

6. PRUEBA 1

La primera prueba de simulación de BGP consiste en un segmento formado por tres nodos tal y como se ilustra en la figura 18. Como se puede observar, cada uno de los nodos pertenece a un Sistema Autónomo (AS) concreto, el Nodo 1 al AS 25, el Nodo 2 al AS 12 y el Nodo 3 al AS 30. Estos nodos tienen conexiones entre ellos (con identificadores 99 y 100). En esta simulación no se producirá ningún error de transmisión aunque sí se producirá la caída del Nodo 3 en el instante $t=10.0s$.

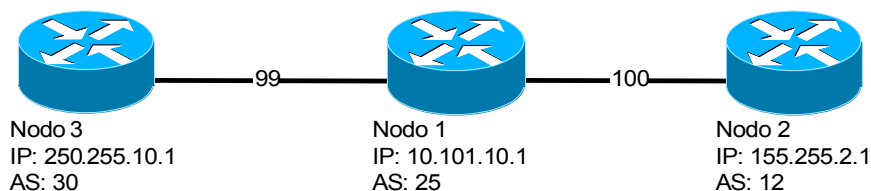


Figura19. Esquema del escenario de la Prueba 1.

Lo primero en el código de simulación será definir los parámetros de cada nodo BGP, es decir, dirección IP, identificador de AS y *Hold Timer* de cada nodo. Puesto que no habrá error de transmisión, las variables globales *error_tx* y *t_errorTx* tendrán los valores *false* y *0.0* respectivamente, indicando a los nodos BGP que no habrá error en transmisión. Una vez se llama a los constructores de los nodos BGP, el siguiente paso será generar los gestores de las conexiones para cada nodo, habrá uno por cada extremo de cada conexión, es decir, uno en el Nodo 1 para la conexión 100 (*gn1_100*) y otro para la conexión 99 (*gn1_99*), uno en el Nodo 2 para la conexión 100 (*gn2_100*) y uno en el Nodo 3 para la conexión 99 (*gn3_99*). Una vez tenemos los gestores de conexión, el siguiente paso es indicar cuándo, en la simulación, se activará cada uno de los nodos, es decir, a partir de que instante estarán listos para enviar y recibir mensajes, este instante será *0.0s* para todos los nodos. El siguiente paso es hacer efectivas en los nodos las conexiones, con esto conseguimos que cada nodo acepte y envíe mensajes a los nodos indicados y sólo a estos, para ello usamos el operador ">>" heredado de la clase *Module*, que se usa de la forma *ni>>nj* y que hace que el Nodo *i* tenga al Nodo *j* como receptor de mensajes y, por tanto, el Nodo *j* tenga al Nodo *i* como transmisor de mensajes; además se usa el método *BGPnode::link(idcon)* para que cada nodo BGP interprete la conexión identificada por *idcon* como propia y acepte mensajes de ésta. Una vez tenemos configurado el conexionado de los nodos BGP, el siguiente punto es indicar el instante en que se establecerá cada una de las conexiones, para ello se emplea el método *gestConn::init(t_ini)*, pero se hará sólo para uno de los gestores extremos de cada conexión, en este caso se hace para el gestor del Nodo 2 para la conexión 100 en el instante *1.0s* y para el gestor del Nodo 3 para la conexión 99 en el instante *2.0s*. Finalmente se indica el instante en que cada nodo quedará inactivo, si no queremos que un nodo simule una caída el instante de desactivación será el instante en que finaliza la simulación. Para esto se emplea el método *BGPnode::stop(t_fin)*, y puesto que la simulación será de *20.0s*, los nodos 1 y 2 finalizarán en dicho instante mientras que el Nodo 3, que emula una caída, lo finalizará en el instante *10.0s*. El último paso será arrancar el simulador, más concretamente el manejador de eventos (*Scheduler simInst*) de la librería de simulación (*simul*), mediante la llamada al método *Scheduler.run(...)* pasándole como parámetro el instante de finalización de la simulación. Indicar, que al tratarse de una simulación orientada a eventos, la simulación no dura 20 segundos reales, sino que se

