B.1.5. Escenario que oscila: ejemplo 1 52
B.1.6. Escenario que oscila: ejemplo 2 53
B.1.7. Escenario que oscila: ejemplo 3 55
B.1.8. Escenario wedgie 3/4 57
B.1.9. Escenario full wedgie 59

Bibliografía **61**

Lista de abreviaturas, acrónimos y definiciones **63**

Índice de figuras

1.1. Ejemplo de una topología BGP-4 1
1.2. Tipos de topologías iBGP 2

2.1. Arquitectura a capas de J-Sim 6
2.2. Arquitectura de componentes. Componentes y Puertos 7
2.3. Arquitectura de componentes. Composición de componentes . 8
2.4. Terminal Tcl/Tk 8

4.1. Ejemplo de topología BGP-4 20
4.2. Gráfico de Costes de la topología Figura 4.1 20
4.3. Ejemplo de configuración BGP-4 que oscila 22
4.4. Estado 1: Inicial 22
4.5. Estado 2: Intercambio de información BGP-4 entre los Nodos B y C 23
4.6. Estado 3: La información recibida es contrastada y añadida a la tabla de rutas 24
4.7. Estado 4: Envío de *withdraw* a los vecinos 25
4.8. Estado 5: La información es eliminada y las rutas son borradas de la tabla de rutas. 26
4.9. Estados Finales 26
4.10. Ejemplo de configuración BGP-4 que oscila 27
4.11. Ejemplo de configuración BGP-4 que oscila 27
4.12. Ejemplo de escenario que produce *wedgie 3/4* 29
4.13. Configuración buscada de la topología mostrada en la Figura 4.12 29
4.14. Caída del enlace principal o *Primary Link*, Figura 4.12 30
4.15. Se restablece enlace principal o *Primary Link*, Figura 4.12 . . 31
4.16. Caída del enlace de seguridad o *Backup Link*, Figura 4.12 . . 32
4.17. Ejemplo de escenario que produce *full wedgie* 32
4.18. Estado final estable buscado en la configuración mostrada en la Figura 4.17 33
4.19. Caída del enlace principal o *Primary Link*, Figura 4.17 34

4.20. Caída del enlace de seguridad o *Backup Link*, Figura 4.17 . . 35
4.21. Caída del enlace secundario de seguridad o *Secondary Backup Link*, Figura 4.17 36

B.1. Ejemplo 1: Topología de un escenario básico. 41
B.2. Ejemplo 2: Topología de un escenario avanzado. 45
B.3. Ejemplo 3: Topología de un escenario de Grandes dimensiones 48
B.4. Ejemplo 4: Topología de un escenario de Grandes dimensiones 50
B.5. Ejemplo 5: Topología de un escenario básico que oscila 52
B.6. Ejemplo 6: Topología de un escenario básico que oscila 53
B.7. Ejemplo 7: Topología de un escenario avanzado que oscila . . 55
B.8. Ejemplo 8: Topología de un escenario que sufre *wedgie 3/4* . . 57
B.9. Ejemplo 9: Topología de un escenario que sufre *full wedgie* . . 59

Capítulo 1

Introducción

1.1. Qué es BGP-4

Border Gateway Protocol es un protocolo de encaminamiento externo entre Sistemas Autónomos (AS). Se utiliza para el intercambio de información de encaminamiento entre sistemas BGP-4. Ésta incluye información de accesibilidad a las redes de los diferentes Sistemas Autónomos, así como la forma de llegar hasta ellos.

Dicha información es suficiente para construir un grafo de conectividad en cuanto a esta accesibilidad, pudiendo podar bucles de encaminamiento y aplicar ciertas políticas de decisión a nivel de AS.

BGP-4 es el protocolo de encaminamiento más extendido entre los principales proveedores de Internet para intercambiar sus redes y así tener acceso a los contenidos que ofrecen y sus clientes.

1.2. La red BGP-4

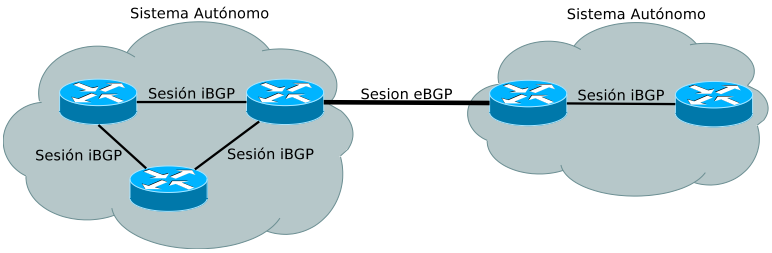
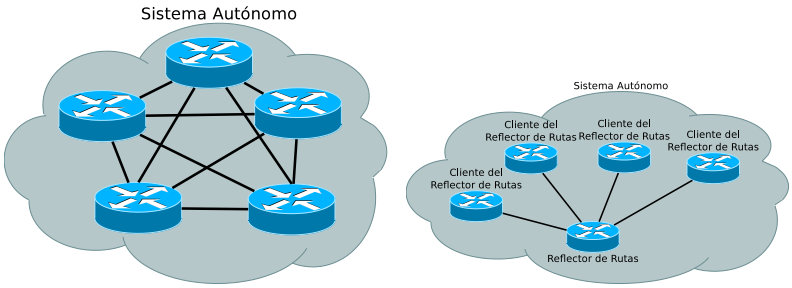


Figura 1.1: Ejemplo de una topología BGP-4

La red o topología BGP-4 está compuesta por nodos o encaminadores que se agrupan en AS como muestra la Figura 1.1. Los nodos se conectan entre sí con sesiones BGP-4.

Cuando la sesiones se mantienen entre nodos de distintos AS se denomina BGP externo (eBGP) y los nodos reciben el nombre de nodos frontera. Cuando es entre nodos de un mismo AS se denomina BGP interno (iBGP). Las sesiones iBGP pueden existir entre todos los componentes del AS según muestra la Figura 1.2a o, si existe un nodo con características de Reflector de Rutas (RR), únicamente entre éste y el resto de los nodos del AS según muestra la Figura 1.2b.



(a) Topología iBGP sin Reflector de Rutas

(b) Topología iBGP con Reflector de Rutas

Figura 1.2: Tipos de topologías iBGP

1.3. Requisitos para el estudio de BGP-4

Para poder estudiar BGP-4 en profundidad es necesario entender y conocer el protocolo, disponer del código fuente.

BGP-4 es también, un protocolo usado en redes de gran tamaño como Internet. Poder analizar redes de estas características es otro requisitos que necesitamos para estudiar BGP-4.

Una aspecto interesante para poder realizar el estudio, es que la forma en que se manejan los datos de configuración y resultados de simulación sean sencillos de entender y crear.

1.4. Alternativas para el estudio de BGP-4

Para realizar un estudio de BGP-4 existen varias alternativas:

- La base de datos de Réseaux IP Européens (RIPE)[2] contiene detalles de registro de direcciones IP y números AS. Muestra la organización que mantienen los recursos, cuando fueron hechas las asignaciones, datos de las redes e información de encaminamiento. La información de encaminamiento puede ser utilizada para el estudio de intercambio de rutas BGP-4.

- Los sistemas de emulación copian todo el funcionamiento de un nodo concreto. Se suele necesitar el *firmware* del nodo que se desea emular. Una emulación suele ser en tiempo real y son fieles a lo que el nodo emulado haría en la realidad, lo que proporciona datos muy fiables, similares a los reales. Sin embargo tienen grandes problemas de escalabilidad, debido a la gran cantidad de recursos que consumen, y generalmente, al no tener acceso al código original del protocolo, es difícil detectar problemas internos del protocolo emulado. Tampoco pueden utilizarse para la implementación de nuevos protocolos basados en los antiguos debido a la rigidez del sistema.

- Los sistemas de simulación implementan las especificaciones teóricas de los elementos simulados. El realismo de los resultados obtenidos es menor y dependen de una correcta implementación del protocolo para obtener resultado fiables. No se busca detectar errores de implementación de protocolos en sistemas reales, sino comportamientos teóricos de elementos de red en topologías concretas. Otro de los usos de las herramientas de simulación es el estudio de nuevos protocolos y el posible impacto que pueda tener éste en la red. Los recursos necesarios para simular un entorno son mucho menores que los necesarios para emular el mismo entorno, por lo que pueden simular redes de mayores. Como consecuencia, los simuladores tienen mayor escalabilidad. Otra características interesante de los simuladores es la gran cantidad de recursos disponibles en la red.

Entre las alternativas el simulador es lo que mejor se amolda a los requisitos necesarios. El simulador que utilizaremos será el J-Sim. Es un programa de código abierto y dispone de una implementación incompleta de BGP-4.

Capítulo 2

JavaSim

JavaSim (J-Sim)[3] es un simulador basado en arquitectura de componentes. Se ha construido en base al modelo de componentes de programación autónoma. Al igual que el COM / COM +, JavaBeansTM, o CORBA, la entidad básica de J-Sim es el componente, pero a diferencia de otros paquetes de software basados en componentes y normas, los componentes de J-Sim son autónomos y representan circuitos integrados de software, Sección 2.1.1.

J-Sim está implementado en Java y proporciona una interfaz de secuencia de comandos que permite integrarse con el lenguaje de script Tool Command Language (Tcl), Sección 2.1.2

Para modelar las simulaciones de redes, J-Sim define un paquete de conmutación generalizada del modelo de redes, como una capa superior de la Arquitectura de componentes autónomos (ACA). Este modelo define la estructura jerárquica de los nodos y componentes genéricos de red, según la estructura en capas de J-Sim.

2.1. Arquitectura de J-Sim

La arquitectura de J-Sim sigue el modelo de capas de la Figura 2.1.

1. La capa ACA contiene los elementos Componente[1] y Puerto[2] de la Arquitectura de componentes autónomos

2. La capa NET contiene primitivas de simulación de red y herramientas tales como Paquete[3], Módulo[4], Dirección[5], modelos de tráfico como

[1] `drcl.comp.Component`
[2] `drcl.comp.Port`
[3] `drcl.net.Packet`
[4] `drcl.net.Module`
[5] `drcl.net.Address`

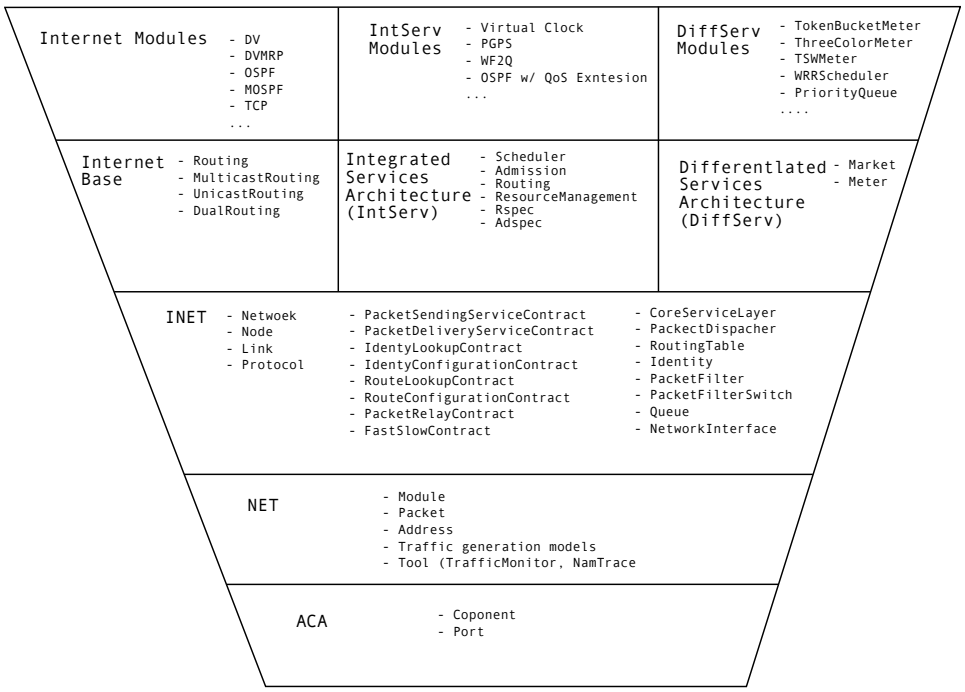


Figura 2.1: Arquitectura a capas de J-Sim

CBR[6] y monitores de controlar el tráfico[7] y el generador de trazas NAM[8].

3. La capa de INET contiene los componentes INET. En concreto, se definen

   - Los componentes básicos; red[9], nodo[10] y enlace[11]; para la construcción de redes jerárquicas en el simulador.
   - El componente protocolo[12] que sirve como clase base para implementar módulos de protocolo.
   - La capa de servicios del núcleo[13] y los componentes que la constituyen dicha capa.

4. Las arquitecturas especificas derivadas de la capa INET.

[6] `drcl.net.traffic`
[7] `drcl.net.tool.TrafficMonitor`
[8] `drcl.net.tool.NamTrace`
[9] `drcl.inet.Net`
[10] `drcl.inet.Node`
[11] `drcl.inet.Link`
[12] `drcl.inet.Protocol`
[13] `drcl.inet.CoreServideLayer`

5. Protocolos de módulos y/o algoritmos específicos de una arquitectura de red se aplican en la parte superior de esa capa de arquitectura.

2.1.1. Arquitectura de componentes autónomos

La Arquitectura de componentes autónomos (ACA) imita la arquitectura de componentes diseño de circuitos integrados. El comportamiento de estos componentes pueden ser implementado y probado forma individual y, posteriormente desplegados en un sistema.

Los elementos del paquete `drcl.comp`, se encargan de la gestión de la arquitectura de componentes, la comunicación, control de tiempo de simulación, etc.

