Detalle de requerimientos funcionales

INFORMACIÓN GENERAL

Nombre del proyecto:	Rhonda
Solicitado por	Etson Guerrero
Licencia:	MIT
Persona de contacto:	Jorge Lainfiesta
Fecha de captura	5 de mayo del 2015
Fecha de finalización	*9 de junio del 2015*

INFORMACIÓN DEL PROYECTO

Objetivo	Diseñar e implementar un router que se conecte con otros routers para intercambiar mensajes.
Descripción	El router consta de tres módulos: routing, forwarding, application. En el routing se debe implementar el algoritmo de distance vector para interactuar con la red. El forwarding se envía de enviar los mensajes a los destinos según está descrito por la tabla generada por el distance vector. La aplicación debe mostrar los mensajes recibidos y permitir que el usuario envíe mensajes.
Tecnologías a usar	Python, syslog

FACTORES DE LOGRO POR MÓDULO

Routing	- Se establecen conexiones con otros routers según una configuración de red dada. - Se aceptan conexiones de otros routers según una configuración de red dada. - Si se pierde la conexión con un router el sistema sigue operando sin cerrarse abruptamente. - El sistema calcula correctamente la tabla de direcciones utilizando el algoritmo de Distance Vector. - Si se recupera la conexión con un router el sistema vuelve a integrarlo en el funcionamiento de la red.
Forwarding	- El sistema envía los mensajes que recibe a los nodos correspondientes según la tabla de direcciones.
Aplicación	- La aplicación muestra los mensajes provenientes de otros nodos. - La aplicación permite a un usuario enviar mensajes a través de la red.

ESPECIFICACIÓN DE MÓDULOS

Módulo de Routing

Este módulo se encarga de ejecutar el algoritmo de ruteo y establecer las rutas más cortas hacia todos los destinatarios en la red. Cada uno de los enlaces entre dos routers tendrá un costo asociado.

Al iniciar, cada Router debe recibir como parámetro un listado con los nodos adyacentes y los costos entre los nodos. Estos parámetros deben ser almacenados en un archivo de configuración, el cual será interpretado cada vez que el Router sea iniciado. Este conocimiento inicial acerca de sus nodos adyacentes debe ser tomado en cuenta para que cada router comience a configurar sus tablas de ruteo en el momento de ser encendido.

El algoritmo a implementar, será un caso de Distance Vector (DV), donde se mantendrán sesiones abiertas entre los routers adyacentes y se comunicarán entre si toda la información relativa a los cálculos y actualizaciones de sus DV para todos los posibles destinos de esta red.

El protocolo estándar de comunicación entre los routers, se describe a continuación:

- Cada Router se conecta al puerto 9080 de sus adyacentes. Por lo que los Routers deben tener dos modalidades de operación, un thread cliente que está enviando sus notificaciones de DV a través del puerto 9080 de sus vecinos y un thread servidor que está escuchando en el puerto 9080 las notificaciones de DV que sus vecinos le están enviando.

- La inicialización de conexión requiere el envío de un saludo inicial con el siguiente formato:

  ```
  From:<Router que notifica>
  Type:HELLO
  ```

- El Router que recibe la solicitud de conexión debe asignar recursos para atender a su Nuevo vecino y en todo momento mantener un log que muestre cuantos routers adyacentes han iniciado sesión.

- Al aceptar la conexión, el Router que recibe la solicitud debe contestar con el siguiente mensaje:

  ```
  From:<Router que notifica>
  Type:WELCOME
  ```

- En todo momento los Routers deben garantizar dos conexiones con cada adyacente. Una conexión donde se comportan como el Servidor del módulo de ruteo donde únicamente escuchan mensajes y notificaciones provenientes de su vecino y la otra conexión donde participan como Cliente en la comunicación de ruteo, siendo a través de este segundo canal donde ocurre la comunicación desde su router hacia su vecino, informando de cambios en los Distance Vector.

- Cada Router debe tener un parámetro de configuración donde se pueda configurar el intervalo T en el cual se estarán enviando los mensajes entre los routers. Por defecto este intervalo será inicializado en 30 segundos.