generarán eventos (envío y recepción de mensajes) hasta el instante 20 en una cola de eventos para el manejador de eventos, y éste los activará uno tras otro. En realidad el manejador de eventos, y por tanto la simulación, durará un tiempo ligeramente superior a los 20 segundos, esto se hace así para permitir que al finalizar los nodos, se genere el último fichero contenedor de RIBs para cada nodo del sistema, que será en $t=20s$ para los nodos que permanecen toda la simulación. Si el manejador de eventos terminase en $t=20s$, no se obtendrían dichos ficheros.

Una vez descrito el esquema que se va a probar y la configuración de éste para que se comporte como se desea, vamos a estudiar su evolución. Para ello estudiaremos las distintas visiones que va teniendo cada uno de los nodos del sistema a través de sus Bases de Información de Enrutamiento (RIBs) y más concretamente de su RIB Local (Loc_RIB).

6.1. Nodo 1

La primera visión que tendrá este nodo será una vez queda establecida la conexión con el Nodo 2, es decir, tras el ya mencionado intercambio de mensajes OPEN y KEEPALIVE y tras haber procesado el primer mensaje UPDATE que le envía el Nodo 2; esta visión se muestra en la siguiente figura.

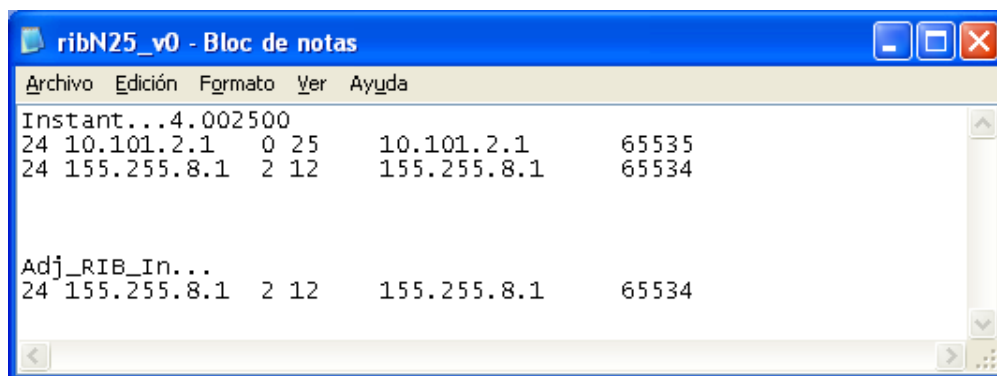


Figura 20. Prueba 1. RIBv0 Nodo1.

La siguiente se producirá una vez haya establecido conexión con el Nodo 3 y haber procesado el primer mensaje UPDATE. La podemos observar en la siguiente figura.

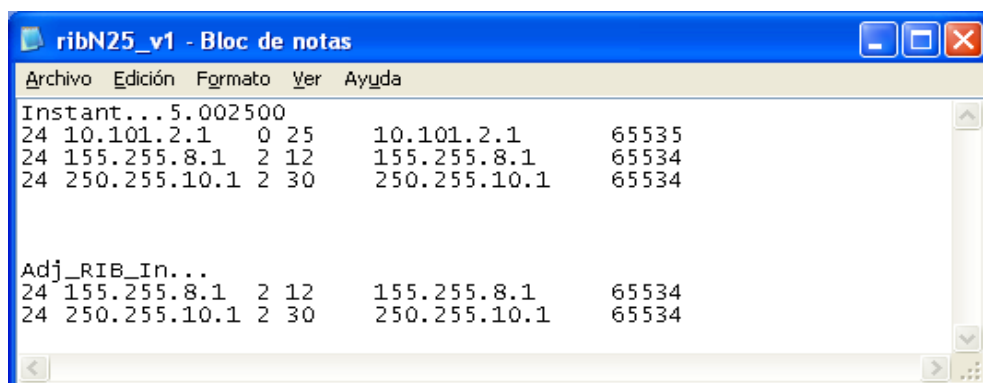


Figura 21. Prueba 1. RIBv1 Nodo1.