Componente y Puerto

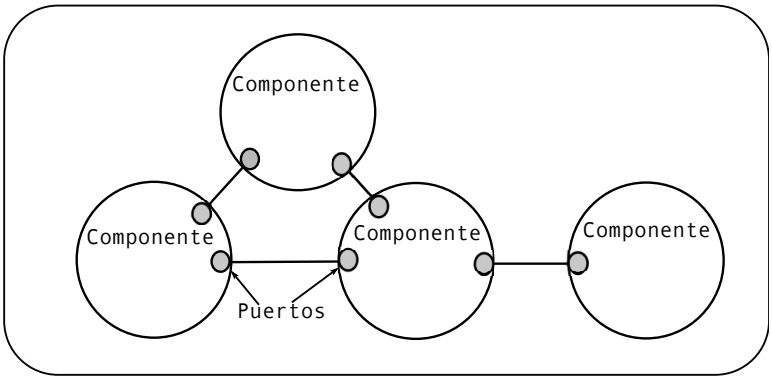


Figura 2.2: Arquitectura de componentes. Componentes y Puertos

En la ACA de J-Sim la entidad básica es el componente[14]. Estos componentes poseen uno o más puertos[15]. Los puertos pueden ser conectados entre si para que puedan comunicarse de tal manera que cada vez que llegan datos a través de uno de estos puertos, éstos son tratados en un nuevo contexto de ejecución o hilo y pueden generar a su vez salidas en otros puertos del componente. La Figura 2.2 muestra un ejemplo de este tipo de arquitectura.

Los componentes soportan la composición de tal manera que un componente puede contener varios componentes como muestra la Figura 2.3.

Por otro lado, se definen como componentes hermanos aquellos que pertenecen al mismo componente denominado componente padre y estos son componentes hijos de él. La comunicación entre componentes se da entre hermanos o entre padre e hijo. Un componente y todos sus hijos son tratados como una unidad por el resto de componentes del mismo sistema.

[14] `drcl.comp.Component`
[15] `drcl.comp.Port`

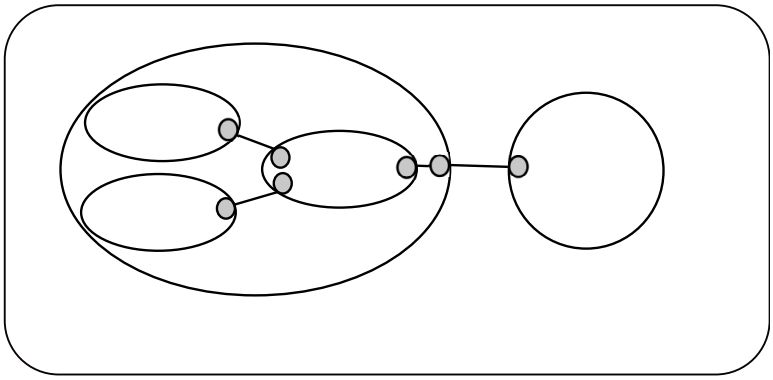


Figura 2.3: Arquitectura de componentes. Composición de componentes

2.1.2. Tcl y su entorno gráfico Tcl/Tk

J-Sim está implementado en Java y no necesita ningún lenguaje de script para ejecutarse . Sin embargo puede utiliza Tool Command Language (Tcl), para unificar los componentes del sistema. El entorno gráfico consiste en un terminal donde pueden ejecutar comandos y cargar Scripts de este lenguaje, y ver los resultados de los mismos como muestra la Figura 2.4.

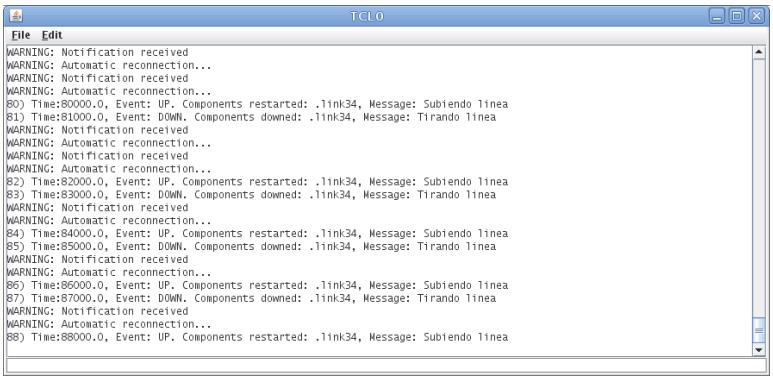


Figura 2.4: Terminal Tcl/Tk

Además existe un sistema que hace referencia a la jerarquía de componentes y puertos. Ésta es muy similar al sistema de archivos de UNIX y se usa la misma notación representando los componentes y sus puertos como rutas, de la misma manera que se representan los archivos y directorios en UNIX.

2.1.3. Arquitectura de las mejoras

La arquitectura de los cambios y mejoras ha seguido el mismo modelo que la arquitectura del simulador. Estos cambios por otro lado se han hecho

en paquetes diferentes para conservar integro el programa original, por si sufre actualizaciones. Los paquetes son los siguientes:

- `tid.inet.InetUtil` Rutinas de automatización para la configuración de topologías en Tcl. Sección 2.2.1
- `Infonet.javasim.bgp4` Desarrollo del protocolo BGP-4 para el simulador. Sección 2.2.2
- `tid.inet` Sistema de abstracción de Tcl de la configuración de topologías. Sección 2.2.3
- `tid.inet.protocols` Protocolos y herramientas creadas para el simulador. Sección 2.2.4
- `tid.events` Sistema de abstracción de Tcl de los eventos de red. Sección 2.2.3

2.2. Desarrollo

2.2.1. Rutinas de automatización para la configuración de topologías en Tcl

La clase InetUtil original[16] contiene métodos para la automatización que pueden ser utilizados por Tcl. Se ha añadido mas funcionalidad en la clase homónima que se encuentra en el paquete `tid.inet`. Estas funcionalidades se usan principalmente en la Sección 2.2.3.

Estas rutinas representan operaciones básicas de red, gestionan interfaces virtuales, el comportamiento en caídas de elementos del sistema etc.

2.2.2. Desarrollo del protocolo BGP-4

Partiendo del paquete original desarrollado por el grupo Infonet[4] de la Universidad de Louvain-la-Neuve para dar soporte completo al protocolo BGP-4. Este a su vez se basaba en la implementación hecha por SSFNET[5]. Esta paquete es una buena base para el desarrollo pero carece de gran parte de las funcionalidades del protocolo y tiene fallos importantes en el código.

Correcciones al paquete

La lista completa de correciones del protocolo están en el registro de cambios de la implementación adjuntada. Las mejoras añadidas a la implementación original del equipo Infonet son de dos tipos, errores de implementación

[16] `drcl.inet.InetUtil`

en la lógica interna de BGP-4 y funcionalidades que no están correctamente implementadas.

Los problemas de funcionalidad solucionados son:

- Soporte de la reflexión de rutas.
- Soporte para el uso de atributos en las sesiones BGP-4.
- Soporte para el uso de de políticas en las sesiones BGP-4.
- Soporte de restablecimiento de conexión entre sesiones que han perdido la comunicación.

Los principales problemas de implementación solucionados son:

- Un correcto soporte de recuperación de la conexión entre dos nodos en caso de que esta se pierda en algún momento.

2.2.2.1. Cambios a nivel de implementación

Las modificaciones a nivel de implementación que se han hecho en el paquete original se han hecho para adaptar el protocolo a los requisitos de la Sección 2.2.3. Estos cambios afectan a:

- La forma en que el protocolo recibe la configuración de las diferentes sesiones BGP-4. No solo cambia la forma en que recibe la configuración sino también la cantidad de parámetros que pueden ser configurados.
- Los procedimientos que gestionan las de caídas y restablecimiento de enlaces en una simulación.
- La forma en que se crean los ficheros de trazas.

2.2.2.2. Funcionalidades añadidas a BGP-4

De manera experimental se han añadido modificaciones al protocolo no contempladas en la especificación inicial de BGP-4. Las nuevas funcionalidades implementadas son opcionales y modifican ligeramente el comportamiento del protocolo:

- Soporte de *Multipath* a nivel del protocolo.
- Se puede modificar el proceso de decisión de BGP-4 de tal manera que se puede cambiar el momento en que dicho proceso decide que dos rutas son equivalentes.

2.2.3. Abstracción de Tcl de la configuración de topologías y gestión de eventos

Una de las carencias del simulador consiste en la complejidad de crear escenarios. Las configuraciones se crean manualmente para cada nodo y protocolo utilizando sus primitivas. Realizar cambios sobre una topología suele ser bastante costoso y simular topologías de gran cantidad de tamaños resulta prácticamente imposible.

El lenguaje de script Tcl facilita la creación de topologías. Utilizando sus rutinas de automatización se consigue un mayor grado de abstracción. Sin embargo es necesario conocer el lenguaje Tcl y aun así sigue siendo muy complejo y costoso realizar las configuraciones.

Para conseguir un mayor grado de abstracción en la configuración de simulaciones se ha adaptado el simulador para que use el lenguaje XML. Extensible Markup Language (XML) nos permite almacenar la información de la simulación de forma clara, sencilla y estructurada. La lectura de la información se realiza iniciando un Script Tcl, y no es necesario tener conocimientos de Tcl ni de la estructura interna del simulador.

Estos ficheros de simulación nos permiten automatizar el proceso de creación de las configuraciones. Reduciendo la complejidad para crear nuevas simulaciones con un gran número de nodos y configuraciones más complejas.

Los ficheros tienen estructuras definidas y claras que ayudan a su lectura y facilitan su reutilización en futuras simulaciones.

Los ficheros de simulación son dos. El primero de los ficheros, la topología, contiene la información correspondiente a la configuración de la red. El segundo fichero, los eventos, almacena información acerca de lo que ocurre durante la simulación.

El fichero de topología contiene:

- La configuración de la red.
- La configuración de los nodos de una red, sus conexiones e interfaces.
- La configuración de los protocolos soportados. Estos implementan los requisitos descritos en la Sección 2.2.3.1

El fichero de eventos contiene:

- El tiempo de simulación.
- La configuración de los eventos y el momento en que ocurren.

El formato de los ficheros de configuración pueden ser consultado en la Sección A.3, en el se incluyen como tiene que ser escritos incluyendo todos los protocolos que actualmente son soportados.

2.2.3.1. Nuevos requisitos para los protocolos

Es necesario que el protocolo cumpla una serie de requisitos, estos son imprescindibles para que el sistema pueda leer la configuración tanto de los parámetros del protocolo como del registro de eventos, y sea capaz de iniciar la simulación del protocolo.

- Implementar la interfaz del Protocolo[17]. Ésta contiene las primitivas de configuración del protocolo y las necesarias para iniciar la simulación. La documentación referente a dichas primitivas se puede consultar en el JavaDoc de la interfaz.
- Modificar las factorías que existen en el paquete de factorías[18]. Estas factorías realizan las llamadas a las primitivas de los protocolos que los configuran. Estas factorías son:
  - `doConfigFactory` que realiza las llamadas a las primitivas de configuración del protocolo.
  - `GeneralParametersFactory` que realiza las llamadas a las primitivas de configuración de los parámetros generales del protocolo.
  - `mapStringFactory` que realiza las llamadas para crear una cadena necesaria para configurar el entorno de simulación. El formato de dicha cadena viene explicado en el JavaDoc de la clase `drcl.inet.InetUtil`.
- En el fichero de configuración XML hay dos secciones dedicadas a los protocolos. Ambas tienen que tener un atributo `type` con el valor del id del protocolo para poder identificarlo en las factorías. Estas son:
  - Colgando directamente de la raíz del fichero de configuración, con la etiqueta `protocol`, está sección opcional esta orientada a la configuración general del protocolo.
  - En la etiqueta `router` se encuentra `session` que contiene la configuración de la sesión de ese protocolo para ese nodo. Un protocolo con una.

2.2.4. Protocolos y herramientas creadas para el J-Sim

2.2.4.1. GP-BGP

El simulador a sido utilizado como soporte en el proyecto europeo 4Ward[6] y para contrastar los resultados de un nuevo protocolo basado en BGP-4 llamado GP-BGP. La especificación del protocolo se encuentra en el entregable

[17] `tid.inet.protocols.Protocol`
[18] `tid.inet.protocols`

5.2 *Mechanisms for Generic Paths*. Los resultados obtenidos van a exponerse en el congreso Monami-2010[7].

2.2.4.2. TrafficInspectionTool

Este protocolo se ha creado para contener las herramientas inspección de tráfico configurables desde los archivos XML. La herramienta que soporta es la traza de rutas

2.2.5. Correcciones del simulador

A lo largo del proyecto se han solucionado fallos y se han introducido pequeñas mejoras. Las más destacables son:

- Las desconexiones de algunos componentes para simular caídas del sistema son tratadas por medio de excepciones. Éstas no siempre lo hacían de forma correcta y causaban que los hilos de ejecución de los componentes asociado a estas caídas terminara su ejecución.
- Cuando un mensajes era recibido en una interfaz virtual era tratada desde la Capa de Servicios de Red (CSL).
- Existían discordancias en las constantes usadas por el simulador para algunas de sus herramientas y servicios.
- Se han creado funcionalidades en la clase `drcl.inet.Node` para la gestión de direcciones del nodo y el estado de las conexiones con nodos adyacentes.

El historial de cambios esta disponible en el registro de cambios adjuntos al proyecto.

2.2.6. Fallos conocidos que no han sido corregidos

Existen problemas cuando dos nodos tienen interfaces de red con la misma de dirección aunque éstos no sean visibles entre ellos.

El simulador no tiene soporte para IPv6. Se estudió la posibilidad de darle soporte pero se descartó por lo costoso que resultaba.

Capítulo 3

Validación de BGP-4

Para comprobar que el simulador esta correctamente implementado se ha hecho un conjunto de simulaciones que prueben el correcto funcionamiento de BGP-4 en el simulador y las capacidades del mismo.