- Al vencerse dicho intervalo, el Router debe enviar una notificación en base al estado de su tabla de Distance Vectors, teniendo dos posibles escenarios:

1. Se han realizado cambios en los Distance Vectors. Es decir, se ha encontrado una nueva ruta con un costo más corto para un destino determinado. En este caso se debe enviar la actualización de rutas a los routers adyacentes, siguiendo este formato:

   ```
   From:<Router que notifica>
   Type:DV
   Len:<Cantidad de Líneas>
   <Dirección Destino> :<Costo de la Ruta>
   <Dirección Destino> :<Costo de la Ruta>
   …
   <Dirección Destino> :<Costo de la Ruta>
   ```

   Por ejemplo, en el gráfico anterior, al iniciarse el Router del GrupoA debe configurar los DV hacia sus adyacentes, por lo que su primera notificación debe incluir dicha información, quedando de la siguiente manera:

   El mensaje desde el GrupoA hacia el GrupoC

   ```
   From:A
   Type:DV
   Len:2
   C;3
   B:1
   ```

   De esta forma, el router del GrupoC podrá conocer que el Router del grupo A conoce dos rutas, una hacia la dirección C con un costo de 3, y una hacia la dirección B con un costo de 1. El análisis se debe llevar a cabo de la manera vista en clase.

   Recuerde que únicamente se transmiten los Distance Vector que hayan sido actualizados o cambiados desde el último envío de notificaciones.

2. No han ocurrido cambios en los Distance Vectors. Si durante el intervalo T no han ocurrido cambios en los Distance Vectors, entonces únicamente se envía un mensaje “keep alive” con el siguiente formato:

   ```
   From:<Router que notifica>
   Type:KeepAlive
   ```

Si un Router no recibe mensajes de su adyacente por un intervalo de tiempo U, entonces el Router es libre de asignar un costo infinito a la conexión entre él y el Router que no se ha comunicado, provocando una actualización en las rutas. Este intervalo U tiene un valor por defecto de 90 segundos y debe ser parametrizable. En este proyecto, un valor infinito será representado como un número lo suficientemente grande, 99.

El módulo de ruteo también debe ser diseñado para soportar reconexiones con los router vecinos.

Forwarding

El módulo de forwarding es el encargado de analizar cada mensaje de capa de aplicación y reenviarlo por la ruta que corresponda en base al análisis de la capa de ruteo. Este módulo estará escuchando en el puerto 1981 y podrá recibir mensajes de dos fuentes:

- La aplicación que está corriendo en ese Nodo.
- Un mensaje reenviado por otro Nodo.

En cualquiera de los dos casos, esta capa debe revisar la tabla de ruteo válida en ese momento para reenviar el mensaje por la ruta que corresponda. La decisión se realiza en base a la dirección destino del mensaje a enviar.

Esta capa debe mostrar en pantalla una impresión de cada mensaje que ha pasado por este módulo, mostrando quien le ha enviado el mensaje y hacia donde fue reenviado/entregado el mensaje. Esta impresión nos permitirá realizar el “debugging” de la red.

Application

Cada Router está ejecutando una Aplicación de Mensajería, cuya única función es el envío y recepción de mensajes que probarán la red.

Esta aplicación debe permitir al usuario enviar mensajes a cualquiera de los posibles destinos de la red. El formato de los mensajes es el siguiente:

```
From:<Dirección de la aplicación origen>
To:<Dirección de la aplicación destino>
Msg:<Mensaje>
EOF
```

CALENDARIZACIÓN DE ENTREGABLES

Tipo Tarea	Tarea	Fecha Inicio	Fecha Fin	Responsable
Análisis	Especificación de Requerimientos	5/5/15	5/19/15	Todos
Análisis	Diseño (Diagramas UML)	5/5/15	5/19/15	Todos
Desarrollo	Wendy	5/5/15	5/19/15	Luis
Desarrollo	Worker	5/5/15	5/19/15	Benito
Desarrollo	Distance Vector	5/5/15	5/19/15	Benito / Luis
Desarrollo	Logger / Shell	5/5/15	5/19/15	Jorge
Desarrollo	Forwarding / Recibe solicitudes	5/19/15	5/26/15	Luis
Desarrollo	Forwarding / Procesa orden	5/19/15	5/26/15	Jorge
Desarrollo	Forwarding / Reenvio de orden	5/19/15	5/26/15	Benito
Desarrollo	App / Interfaz para envío de mensajes	5/19/15	6/2/15	Jorge
Desarrollo	App / Recepción de mensajes	5/19/15	6/2/15	Benito
Pruebas	Pruebas de Desarrollo	5/26/15	6/2/15	Todos
Pruebas	Pruebas de Integración	6/4/15	6/8/15	Todos
Entrega	Entrega Final	6/9/15	6/9/15	Todos