La siguiente y última “foto” del sistema se produce al detectar la caída del Nodo 3 al vencer el *Hold Timer* de la conexión con dicho nodo.

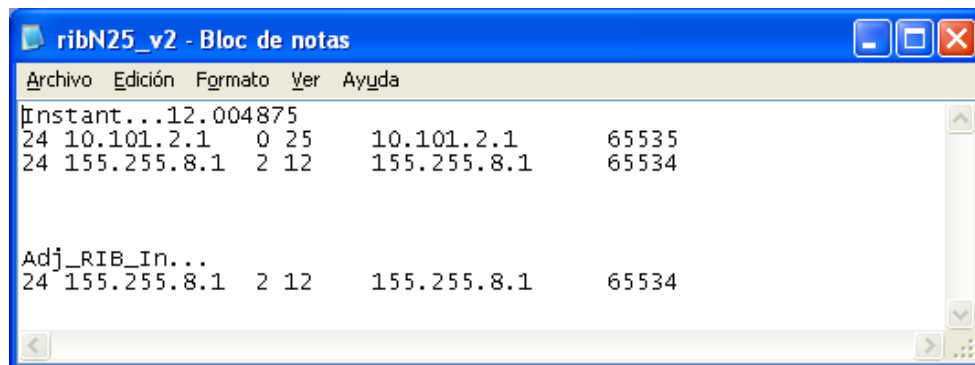


Figura 22. Prueba 1. RIBv2 Nodo1.

Así pues, la evolución del sistema vista por el Nodo 1 es la siguiente.

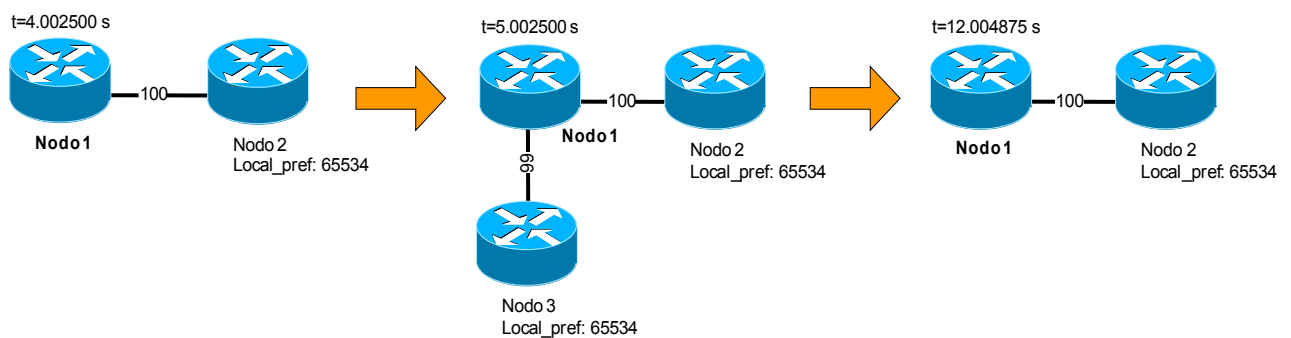
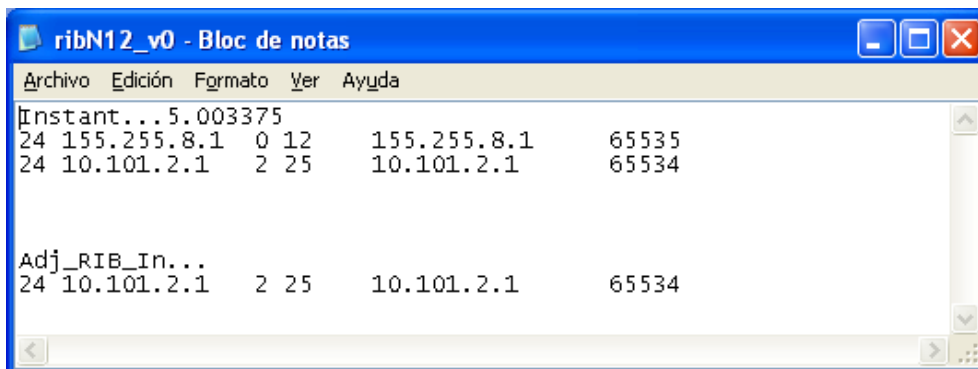