3.1. Validación del simulador y BGP

3.1.1. Escenario básico BGP-4

Objetivo

Validar el simulador y la implementación de BGP-4. La configuración viene detallada en el Sección B.1.1.

Características validadas

- El simulador en topologías pequeñas
- Funcionamiento básico de BGP-4. No incluye el uso de políticas ni atributos
- Correcta creación de los archivos de trazas

Resultados

El simulador se ha comportado correctamente simulando correctamente la topología propuesta.

El tiempo que tarda la simulación en ejecutarse es aproximadamente 600 milisegundos.

3.1.2. Escenario complejo BGP-4

Objetivo

Validar las características avanzadas del protocolo BGP-4 en escenarios de tamaño medio. La configuración puede verse y en la Sección B.1.2.

Características validadas

- Comportamiento del protocolo con un número de nodos inferior a 20
- Configuración compleja de los parámetros de BGP-4 en lo que se incluye. Distinto nivel de decisión de rutas, puerto distinto al puerto por defecto
- Características avanzadas de BGP. Entre las que se incluyen: Reflector de Rutas, uso de atributos avanzados, políticas.
- Características del simulador que consiste en la posibilidad de que las interfaces usen determinadas conexiones del simulador.
- Eventos del simulador de caída y recuperación

Resultados

El simulador se ha comportado correctamente simulando correctamente la topología propuesta.

Los tiempos del simulador han sido:

- Tiempo de ejecución: 8745 ms
- Tiempo de simulación: 7990 ms

3.1.3. Escenarios Extensos BGP-4

Objetivo

Comprobar la escalabilidad del simulador en escenarios de gran tamaño. La máquina usada para esta prueba tiene las siguientes características.

- Procesador: Intel Pentium IV 2800 MHZ
- Memoria Ram: 1 GB
- Sistema operativo: Debian lenny 5.02
- Máquina virutal: JVM 1.6
- Memoria Maxima de la Máquina virtual: 512 Mb

Resultados

Las configuraciones planteadas son las descritas en Sección B.1.3 y Sección B.1.4. Sección B.1.4 además de añadir más nodo al anterior, intenta ver el impacto del aumento de conexiones en el simulador.

Tiempos de simulación

- Tiempo de simulación de Sección B.1.3: 103022 ms
- Tiempo de simulación de Sección B.1.4: 346506 ms

Conclusiones

Las simulaciones con más de 100 nodos ya son difíciles de predecir y es una situación en donde se necesitan herramientas como simuladores para estudiar comportamiento del protocolo.

Ante un escenario de estas características la respuesta ha sido óptima, pese a la máquina utilizada no disponía de gran potencia, se ha conseguido simular más de 130 nodos. Los tiempos son más que aceptables, y los eventos que ha habido garantizan que los anuncios se han distribuido por todos los nodos del sistema.

En la segunda simulación se observa un gran aumento del tiempo de simulación. Esto es debido a que la relación que existe entre el número de nodos simulados y el tiempo de simulación es exponencial.

Partiendo de la segunda simulación; si se dividen los AS en grupos, se pueden hacer tres, los pertenecientes al primer, segundo y tercer millar; los tiempos de simulación son 20 segundos para los dos primeros y 60 para el tercer grupo. Según estos datos, se puede deducir que penaliza más el número de conexiones entre nodos que el número de nodos en si.

Por lo tanto el simulador, pese a tener unos costes de exponenciales, es capaz de realizar simulaciones con un gran número de nodos.

Capítulo 4

Estudio de BGP-4

4.1. Necesidad de estudio

BGP-4 es un protocolo utilizado para comunicar AS e intercambiar información entre ellas. Esta comunicación es distinta a la que hay entre los nodos del AS y ésta pensada para redes de nodos heterogéneas en donde los nodos no necesitan conocer toda la topología, solo sus nodos vecinos.

El objetivo es que la información sobre el acceso a las distintas redes llegue a cada uno de los nodos BGP-4 buscando una comunicación simple y eficiente. para conseguirlo la comunicación se realiza únicamente entre nodos vecinos. Busca evitar las posibles de oscilaciones , redundancia de información, ineficiencia de los mensajes etc.

El protocolo usa políticas y atributos en el proceso de decisión, el cual decide si la información recibida debe ser utilizada y transmitida a los nodos vecinos.

Este protocolo es usado para el intercambio de información sobre rutas a gran escala por la mayoría de Proveedor de Internet (ISP) y da acceso a los usuarios los contenidos de Internet. Esto le convierte a BGP-4 en un punto critico para el funcionamiento de Internet y la experiencia final de los usuarios en la red.

El estudio del protocolo ayuda a mejorar esta experiencia y permite prever posibles problemas y conflictos.

4.2. Proceso de decisión

El proceso de decisión es uno de los puntos críticos del protocolo ya que es el encargado de tomar las principales decisiones referentes a la información que tiene que ser trasmitida. Analizando el proceso de decisión se pueden entender mejor el comportamiento de BGP-4

Se define el grafo de costes como un grafo valuado y dirigido de una red

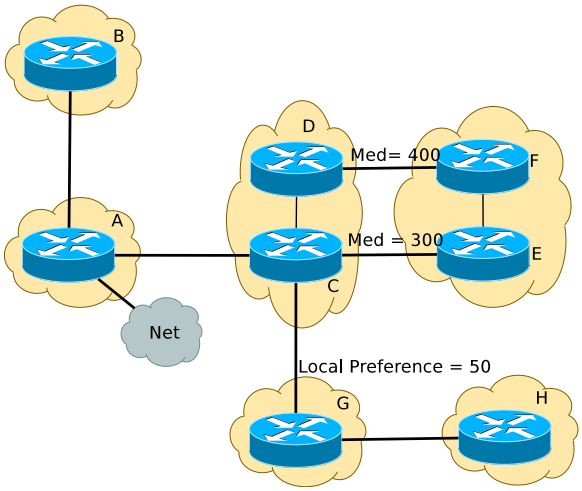


Figura 4.1: Ejemplo de topología BGP-4

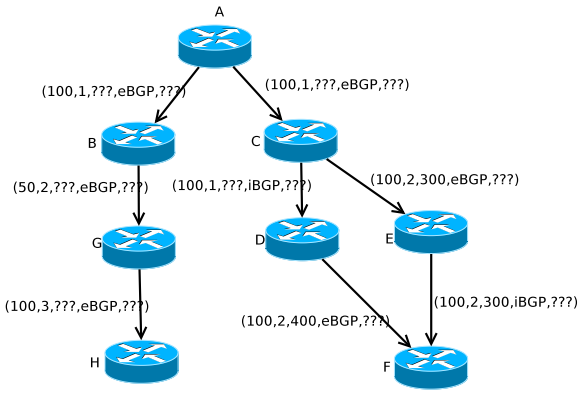


Figura 4.2: Gráfico de Costes de la topología Figura 4.1

relativa a una ruta. Los nodos son los propios nodos de la red, el flujo de la información entre los diferentes nodos son las aristas y los pesos representan el coste de publicar la información sobre una red por ese camino a un vecino.

El valor del coste viene determinado por los atributos de la información publicada que se trasmite entre los nodos y que recopila información del camino que ha visitado.

El proceso de decisión de BGP-4 selecciona que información debe usar y publicar utilizando el grafo de costes para seleccionarla y controlarla.

Los atributos que miden este coste son:

- *Local Preference*: Es el valor que más peso tiene en el proceso. Suele tomar por defecto el valor de 100. Este valor es utilizado para dar preferencia a la información dependiendo del origen de la misma independientemente del número de AS de distancia.

- *AS_PATH (ASPATH)*: Este valor almacena los números de todos los AS que ha recorrido la información hasta el momento. El peso de este valor viene determinado por la cantidad de los AS recorridos.

- *Multi-Exit Discriminator (MED)*: Este atributo toma valores numéricos. Cuanto mayor sea el valor, menor es el peso de la ruta. Es utilizado cuando la información entre dos AS es recibida por dos caminos distintos dando prioridad a una de ellos.

- *Originator*: El siguiente nivel comprueba si el origen del anuncio es interno o externo. Llegados a este nivel, las rutas internas son preferidas a las que tienen origen externo.

- *Tie Braking*: No es un atributo propiamente si no una forma de romper el empate en caso de que siga existiendo. El método para romperlo puede ser cualquiera como la dirección IP o de forma aleatoria.

El peso de la información se puede definir por tanto como una tupla con cinco campos: *Local Preference, ASPATH, MED, Originator* and *Tie Breaking*. Para la Figura 4.1 su grafo de costes está reflejado en la Figura 4.2.

Uno de los principales problemas del proceso de decisión de BGP-4 es que el grafo de costes puede no ser monótonamente creciente lo que causa que el protocolo no actúe como se espera. Dos ejemplos de este son los problemas de Oscilación y las *Widgies*

4.2.1. Oscilación

Algunas de las configuraciones de redes BGP-4 se observa como el sistema oscila entre varios estados hasta que consigue llegar a un punto estable o incluso no llega a estabilizarse nunca.

4.2.1.1. Analisis

La Figura 4.3 muestra una topología que esta continuamente oscilando y no alcanza un estado estable en ningún momento. Los resultados de la simulación se describen en el Sección B.1.5.

La topología mostrada en la Figura 4.3 se compone de tres nodos. El Nodo A tiene el acceso a la red publicada y hace el primer anuncio a sus dos vecinos, los nodos B y C. Mediante el atributo *local preference*, las rutas que son anunciadas desde el nodo A tienen menos preferencia para los nodos B y C.

En la situación inicial, los nodos B y C tienen sesión BGP-4 cerrada. cuando se estabilizan las sesiones BGP-4 para la ruta anunciada, los nodos B y C solo cuentan con la información que proviene del nodo A por lo que

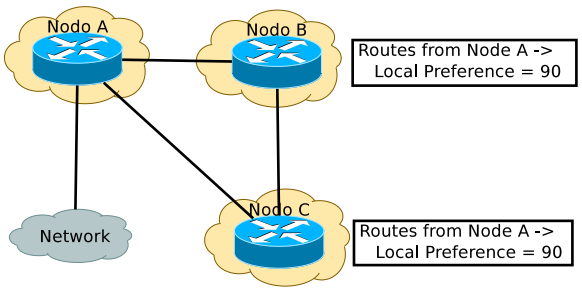


Figura 4.3: Ejemplo de configuración BGP-4 que oscila

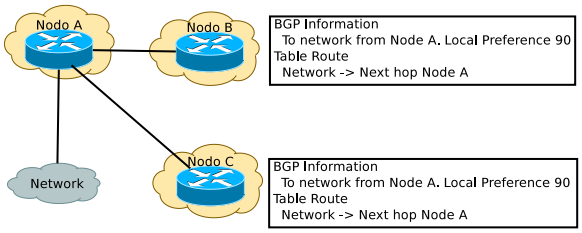


Figura 4.4: Estado 1: Inicial

se convierte en la única ruta para acceder a los contenido y ésta es añadida en la tabla de rutas del nodo como muestra la Figura 4.4.

En un momento determinado comienza la comunicación BGP-4 entre los nodos B y C. Cuando las sesiones se estabilizan comienzan a enviar la información que ambos poseen relativa a la red anunciada en un primer momento por el nodo A, como muestra la Figura 4.5.

La información es recibida por los nodos B y C. Cada uno de los nodos disponen de dos anuncios de la misma ruta, una que tiene el origen el nodo A y la que ha sido anunciada por B y C. La configuración de red hace que se prefiera la nueva a la antigua y, por lo tanto, ésta es utilizada en la tabla de rutas como muestra la Figura 4.6.

El cambio en la tabla de rutas ocasiona a su vez un *withdraw*[1], a los nodos vecinos, Figura 4.7, comunicándoles que la ruta a la red no se encuentra disponible a través de él. Éstos *withdraw* son posteriores a la recepción de los anuncios siempre y cuando ambos nodos hayan conseguido realizar el anuncio inicial.

Cuando los *withdraw* son recibidos se elimina la información que había sido anunciada anteriormente, Figura 4.8. Los nodos restauran la ruta original a la red a través del nodo A y vuelven a generarse nuevos anuncios de esta ruta. Comienza así el sistema a oscilar, ya que se vuelve al estado inicial, Figura 4.4.

La causa de que el sistema cicle se debe a que el grafo de coste no es

[1]Mensaje enviado por una sesión BGP-4 para anular una ruta previamente anunciada

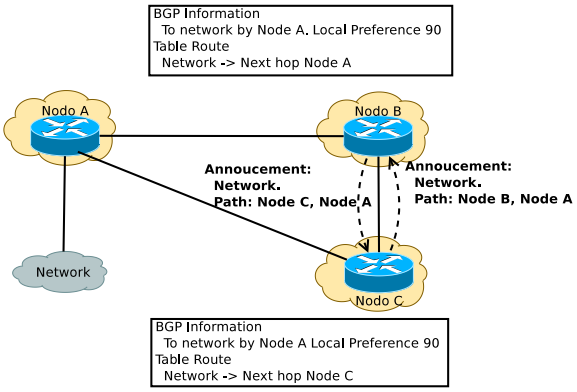


Figura 4.5: Estado 2: Intercambio de información BGP-4 entre los Nodos B y C

creciente, lo que produce que los nodos B y C prefieren la información de su compañero a la del nodo A. Esta situación finalizaría en un nodo si:

- Procesa el anuncio de la ruta recibida del vecino entre la recepción del *withdraw* y el anuncio que este mensaje produce sea enviado.
- Procesa el *withdraw* de la ruta recibida del el vecino entre la recepción del anuncio y antes de que el *withdraw* que produce este anuncio sea enviado.