Figura 23. Prueba 1. Evolución según el Nodo1.

Como se puede ver, inicialmente el Nodo 1 sólo tiene conocimiento del Nodo 2, lo cual es lógico, ya que ésta es la primera conexión en activarse (la 100), posteriormente se establece conexión con el Nodo 3 y cambia la visión, el Nodo 1 tiene conocimiento de los nodos 2 y 3. Tras la caída del Nodo 3 vence el *Hold Timer* de la conexión 99 para el Nodo 1, con lo que se activa el proceso de supresión de rutas y el Nodo 1 vuelve a ver únicamente al Nodo 2.

6.2. Nodo 2

La primera visión del sistema para el Nodo 2 se producirá tras establecerse la conexión con el Nodo 1 y recibir el primer mensaje UPDATE.



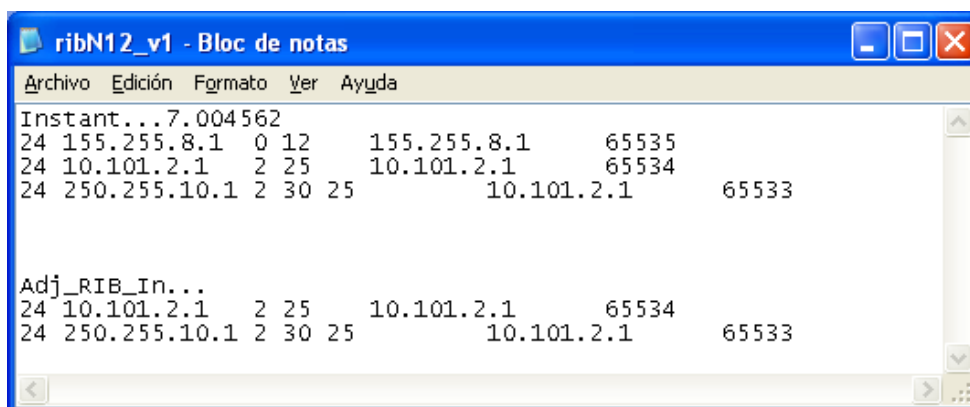
```

ribN12_v0 - Bloc de notas
Archivo Edición Formato Ver Ayuda
Instant...5.003375
24 155.255.8.1 0 12 155.255.8.1 65535
24 10.101.2.1 2 25 10.101.2.1 65534

Adj_RIB_In...
24 10.101.2.1 2 25 10.101.2.1 65534
  
```

Figura 24. Prueba 1. RIBv0 Nodo2.

La visión del sistema evolucionará tras conocer el Nodo 1 la existencia del Nodo 3, al actualizar éste su Loc_RIB anunciará a su vecino (el Nodo 2) la existencia del Nodo 3 y la alcanzabilidad de éste a través de si mismo con un mensaje UPDATE.



```

ribN12_v1 - Bloc de notas
Archivo Edición Formato Ver Ayuda
Instant...7.004562
24 155.255.8.1 0 12 155.255.8.1 65535
24 10.101.2.1 2 25 10.101.2.1 65534
24 250.255.10.1 2 30 25 10.101.2.1 65533

Adj_RIB_In...
24 10.101.2.1 2 25 10.101.2.1 65534
24 250.255.10.1 2 30 25 10.101.2.1 65533
  
```

Figura 25. Prueba 1. RIBv1 Nodo2.

Finalmente, al detectar el Nodo 1 la caída del Nodo 3 y actualizar su Loc_RIB, anunciará al Nodo 2 que la ruta hacia el Nodo 3 deja de ser válida, con lo que la visión del Nodo 2 será la siguiente.

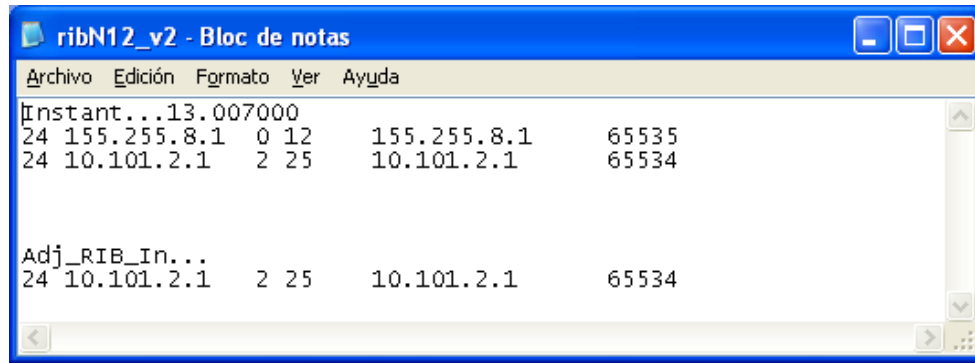


Figura 26. Prueba 1. RIBv2 Nodo2.

Esta última visión del sistema será la definitiva para el Nodo 2. Según lo anterior, la evolución de la visión del sistema según el Nodo 2 será la siguiente.

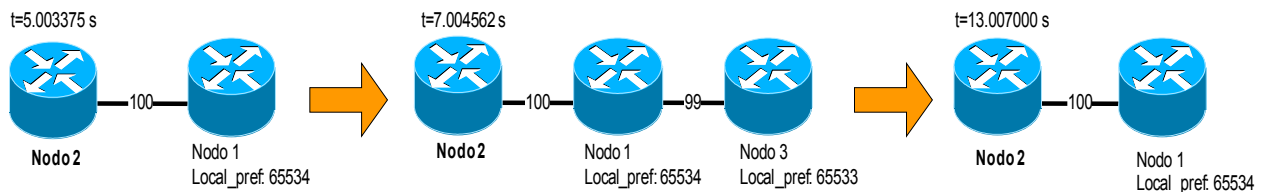


Figura 27. Prueba 1. Evolución según el Nodo2.

6.3. Nodo 3

La visión del sistema según el Nodo 3 comenzará tras establecer conexión con el Nodo 1 y procesar el primer mensaje UPDATE recibido. Puesto que es el Nodo 3 el que inicia la conexión con el Nodo 1, es lógico que su conocimiento del Nodo 1 sea posterior al del Nodo 1 sobre el Nodo 3, pues el Nodo 1 recibe antes el mensaje UPDATE.

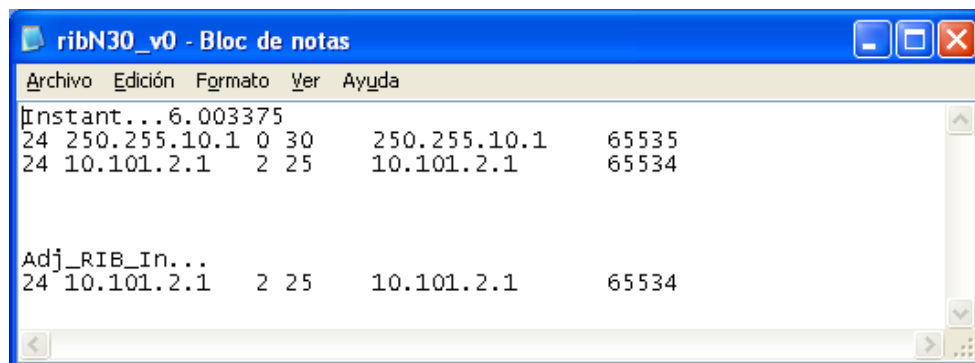


Figura 28. Prueba 1. RIBv0 Nodo3.