Si alguna de estas condiciones se diera en uno de los nodos, uno ellos tendría ambas informaciones para decidir cuál se queda y el otro no, no se crea conflicto y finalmente se consigue una situación estable. Sin embargo hay dos posibles estados finales igualmente probables, con lo cual aparecen también problemas de convergencia. Los posibles estados finales son la Figura 4.9a y la Figura 4.9b.

Si se profundiza más en el análisis del escenario se observa que, si existe una primera iteración de la oscilación oscilación el sistema, éste no consigue estabilizarse. Se puede demostrar que en este escenario siempre ocurre.

Tomando como caso base que haya al menos una iteración de la oscilación, analizamos las dos condiciones:

- Los dos anuncios son enviados en un primer momento sin recibir un anuncio que lo anule. Esto esta implícito en la condición de que exista al menos una iteración.
- El primer *withdraw* se envía antes de recibir el primer *withdraw* originado por el vecino al recibir el primer anuncio. Aunque esta implícito en la premisa de que exista una iteración, si ocurre la primera condición, el *withdraw* nunca es recibido antes de que sea enviado el primero.

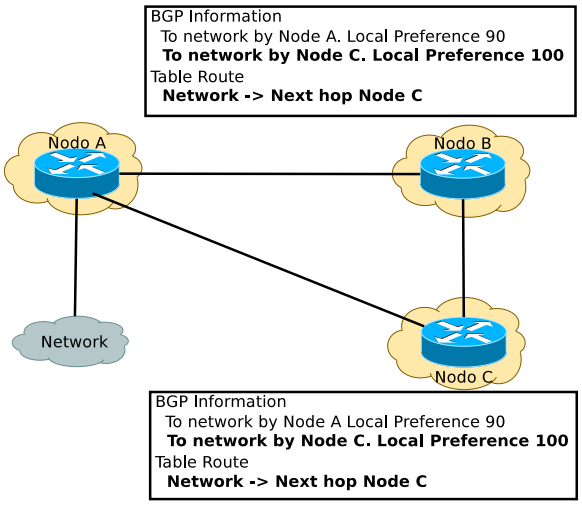


Figura 4.6: Estado 3: La información recibida es contrastada y añadida a la tabla de rutas

Esto se debe a que el anuncio enviado por el vecino no puede haber ocurrido antes que el envío del *withdraw*; es decir, si un nodo recibe un anuncio después de haber enviado su propio anuncio, el *withdraw* que genera el anuncio recibido es enviado después del anuncio enviado y, debido a que TCP garantiza el orden de los mensajes recibidos, el orden de recepción se mantiene en el vecino. El anuncio es, por tanto, procesado y genera otro *withdraw* antes de que el *withdraw* recibido sea procesado.

.

El caso base no solo nos garantiza existe al menos una iteración, si no que el tipo de los mensajes siempre alterna entre anuncio y *withdraw* tanto recibido como enviado. para el resto de casos realizamos una inducción fuerte para demostrar que siempre oscila. Los casos recursivos son dos:

- Para todo "M" y "N" tal que m = n + 1 y que representan el orden de los anuncios o *withdraws* enviados y procesados por un nodo, suponiendo que se mantiene el orden de los anuncios y *withdraws* hasta el N-*Withdraw* que genera el M-Anuncio en cada nodo. Por inducción podemos decir que el N-anuncio recibido por ambos nodos es procesado antes del N-*Withdraw*. El N-Anuncio genera y envia el N-*Withdraw*, posteriormente se procesará el N-*Withdraw* recibido que genera a su vez el M-Anuncio que, asumiendo que usamos TCP, será recibido y procesado después del N-*Withdraw* evitando que BGP-4 se estabilice. Concretando para dos nodos B y C, el N-Anuncio-C recibido el nodo B es procesado antes que el N-*Withdraw*-C, este ha sido generado por

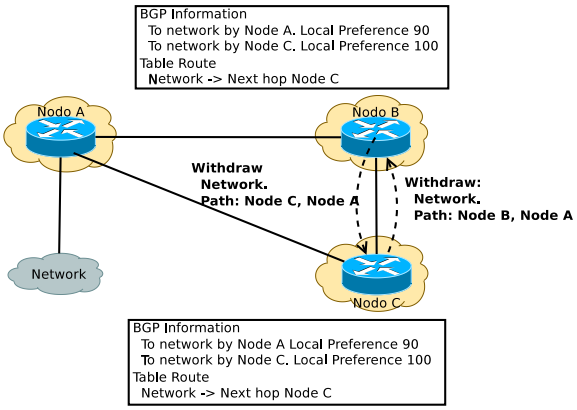


Figura 4.7: Estado 4: Envío de *withdraw* a los vecinos

N-Anuncio-B, N-*Withdraw*-C envía M-Anuncio-B que es posterior por tanto a N-Anuncio-B. Para el nodo C, el N-Anuncio-B recibido por este nodo es procesado antes que el N-*Withdraw*-B, este ha sido generado por N-Anuncio-C, N-*Withdraw*-B envía M-Anuncio-C que es posterior por tanto a N-Anuncio-C. Se entiende por N-Anuncio-C, el enésimo anuncio realizado por el nodo C.

- Para todo "M", "N" tal que m = n + 1 y que representan el orden de los anuncios o *withdraws* enviados y procesados por un nodo, suponiendo que se matiene el orden de los anuncios y *withdraws* hasta el M-Anuncio que genera el M-*Withdraw* en cada nodo. Por inducción podemos decir que el N-*Withdraws* recibido por ambos nodos es procesado antes del M-Anuncio. El N-*Withdraw* genera y envia el M-Anuncio, posteriormente se procesará el M-Anuncio recibido que genera a su vez el M-*Withdraw*, asumiendo que usamos TCP, sera recibido y procesado despues del M-Anuncio, evitando que BGP-4 se estabilice. Para dos nodos B y C, el N-*Withdraws*-C recibido recibido por el nodo B es procesado antes que el M-Anuncio-C. Este ha sido generado por N-*Withdraw*-B, M-Anuncio-C envia el M-*Withdraws*-B que es posterior por tanto a N-*Withdraws*-C. Para el nodo C el N-*Withdraws*-B recibido por este nodo es procesado antes que el M-*Withdraw*-B, este ha sido generado por el N-Anuncio-C, M-Anuncio-B envia M-*Withdraw*-C que es por tanto posterior a N-*Withdraw*-C

Una de las condiciones de este razonamiento es que, en el proceso de decisión de un nodo, el mensaje es procesado en el momento en que se recibe. Si esta condición no se da y existe un tiempo entre la recepción y procesado, otro mensaje puede ser recepción y procesado en este periodo de tiempo y causar que el sistema deje de oscilar.

En topologías más complejas, también puede ocurrir condiciones para que

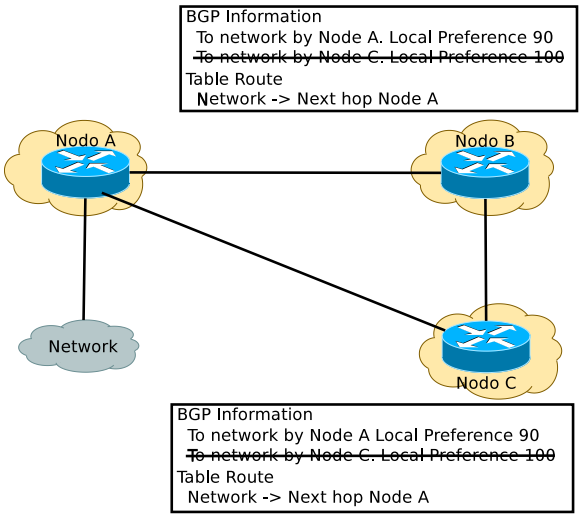


Figura 4.8: Estado 5: La información es eliminada y las rutas son borradas de la tabla de rutas.

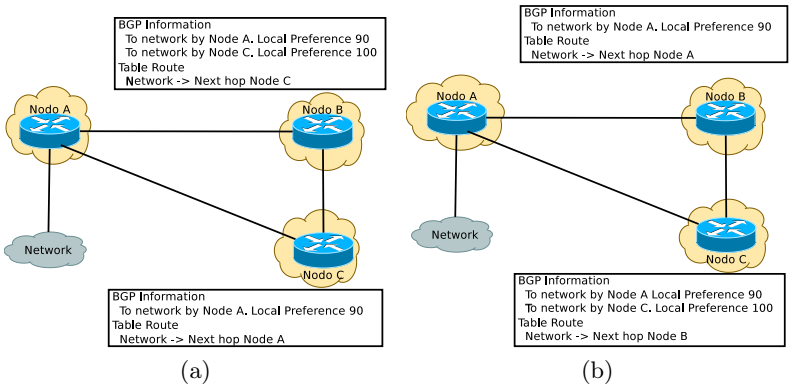


Figura 4.9: Posibles estados finales de la Figura 4.3

se estabilice BGP-4, aunque no te garantizan que se produzcan oscilaciones. En estas topologías al menos uno de los AS que causan que el sistema cicle, deben tener dos nodos o más y que las sesiones eBGP se encuentren en distinto nodo.

La Figura 4.10 muestra una topología en donde la oscilación de la red puede terminar. El periodo de tiempo entre la recepción del anuncio por parte B-2 a través desde el nodo C y el posterior *withdraw* del nodo B-2 al nodo C no es despreciable, puesto que el anuncio tiene que llegar hasta el nodo B-1 para que sea este el que genere el *withdraw* a B-1 que posteriormente será enviado desde el nodo B-2 al nodo C. Si en este tiempo el nodo B-2 recibe el anuncio del nodo C con la ruta a través de él, el Nodo B-2 no retransmitirá

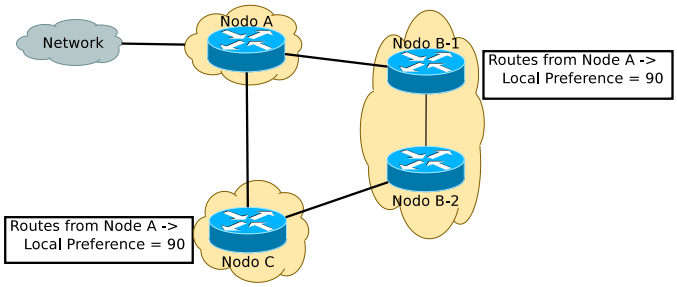


Figura 4.10: Ejemplo de configuración BGP-4 que oscila

el *withdraw*. Lo mismo ocurre desde que el nodo B-2 recibe el *withdraw* de el nodo C hasta que envía el anuncio desde el nodo B-2 al nodo C. Lo resultados de la simulación están en la Sección B.1.6.

Como conclusión se puede decir que toda conexión entre una pareja de nodos nodos es susceptible de sufrir oscilaciones si en una topología se puede acceder a una red por más de un camino y, por políticas o valores de a atributos, ambos prefieren las rutas que recorren el otro nodo de la pareja. Este efecto es inevitable ya que los nodos no tienen conciencia del problema, sin embargo si se introducen retardos aleatorios en el proceso de decisión puede reducir el número de iteraciones de la oscilación

En escenarios grandes, en los puntos que, por causa del uso de atributos, se produzcan costes negativos en el grafo de costes, son susceptibles de sufrir este efectos. En la Figura 4.11 se puede ver una topología y los puntos susceptibles a oscilar. La simulación de esa topología esta en el Sección B.1.7.

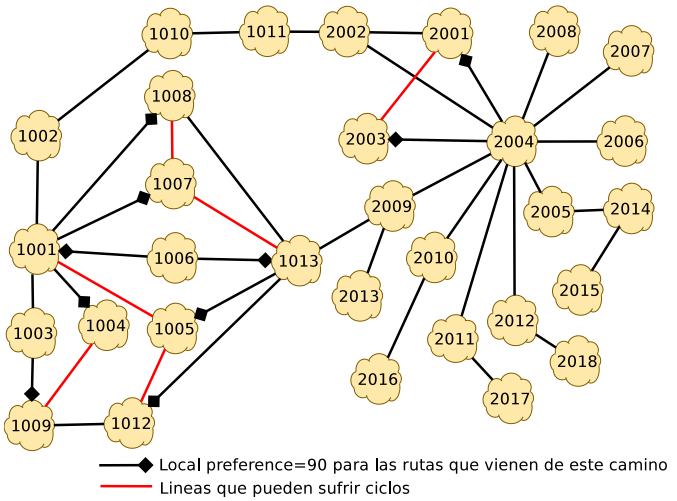


Figura 4.11: Ejemplo de configuración BGP-4 que oscila

4.2.1.2. Impacto de Oscilación en la Red

Cuando se producen oscilaciones en un sistema, los afectados son:

- Los nodos que no consiguen estabilizar sus sesiones BGP-4. Se produce un aumento del tráfico de mensajes. y la información no se estabiliza hasta que la oscilación acaba.

- El resto de nodos de la topología que reciben la información residual de la oscilación. Éstos también se ven afectados porque la información que están recibiendo constantemente de las rutas afectadas causa que la elección de la mejor ruta por el proceso de decisión sea distinto dependiendo de como cambie dicha información recibida y por tanto se produzcan oscilaciones en los nodos.

- Los usuarios finales que utilizan la información para acceder a los contenidos también se ven afectados, ya que durante el tiempo que dure la oscilación el acceso a las rutas publicadas por los nodos es intermitente.

4.2.1.3. Conclusión

La oscilación se produce cuando se empiezan compartir rutas entre dos nodos y el grafo de costes para esos nodos no es creciente.

Si bien estas oscilaciones ocurren frecuentemente en sistemas de gran tamaño, éstas suelen estabilizarse tras un corto periodo de un tiempo y pueden ignorarse.

Sin embargo en determinadas ocasiones puede ocurrir que el tiempo que tarda en estabilizarse un sistema que oscila sea mayor o incluso no consiga estabilizarse. Si esto sucede, las consecuencias no son triviales y, durante el tiempo de oscilación, se pierde el acceso estable a la información publicada

Otros de los efectos negativos que produce la oscilación es que el sistema no converge. Ya que existen al menos dos estados finales distintos.

Las oscilaciones no se pueden evitar, ya que el uso de atributos produce que el grafos de costes no sea creciente. Sin embargo, introducir retardos aleatorios entre los anuncios, puede mitigar los efectos de las oscilaciones.

4.2.2. Wedgies

BGP-4 es una herramienta determinista para compartir rutas. Sin embargo existen ciertas configuraciones para las cuales hay más de un posible estado final. Estos estados estables pueden ser alcanzados por BGP-4 de manera no determinista y son denominados *BGP wedgie*. T. Griffin fue el primero en dectar este problema y ha realizado muchos estudios sobre el impacto de las mismas[8].

4.2.2.1. Análisis

La Figura 4.12 muestra una configuración BGP susceptible de sufrir una *wedgie*. La topología centraliza el tráfico por el *Primary Link* o enlace principal y dispone de un *Backup link* o enlace de seguridad destinado a evitar perder el acceso a los contenidos debido a una caída del enlace principal. Para conseguir esto se aplican políticas de atributos en el AS 32080 haciendo a las rutas anunciadas por ese enlace, menos preferibles.

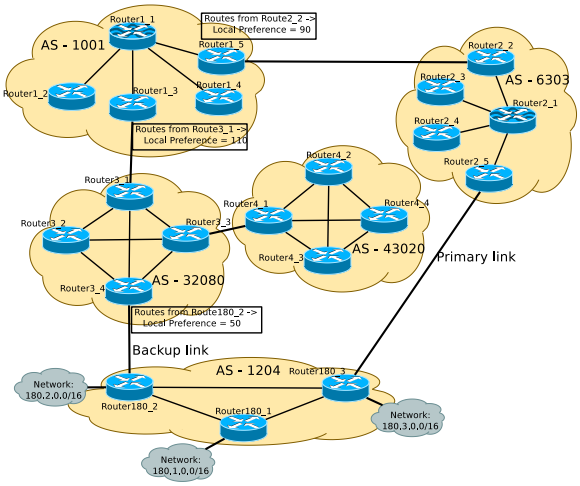


Figura 4.12: Ejemplo de escenario que produce *wedgie 3/4*

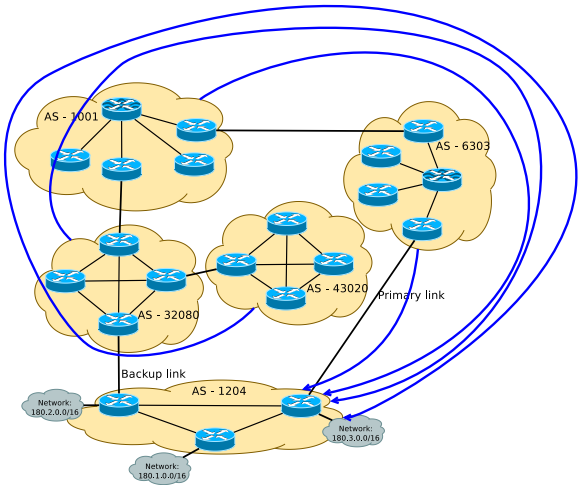


Figura 4.13: Configuración buscada de la topología mostrada en la Figura 4.12

El AS 1001 tiene preferencia por las rutas anunciadas por el AS 32080 y aplica políticas para ello en los nodos frontera.

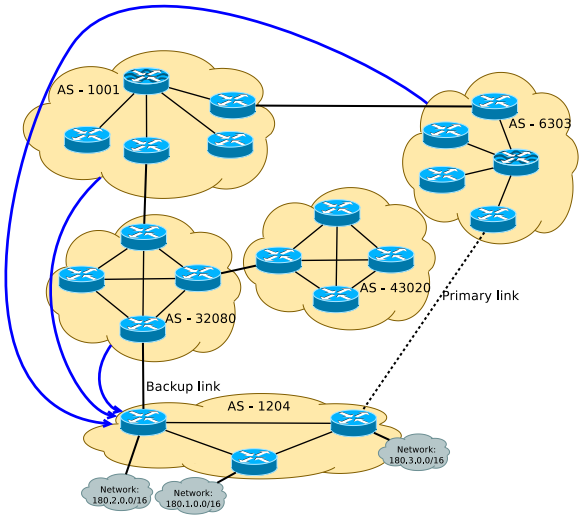


Figura 4.14: Caída del enlace principal o *Primary Link*, Figura 4.12

La Figura 4.13 representa el estado estable final deseado. Las lineas representan el tráfico de datos canalizado por el enlace principal.

En un momento determinado se pierde el acceso a los datos por una caída en el enlace principal. Cuando se alcanza una estado estable, se observa como el acceso a los contenidos se hace mediante el enlace de seguridad, Figura 4.14.

Al restablecerse en enlace principal los AS 1001, 32080 y 43020 siguen usando el enlace de seguridad para acceder a los datos, Figura 4.15.

Para volver al estado final deseado es necesario la caída del enlace de seguridad eliminando toda referencia a esa ruta, Figura 4.16.

En el estado representado por la Figura 4.14, el nodo frontera entre los AS 1001 y 6303 contiene la información de acceso de la ruta usando el enlace de seguridad. Se ha aumentado la prioridad de la información por medio del atributo *local preference*. En el momento de restablecer la conexión, Figura 4.15, la información publicada por el AS 6303 es menos prioritaria y, por lo tanto, descartada. Para recuperar el estado deseado es necesario eliminar la información publicada por el enlace de seguridad cortando la comunicación por el enlace de seguridad. Esta *Wedgie* se denomina *Wedgie 3/4*.

La Figura 4.17 Muestra un escenario donde hay más de un enlace de seguridad. Se quiere dirigir de nuevo todo el tráfico por el enlace principal, Figura 4.18, y se configuran el atributos *local preference* para que esto ocurra.

Al perder la comunicación en el enlace principal, el tráfico se redirige por el enlace de seguridad.

Al restablecerse el tráfico del enlace principal, el tráfico de los AS 1001, 3280 y 43020 siguen usando el enlace de seguridad, Figura 4.19, por el mismo motivo que el anterior ejemplo. Hasta este punto todo es igual que en el

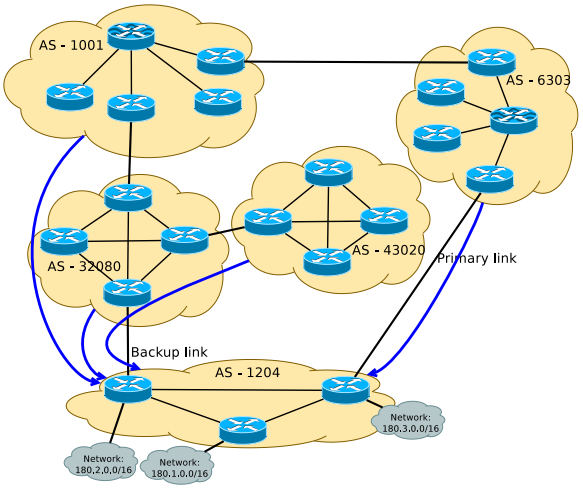


Figura 4.15: Se restablece enlace principal o *Primary Link*, Figura 4.12

anterior ejemplo.

Para eliminar las referencias al enlace de seguridad se corta su comunicación. Sin embargo en esta ocasión el tráfico no vuelve a utilizar el enlace principal si no que usa el enlace de seguridad secundario Figura 4.20.

Al restablecerse la comunicación del enlace de seguridad, el AS 1001, punto crítico del sistema, no utiliza la información facilitada por el AS 6303 y no restablece la ruta atraviesa el enlace principal. La ruta que atraviesa el enlace de seguridad secundario es retirada cuando el AS 43020 publica la ruta que atraviesa el enlace de seguridad y que atraviesa el AS 43020. Esta ruta es preferida a que utiliza la que atraviesa en enlace principal. Simultáneamente, el AS 32080 publica la su ruta que atraviesa enlace de seguridad, ya que es la mejor la información disponible, y es ésta la que finalmente se utiliza en el estado estable, Figura 4.19. Nótese que las rutas anunciadas por los AS 43020 y 32080 son distintas ya que la primera atraviesa los AS 1204, 32080 y 43020, y la segunda los AS 1204 y 32080.

Para conseguir la situación deseada, Figura 4.18, se debe cortar la comunicación de todos los enlaces de seguridad, Figura 4.21, eliminando del AS 1001 todas las referencias rutas de estos enlaces, y la ruta a través del enlace principal es publicada por todos el resto de AS. Esta *wedgie* se denomina *full wedgie*

En la Sección B.1.8 y la Sección B.1.9 se ven los resultados de simular la Figura 4.12 y la Figura 4.17.

4.2.2.2. Situación real

Las topologías representadas por la Figura 4.12 y la Figura 4.17 pertenecen a escenarios teóricos, pero se basan en un modelo real cliente-proveedor

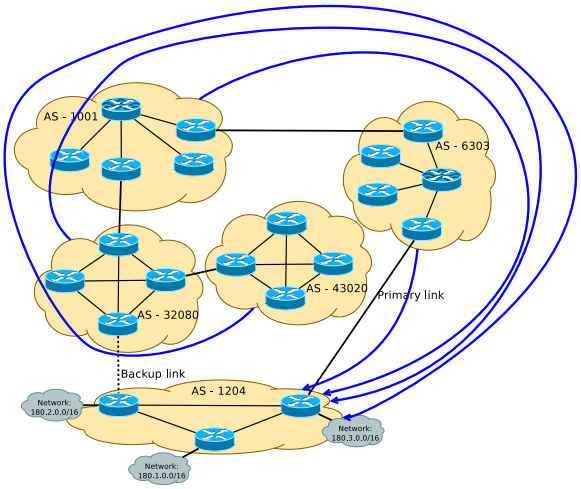


Figura 4.16: Caída del enlace de seguridad o *Backup Link*, Figura 4.12

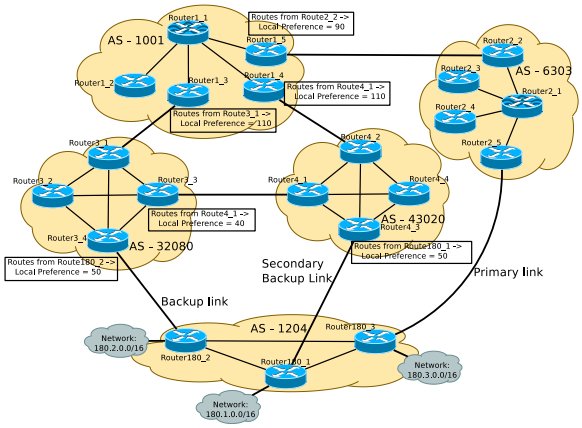


Figura 4.17: Ejemplo de escenario que produce *full wedgie*

de servicios de Internet.

En el modelo los AS 32080 y 43020 representan Proveedores de Internet que ofrecen servicios de Internet a usuarios finales. Éstos se comunican entre si como iguales y a su vez son clientes del AS 1001 el cual le ofrece acceso a los contenidos garantizando alta velocidad.

Los AS 1001 y 6303 representan proveedores de ISP que los interconectan. Su política les permite también comunicarse como iguales, *peer to peer connection*.

Por último el sistema autónomo 1204 representa el lugar donde se encuentran los contenidos a los que queremos acceder.

Los servidores de ISP pueden tener enlaces de seguridad que garantizan el acceso a los datos si ocurre alguna caída de su acceso principal por medio del AS 1001. Esto se ve reflejado en el atributo *local preference* de los AS 32080

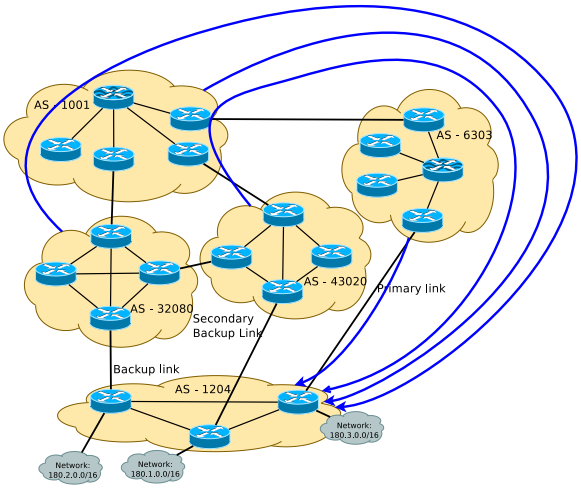


Figura 4.18: Estado final estable buscado en la configuración mostrada en la Figura 4.17

y, solo para el caso de la *full wedgie*, 43020 de los enlaces que le comunican con los contenidos, AS1204.

Los Proveedores de Internet por su parte tienen la política de usar la conexiones con sus clientes que aquellas que tengan como iguales ya que les resulta más económica. Esto se ve reflejado en el atributo *local preference* del AS 1001, el cual da más preferencia a sus clientes.

4.2.2.3. Impacto de las *wedgies*

El principal impacto sobre una red que sufre una *wedgie* es que el tráfico es dirigido por enlaces que no están preparados para ello. Las enlaces de seguridad están ideados para aceptar tráfico en situaciones criticas, y, por lo tanto, suelen tener menor capacidad.

Una *wedgie* no se puede detectar por BGP-4 utilizando métodos tradicionales, y generalmente es necesario realizar operaciones complejas que no se pueden automatizar para estabilizar la situación y requiere conocimientos avanzados del protocolo.

El impacto de las *wedgies* se ve agravado en redes como Internet donde es difícil disponer de un mapa de la red actualizado y donde los AS son entidades distintas que se gestionan de forma independientemente, lo que ocasiona que para solucionar la *wedgie* sea necesario contactar con dichas entidades.

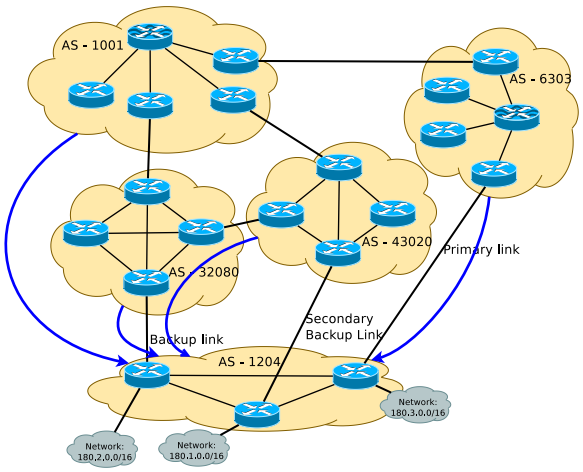


Figura 4.19: Caída del enlace principal o *Primary Link*, Figura 4.17

4.2.2.4. Conclusión

Las *wedgies* son aparentemente fáciles de detectar y solucionar. Sin embargo, si analizamos la red BGP-4 más importante, Internet, no resulta un problema trivial.

Los problemas añadidos de Internet a la *wedgies* dificultan su detección y, una vez detectados, puede ocurrir que la solución dependa de entidades distintas. Si alguna de estas entidades resulta ser una empresa de la competencia, el problema se agrava aún más.

Una entidad puede enfrentarse en un determinado momento a un problema que sea incapaz de detectar y en caso de detectarlo, incapaz de solucionar.

Extrapolando a empresas que dependan directamente del servicio que reciben o proveen, las consecuencias pueden ser muy graves y perdidas económicas.

4.2.3. Proceso de decisión: Conclusión

El proceso de decisión es una parte compleja de BGP-4 y tiene que ser objeto de estudio para poder detectar fallos y problemas que pueda tener.

Uno de los principales puntos críticos se encuentra en el peso que tiene cada ruta, éste depende de factores que hacen que los pesos puedan tomar valores negativos, lo que produce grafos de costes que no son crecientes.

A consecuencia de esto se pueden producir efectos no deseables como oscilaciones y *wedgies* en la red. Las consecuencias de las mismas pueden ser muy graves y las soluciones no son triviales. Muchas de estas soluciones requieren de personas cualificadas supervisen la red.

Prevenir los problemas es complicado y en muchas ocasiones inevitable, debido a la naturaleza del grafo de costes. Platear soluciones más profundas

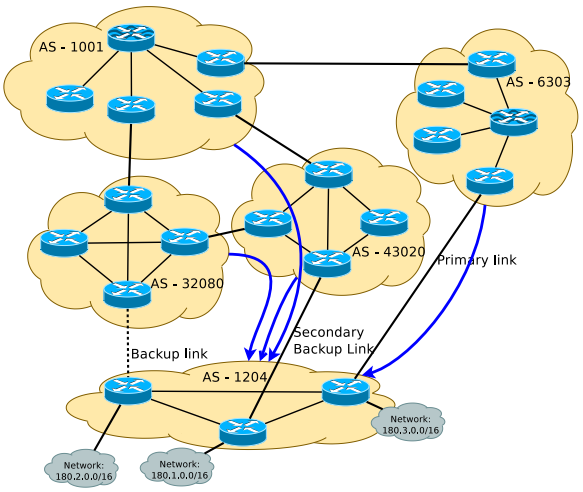


Figura 4.20: Caída del enlace de seguridad o *Backup Link*, Figura 4.17

conllevan modificar el proceso de decisión y, por lo tanto, la forma en que BGP-4 trabaja, lo que puede producir la perdida de las ventajas que este protocolo ofrece.

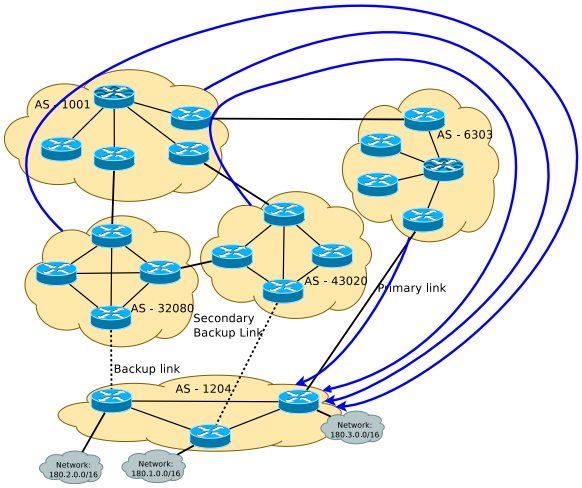


Figura 4.21: Caída del enlace secundario de seguridad o *Secondary Backup Link*, Figura 4.17

Capítulo 5

Conclusión

5.1. Simulador

El protocolo BGP-4 es uno de los pilares que hace que Internet funcione correctamente. Su red abarca gran cantidad de nodos que comparten su información con el resto de nodos de la red.

Sin embargo, pese a estar muy extendido, existen pocas herramientas que permitan estudiar el protocolo y las alternativas que existen son software propietario o son poco escalables.

Uno de los objetivos del proyecto ha consistido en crear un entorno de simulación completo, escalable y de fácil uso para BGP-4. Se ha integrando el lenguaje de marcas XML que permite usar herramientas para crear y automatizar la creación de topologías.

También se ha creado un sistema de trazas que facilitan el análisis del protocolo después de la simulación.

El simulador se ha hecho más modular, lo que facilita la implementación de nuevos protocolos, para su validación, y herramientas para el simulador. La simulación de estos se benefician del uso del XML al igual que BGP-4.

El resultado ha sido un simulador fácil de usar y escalable. No se sacrifica la potencia del simulador y se mantiene la integración con el lenguaje de script Tcl lo que permite crear simulaciones más complejas. Gracias a la modularización del simulacor se pueden implementar nuevos protocolos y herramientas que usen las nuevas características y se beneficien de la potencia del mismo.

5.2. Estudio BGP-4

La segunda parte ha consistido en, una vez desarrolladas las herramientas necesarias para el estudio de BGP-4, utilizarlas para comprender mejor el protocolo.

Con los conocimientos adquiridos en el proyecto, el estudio se ha centrado en el proceso de decisión, que es el punto crítico de BGP-4 y donde se registran la mayor cantidad de problemas.

Se ha observado que los principales puntos conflictivos del proceso de decisión son el uso de políticas y atributos para calcular los pesos de las rutas de cara a su publicación a los nodos vecinos.

El uso de estas característica causa efectos no deseable son la aparición de oscilaciones y *wedgies*. Estos fenómenos difíciles de prever en redes complejas.

Cuando redes de gran tamaño, como Internet, sufren oscilaciones y *wedgies*, los problemas que éstas acarrean pueden derivar en pérdidas económicas en empresas del sector de la telecomunicación.

Apéndice A

Apendice: Uso del simulador

A.1. Ejecución del simulador

Para ejecutar el simulador, basta con ejecutar el archivo ejecutable.

Si se desea ejecutar desde el código, la clase principal donde se encuentra la consola del simulador es `drcl.ruv.System`.

Opcionalmente existe una clase que te permite generar ficheros base para crear simulaciones. Esta clase se encuentra en `tid.util.XMLGenerator`

A.2. Ejecución de simulaciones

Una vez arrancado el simulador se pueden cargar las simulaciones de los ficheros XML. Para hacerlo es necesario lanzar un script Tcl. El script `baseScript.tcl` te solicita los ficheros de topología y eventos.

A.3. Estructura de los ficheros XML

La simulación consta de dos ficheros XML que almacenan lo necesario para la simulación. El fichero de la topología almacena la configuración de la red, los nodos y los protocolos usados. El fichero de eventos almacena la duración de la simulación y que ocurre durante la misma.

El formato de los ficheros XML se pueden consultar en la web

- Topología: http://www.xp-dev.com/wiki/79623/TopologyXML
- Eventos: http://www.xp-dev.com/wiki/79623/EventsXML

Apéndice B

Apendice: Configuraciones

B.1. Escenarios BGP-4

B.1.1. Escenario básico

Topología

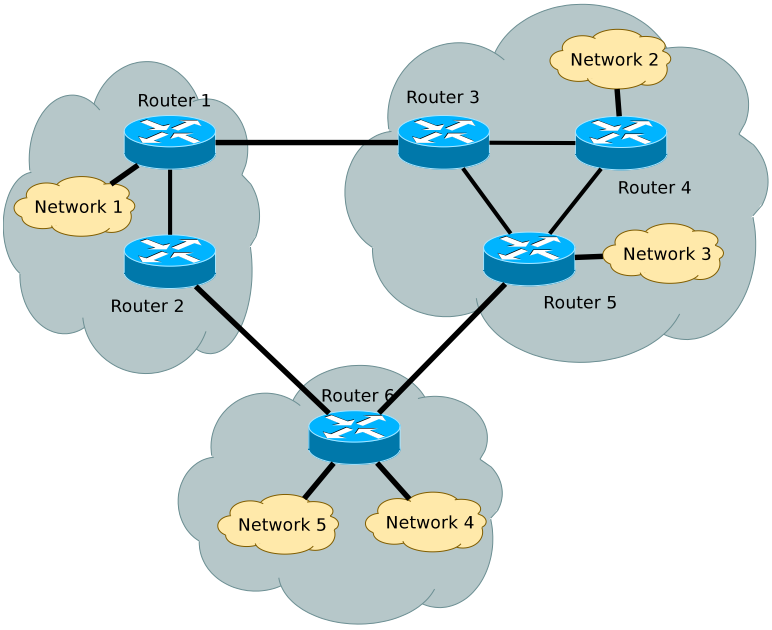


Figura B.1: Ejemplo 1: Topología de un escenario básico.

Configuración

La configuración de los nodos es la siguiente.

```
Router 1:
  Interfaces:
    - principal: 10.0.1.1
  BGP:
    - Interfaz: principal
    - AS num =1001
    - Neighbours:
      - Address = 10.0.1.2, Remote-As = 1001
      - Address = 10.0.2.3, Remote-As = 1002
    - Networks
      - Network 1 = 1.0.0.0/8

Router 2:
  Interfaces:
    - principal: 10.0.1.2
  BGP:
    - Interfaz: principal
    - AS num =1001
    - Neighbours:
      - Address = 10.0.1.1, Remote-As = 1001
      - Address = 10.0.3.6, Remote-As = 1003

Router 3:
  Interfaces:
    - principal: 10.0.2.3
  BGP:
    - Interfaz: principal
    - AS num =1002
    - Neighbours:
      - Address = 10.0.1.1, Remote-As = 1001
      - Address = 10.0.2.4, Remote-As = 1002
      - Address = 10.0.2.5, Remote-As = 1002

Router 4:
  Interfaces:
    - principal: 10.0.2.4
  BGP:
    - Interfaz: principal
    - AS num =1002
    - Neighbours:
      - Address = 10.0.2.3, Remote-As = 1002
      - Address = 10.0.2.5, Remote-As = 1002
    - Networks
      - Network 2 = 2.4.0.0/16
```

```
Router 5:
  Interfaces:
    - principal: 10.0.2.5
  BGP:
    - Interfaz: principal
    - AS num =1002
    - Neighbours:
      - Address = 10.0.2.3, Remote-As = 1002
      - Address = 10.0.2.4, Remote-As = 1002
      - Address = 10.0.3.6, Remote-As = 1003
    - Networks
      - Network 3 = 2.5.0.0/16

Router 6:
  Interfaces:
    - principal: 10.0.3.6
  BGP:
    - Interfaz: principal
    - AS num =1002
    - Neighbours:
      - Address = 10.0.2.5, Remote-As = 1002
      - Address = 10.0.1.2, Remote-As = 1001
    - Networks
      - Network 4 = 3.4.6.0/24
      - Network 5 = 3.5.6.0/24

Networks
- Network 1 = 1.0.0.0/8
- Network 2 = 2.4.0.0/16
- Network 3 = 2.5.0.0/16
- Network 4 = 3.4.6.0/24
- Network 5 = 3.5.6.0/24

Time of simulation = 10000
```

Tiempo simulado: 10000 segundos.

Archivos de la simulación

- `simluacion1-eventos.xml` Fichero de configuración de los eventos.
- `simulacion1-topologia.xml` Fichero de configuración de la topología.
- `simulacion1-script.tcl` Script TCL para lanzar la simulación.
- `trace1.tar.gz` Resultado de las trazas del simulador.

- `simulacion1.dia` Esquema de la topología.

B.1.2. Escenario avanzado

Topología

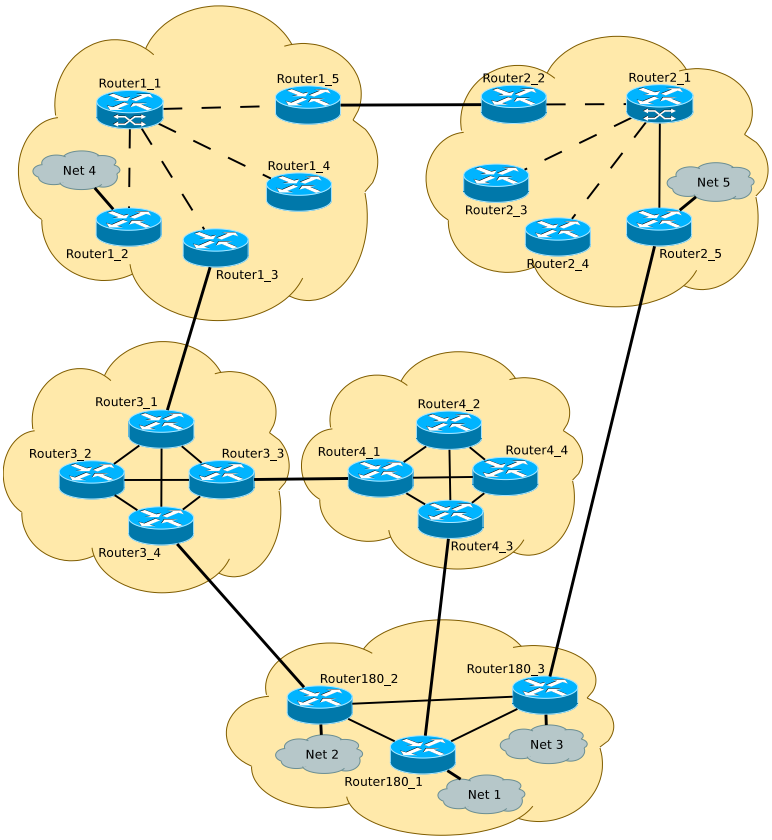


Figura B.2: Ejemplo 2: Topología de un escenario avanzado.

Configuración

La configuración consta de 21 nodos distribuidos en 5 sistemas autónomos según muestra la Figura B.2.

las configuraciones avanzadas para BGP-4

- Puerto por defecto: 5000
- Nivel de comparación de rutas: 4
- Opciones de MED: *Always Compare Med*
- Valor del atributo *local preference* por defecto: 80

Nodos con funcionalidad RR:

- `Router1_1`, Los Cliente de reflexión de rutas (RRC) son todos con todos los nodos del sistema autónomo.
- `Router2_1`, Los RRC son los nodos `Router2_2`,`Router2_3` y `Router2_4`.

Configuraciones particulares de los nodos

- `Router1_3`:
  - *Local preference*: 110 para las rutas del vecino `Router3_1`.
  - *MED*: 300 para las rutas anunciadas a `Router3_1`.
- `Router3_1`
  - *Local preference*: 100 para las rutas del vecino `Router1_3`.
- `Router3_4`
  - *Local preference*: valor 50 por defecto.
  - *Keep Alive Interval*: 400

Politicas aplicadas:

- `Router1_5`: Política de entrada. Establece, para las rutas de `Router2_2` cuando el atributo *AS-Path* contenga el valor 32080, el valor 60 para el atributo *local preference*.
- `Router2_4`: Política de entrada. Establece el atributo *Community* a `32:45` para las rutas del vecino `Router180_3`.
- `Router2_2`: Política de entrada. Descarta todas las rutas que tienen el atributo *Community* con un valor de `32:45` provenientes del nodo `Router2_1`.
- `Router180_2`: Política de entrada. Establece, para las rutas de `Router3_4` cuando el atributo *AS-Path* contenga el valor 32080, el valor 60 al atributo *local preference*.
- `Router1_4` Política de salida. No publica las rutas que contienen en el atributo *NLRI* el valor `230.*`; es decir, no publica las redes `230.1.0.0/16`, `230.2.0.0/16` y `230.2.0.0/16`; al nodo `Router3_3`.

Eventos temporales

- `T = 1000` Caída de la línea entre los nodos `Router180_2` y `Router3_4`.
- `T = 2000` Recuperación de la línea entre los nodos `Router180_2` y `Router3_4`.

- `T = 3000` Caída de la línea entre los nodos `Router1_3` y `Router3_1`.
- `T = 4000` Recuperación de la línea entre los nodos `Router1_3` y `Router3_1`.
- `T = 5000` Caída de la línea entre los nodos `Router180_2` y `Router3_4`.
- `T = 6000` Recuperación de la línea entre los nodos `Router180_2` y `Router3_4`.

Tiempo simulado: 10000 segundos.

Archivos de la simulación

- `simluacion2-eventos.xml` Fichero de configuración de los eventos.
- `simulacion2-topologia.xml` Fichero de configuración de la topología.
- `simulacion2-script.tcl` Script TCL para lanzar la simulación.
- `trace2.tar.gz` Resultado de las trazas del simulador.
- `simulacion2.dia` Esquema de la topología.

B.1.3. Escenario extenso 1

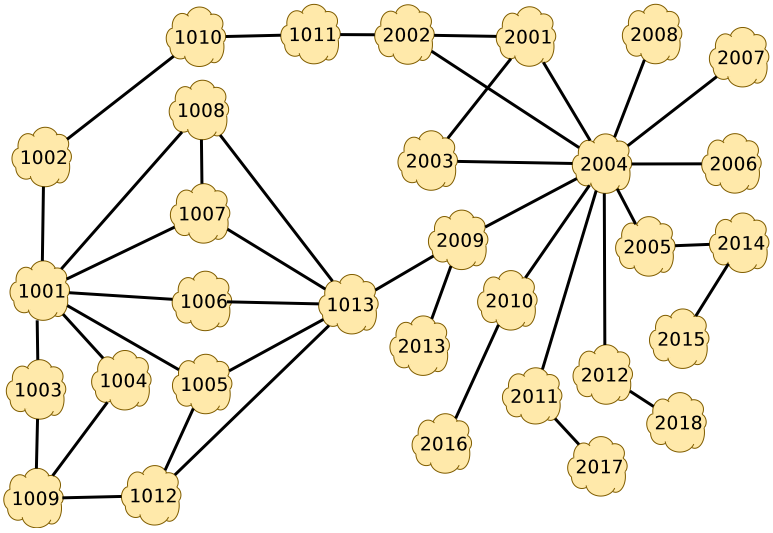


Figura B.3: Ejemplo 3: Topología de un escenario de Grandes dimensiones

.

Configuración

139 nodos distribuidos en 31 Sistemas Autónomos.
Los Sistemas Autónomos 1001, 1013 y 2004 tienen un reflector de rutas.
Las redes anunciadas son:

- Desde el AS 1008 la red `1.0.0.0/8`
- Desde el AS 1009 la red `2.0.0.0/8`
- Desde el AS 2016 la red `3.0.0.0/8`
- Desde el AS 2002 la red `4.0.0.0/8`

La distribución de los AS se puede ver en la Figura B.3.

Eventos temporales

- `T = 2000` Caída de la línea entre los AS `1013` y `2009`.
- `T = 4000` Recuperación de la línea entre los AS `1013` y `2009`.
- `T = 6000` Caída de la línea entre los AS `1011` y `2002`.
- `T = 8000` Recuperación de la línea entre los AS `1011` y `2002`.

Tiempo simulado: 10000 segundos.

Archivos de la simulación

- `simluacion3-eventos.xml` Fichero de configuración de los eventos.
- `simulacion3-topologia.xml` Fichero de configuración de la topología.
- `trace3.tar.gz` Resultado de las trazas del simulador.
- `simulacion3.dia` Esquema de la topología.

B.1.4. Escenario extenso 2

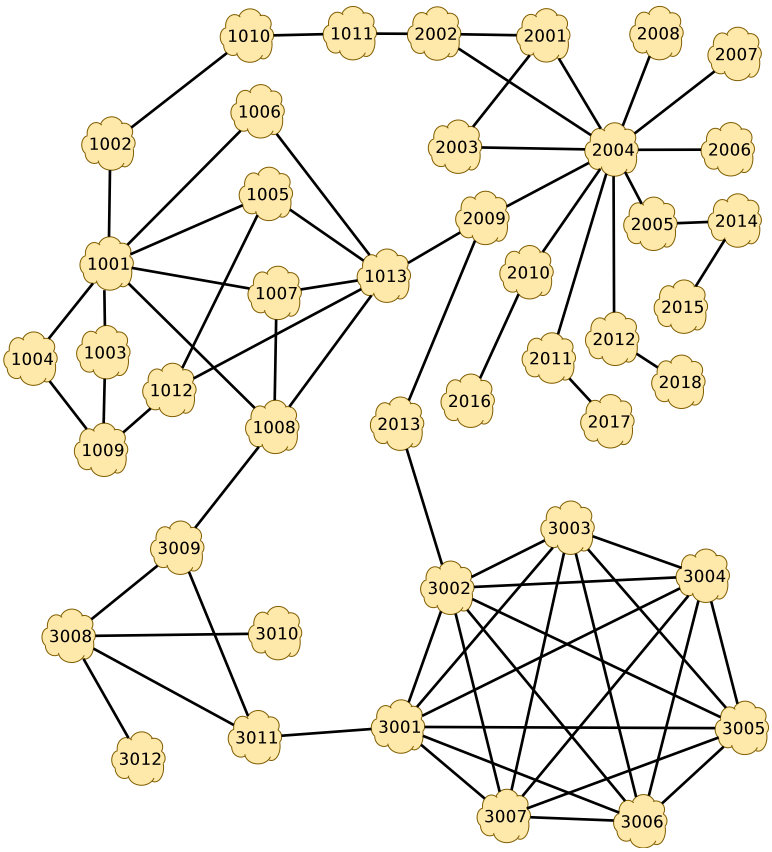


Figura B.4: Ejemplo 4: Topología de un escenario de Grandes dimensiones

.

Configuración

207 nodos distribuidos en 57 AS.
Los AS 1001, 1013, 2004 y 3008 tienen un reflector de rutas
Las redes anunciadas son:

- Desde el AS 1008 la red `1.0.0.0/8`.
- Desde el AS 1009 la red `2.0.0.0/8`.
- Desde el AS 2016 la red `3.0.0.0/8`.
- Desde el AS 2002 la red `4.0.0.0/8`.
- Desde el AS 3004 la red `5.0.0.0/8` y la `6.0.0.0/8` .

- Desde el AS 3008 la red `7.0.0.0/8`.
- Desde el AS 3012 la red `8.0.0.0/8`.

Eventos temporales

- `T = 1000` Caída de la línea entre los AS `1013` y `2009`.
- `T = 2000` Recuperación de la línea entre los AS `1013` y `2009`.
- `T = 3000` Caída de la línea entre los AS `1011` y `2002`.
- `T = 4000` Recuperación de la línea entre los AS `1011` y `2002`.
- `T = 5000` Caída de la línea entre los AS `3002` y `3012`.
- `T = 6000` Caída de la línea entre los AS `3001` y `3011`.
- `T = 7000` Caída de la línea entre los AS `2013` y `3002`.
- `T = 7750` Recuperación de la línea entre los AS `3002` y `3012`.
- `T = 8500` Recuperación de la línea entre los AS `3001` y `3011`.
- `T = 9250` Recuperación de la línea entre los AS `2013` y `3002`.

Archivos de la simulación

- `simluacion4-eventos.xml` Fichero de configuración de los eventos.
- `simulacion4-topologia.xml` Fichero de configuración de la topología.
- `trace4.tar.gz` Resultado de las trazas del simulador.
- `simulacion4.dia` Esquema de la topología.

B.1.5. Escenario que oscila: ejemplo 1

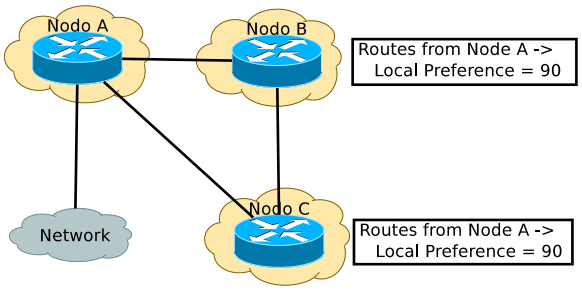


Figura B.5: Ejemplo 5: Topología de un escenario básico que oscila

Configuración

Tres nodos en diferentes AS conectados entre si.
Configuración específica de los nodos:

- `B`: *Local preference*: 90 para las rutas del vecino `A`.
- `C`: *Local preference*: 90 para las rutas del vecino `A`.

Eventos temporales

- `T = 0` Caída de la línea entre los Nodos `B` y `C`.
- `T = 1000` Recuperación de la línea entre los Nodos `B` y `C`.

Archivos de la simulación

- `ciclar1-eventos.xml` Fichero de configuración de los eventos.
- `ciclar1-topologia.xml` Fichero de configuración de la topología.
- `trace-ciclar.tar.gz` Resultado de las trazas del simulador.
- `ejemplo-ciclo-1.dia` Esquema de la topología.

B.1.6. Escenario que oscila: ejemplo 2

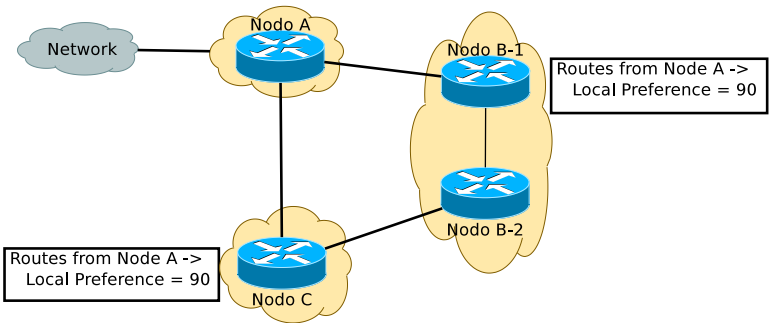


Figura B.6: Ejemplo 6: Topología de un escenario básico que oscila

Configuración

Cuatro nodos nodos repartidos en tres AS.
Configuración específica de los nodos:

- `B-1`:
  - *Local preference*: 90 para las rutas del vecino `A`.
- `C`
  - *Local preference*: 90 para las rutas del vecino `A`.

Eventos temporales

Tiempo de la simulación: 10.000 segundos

- `T = 0` Caída de la conexión entre `B-2` y `C` y entre `B-1` y `B-2`.
- `T = 1000` Recuperación de la conexión entre `B-2` y `C` y entre `B-1` y `B-2`.
- `T = 2000` Caída de la conexión entre `B-2` y `C` y entre `B-1` y `B-2`.
- `T = 3000` Recuperación de la conexión entre `B-2` y `C` y entre `B-1` y `B-2`.
- `T = 4000` Caída de la conexión entre `B-2` y `C` y entre `B-1` y `B-2`.
- `T = 5000` Recuperación de la conexión entre `B-2` y `C` y entre `B-1` y `B-2`.
- `T = 6000` Caída de la conexión entre `B-2` y `C` y entre `B-1` y `B-2`.
- `T = 7000` Recuperación de la conexión entre `B-2` y `C` y entre `B-1` y `B-2`.

Archivos de la simulación

- `ciclar2-eventos.xml` Fichero de configuración de los eventos.
- `ciclar2-topologia.xml` Fichero de configuración de la topología.
- `trace-ciclar-2.tar.gz` Resultado de las trazas del simulador.
- `ejemplo-ciclo-2.dia` Esquema de la topología.

B.1.7. Escenario que oscila: ejemplo 3

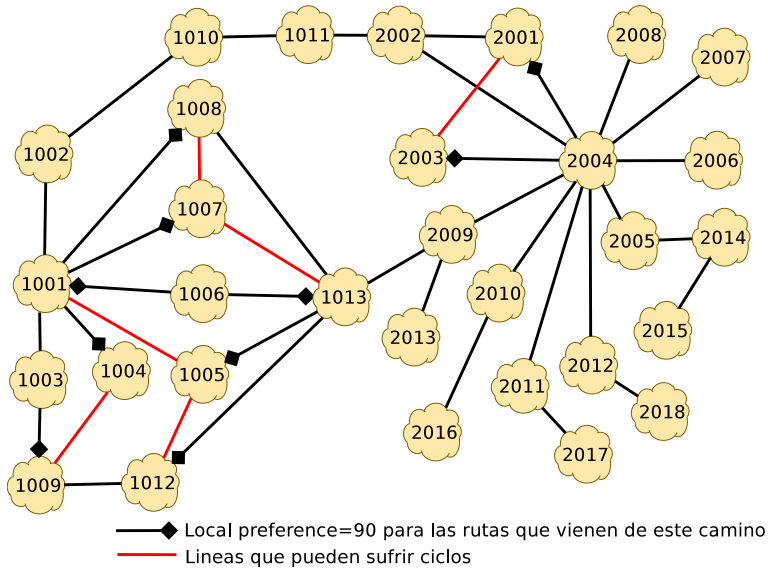


Figura B.7: Ejemplo 7: Topología de un escenario avanzado que oscila

Configuración

139 nodos repartidos en 31 AS.
Configuración específica de nodos:

- `nodo1001_6`: *Local preference*=90 para las rutas del vecino `nodo1006_1`.
- `nodo1004_1`: *Local preference*=90 para las rutas del vecino `nodo1001_4`.
- `nodo1005_2`: *Local preference*=90 para las rutas del vecino `nodo1013_8`.
- `nodo1007_1`: *Local preference*=90 para las rutas del vecino `nodo1001_7`.
- `nodo1008_1`: *Local preference*=90 para las rutas del vecino `nodo1001_8`.
- `nodo1009_1`: *Local preference*=90 para las rutas del vecino `nodo1003_3`.
- `nodo1013_6`: *Local preference*=90 para las rutas del vecino `nodo1013_3`.

Eventos temporales

Tiempo de la simulación: 10.000 segundos.

- `T = 1000` Caída de la conexión entre los AS `1007` y `1008`.

- `T = 1000` Caída de la conexión entre los AS `1007` y `1013`.
- `T = 2000` Recuperación de la conexión entre los AS `1007` y `1008`.
- `T = 2000` Recuperación de la conexión entre los AS `1007` y `1013`.
- `T = 2000` Caída de la conexión entre los AS `1001` y `1005`.
- `T = 3000` Caída de la conexión entre los AS `2001` y `2003`.
- `T = 3000` Recuperación de la conexión entre los AS `1001` y `1005`.
- `T = 4000` Caída de la conexión entre los AS `1004` y `1009`.
- `T = 4000` Recuperación de la conexión entre los AS `2001` y `2003`.
- `T = 5000` Recuperación de la conexión entre los AS `1004` y `1009`.
- `T = 7000` Caída de la conexión entre los AS `1005` y `1012`.
- `T = 7000` Caída de la conexión entre los AS `1007` y `1008`.
- `T = 8000` Recuperación de la conexión entre los AS `1005` y `1012`.
- `T = 8000` Recuperación de la conexión entre los AS `1007` y `1008`.

Archivos de la simulación

- `ciclar3-eventos.xml` Fichero de configuración de los eventos.
- `ciclar3-topologia.xml` Fichero de configuración de la topología.
- `trace-ciclar-3.tar.gz` Resultado de las trazas del simulador.
- `ejemplo-ciclo-3.dia` Esquema de la topología.

B.1.8. Escenario wedgie 3/4

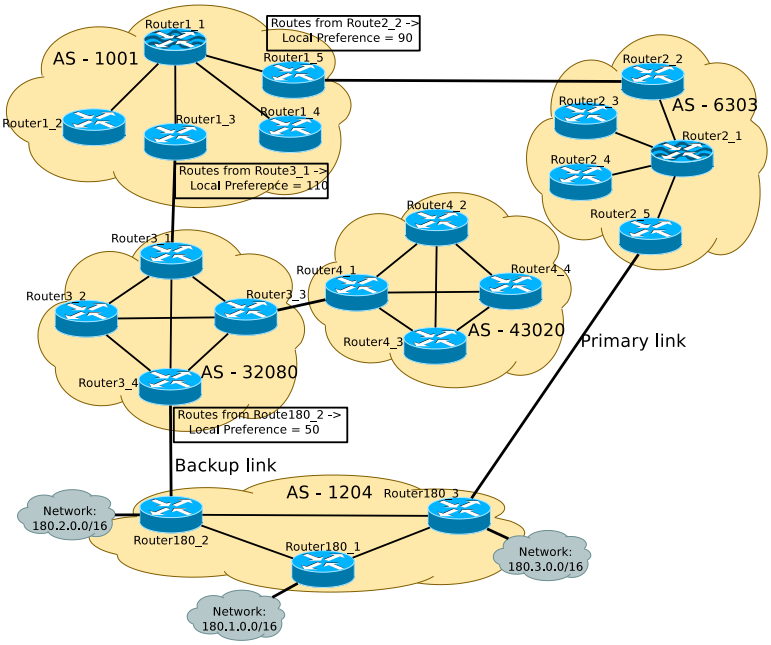


Figura B.8: Ejemplo 8: Topología de un escenario que sufre *wedgie 3/4*

Configuración

La configuración consta de 21 nodos distribuidos en 5 sistemas autónomos.

- `Router1_3`: *Local preference*=110 para las rutas del vecino `Router3_1`.
- `Router1_5`: *Local preference*=90 para las rutas del vecino `Router2_2`.
- `Router3_4`: *Local preference*=50 para las rutas del vecino `Router180_2`.

Eventos temporales

Tiempo de la simulación: 10.000 segundos

- `T = 0` Caída de la conexión entre los nodos `Router180_2` y `Router3_4`.
- `T = 500` Recuperación de la conexión entre los nodos `Router180_2` y `Router3_4`.
- `T = 2000` Caída de la conexión entre los nodos `Router180_3` y `Router2_5`.

- `T = 3000` Recuperación de la conexión entre los nodos `Router180_3` y `Router2_5`.
- `T = 5000` Caída de la conexión entre los nodos `Router180_2` y `Router3_4`.
- `T = 6000` Recuperación de la conexión entre los nodos `Router180_2` y `Router3_4`.

Archivos de la simulación

- `wedgie3_4-eventos.xml` Fichero de configuración de los eventos.
- `wedgie3_4-topologia.xml` Fichero de configuración de la topología.
- `trace-wedgie3_4.tar.gz` Resultado de las trazas del simulador.
- `wedgie3_4.dia` Esquema de la topología.

B.1.9. Escenario full wedgie

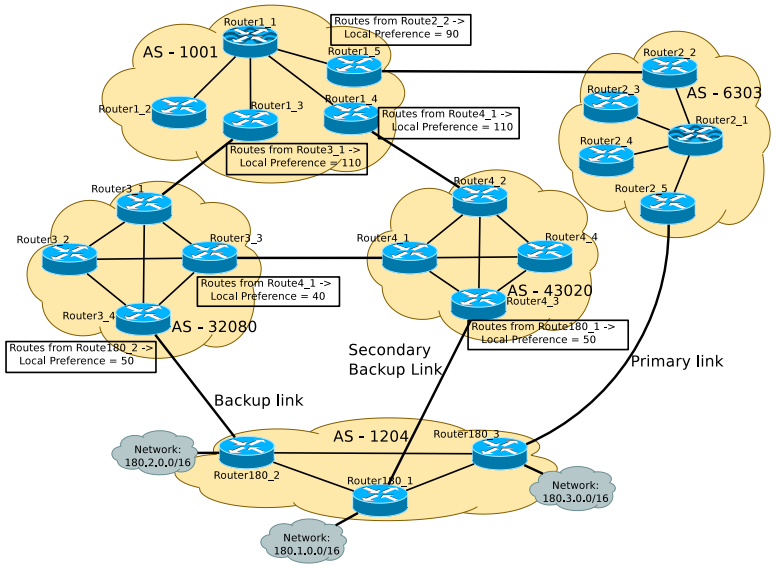


Figura B.9: Ejemplo 9: Topología de un escenario que sufre *full wedgie*

Configuración

Configuración específica de nodos:

- `Router1_3`: *Local preference*=110 para las rutas del vecino `Router3_1`.
- `Router1_3`: *Local preference*=110 para las rutas del vecino `Router4_2`.
- `Router1_5`: *Local preference*=90 para las rutas del vecino `Router2_2`.
- `Router3_3`: *Local preference*=40 para las rutas del vecino `Router4_1`.
- `Router3_4`: *Local preference*=50 para las rutas del vecino `Router180_2`.
- `Router4_3`: *Local preference*=50 para las rutas del vecino `Router180_1`.

Eventos temporales

Tiempo de la simulación: 10.000 segundos

- `T = 0` Caída de la conexión entre los nodos `Router180_2` y `Router3_4`.
- `T = 0` Caída de la conexión entre los nodos `Router180_1` y `Router4_3`.
- `T = 500` Recuperación de la conexión entre los nodos `Router180_2` y `Router3_4`.

- `T = 500` Recuperación de la conexión entre los nodos `Router180_1` y `Router4_3`.
- `T = 2000` Caída de la conexión entre los nodos `Router180_3` y `Router2_5`.
- `T = 3000` Recuperación de la conexión entre los nodos `Router180_3` y `Router2_5`.
- `T = 5000` Caída de la conexión entre los nodos `Router180_2` y `Router3_4`.
- `T = 6000` Recuperación de la conexión entre los nodos `Router180_2` y `Router3_4`.
- `T = 7000` Caída de la conexión entre los nodos `Router180_2` y `Router3_4`.
- `T = 7000` Caída de la conexión entre los nodos `Router180_1` y `Router4_3`.
- `T = 8000` Recuperación de la conexión entre los nodos `Router180_2` y `Router3_4`.
- `T = 8000` Recuperación de la conexión entre los nodos `Router180_1` y `Router4_3`.

Archivos de la simulación

- `full_wedgie-eventos.xml` Fichero de configuración de los eventos
- `full_wedgie-topologia.xml` Fichero de configuración de la topología
- `trace-full_wedgie.tar.gz` Resultado de las trazas del simulador
- `full_wedgie.dia` Esquema de la topología

Bibliografía

[1] Y. Rekhter, T. Li, and S. Hares. A Border Gateway Protocol 4 (BGP-4). RFC 4271 (Draft Standard), January 2006.

[2] Réseaux IP Européens (RIPE). `http://www.ripe.net/ripe`. Last visit: April 2010.

[3] J-sim. `http://j-sim.cs.uiuc.edu/` and `http://sites.google.com/site/jsimofficial/`. Last visit: September 2010.

[4] J-Sim Contributions. `http://sites.google.com/site/jsimofficial/3rd-party-contributions--add-ons`. Last visit: April 2010.

[5] Scalable Simulation Framework Network Models (SSFNet). `http://www.ssfnet.org/`. Last visit: December 2009.

[6] The FP7 4WARD Project). `http://http://www.4ward-project.eu/`. Last visit: July 2010.

[7] Pedro A. Aranda Gutiérrez, Petteri Pöyhönen, Luis Enrique Izaguirre and Francisco Huertas Ferrer. Using bgp-4 to migrate to a future internet. oct 2010.

[8] T. Griffin and G. Huston. BGP Wedgies. RFC 4264 (Informational), November 2005.

Lista de abreviaturas, acrónimos y definiciones

ACA Arquitectura de componentes autónomos

ASPATH AS_PATH Autonomous System Path

AS Sistema Autónomo

BGP-4 Border Gateway Protocol

BSD Berkeley Software Distribution

CSL Capa de Servicios de Red

DML Domain Modeling Language

eBGP BGP externo

EGP Protocolo de encaminamiento externo

iBGP BGP interno

IGP Protocolo de enrutamiento interno

ISP Proveedor de Internet

J-Sim JavaSim

MED Multi-Exit Discriminator

NLRI Network Layer Reachability Information

OSPF Open Shortest Path First

RIP Routing Information Protocol

RIPE Réseaux IP Européens

RIR Registro Regional de Internet

RFC Request For Comments

RR Reflector de Rutas

RRC Cliente de reflexión de rutas

Tcl Tool Command Language

Tk Tool Kit

XML Extensible Markup Language

–¿Qué te parece desto, Sancho? – Dijo Don Quijote –
Bien podrán los encantadores quitarme la ventura,
pero el esfuerzo y el ánimo, será imposible.

Segunda parte del Ingenioso Caballero
Don Quijote de la Mancha
Miguel de Cervantes

–Buena está – dijo Sancho –; fírmela vuestra merced.
–No es menester firmarla – dijo Don Quijote–,
sino solamente poner mi rúbrica.

Primera parte del Ingenioso Caballero
Don Quijote de la Mancha
Miguel de Cervantes