La Loc_RIB del Nodo 3 evolucionará al recibir el mensaje UPDATE del Nodo 1 con la indicación de la ruta hasta el Nodo 2. Esta indicación es bastante próxima (concretamente es el siguiente mensaje que envía el Nodo 1 al Nodo 3) al conocimiento del Nodo 1 por parte del Nodo 3.

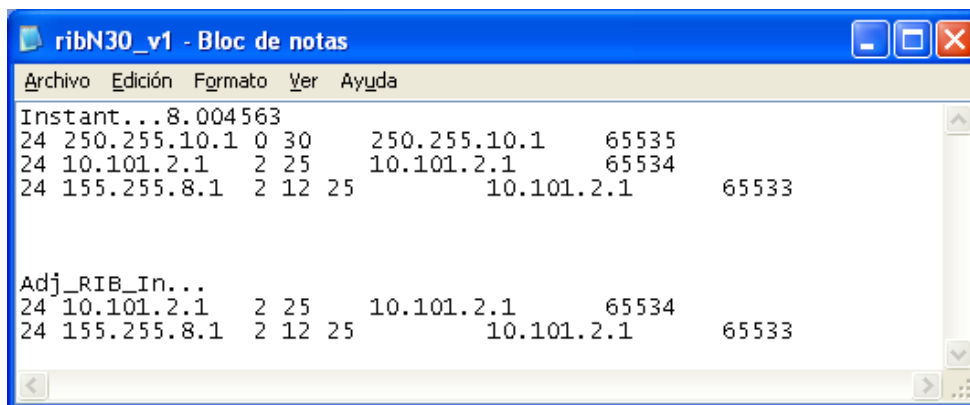


Figura 29. Prueba 1. RIBv1 Nodo3.

Al caer el Nodo 3 en el instante 10.0s, la anterior será la última visión que tenga del sistema. Así, la evolución del sistema según el Nodo 3 es la siguiente.

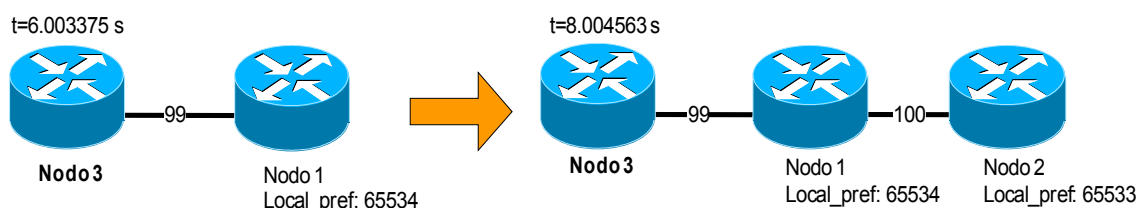


Figura 30. Prueba 1. Evolución según el Nodo3.

6.4. Hold Timer

Respecto al *Hold Timer* de las conexiones, la configuración inicial de los nodos BGP hace que, por defecto, para el Nodo 1 tenga un valor de tres segundos, para el Nodo 2 cuatro segundos y para el Nodo 3 cinco segundos. Estos valores serán los que se incluyan los mensajes OPEN que cada uno de los nodos enviará a su respectivo vecino para iniciar la conexión BGP. Teniendo en cuenta que el Nodo 1 conectará con los otros dos y que su valor del *Hold Timer* es el menor de todos, las dos conexiones tendrán el valor del *Hold Timer* fijado a tres segundos (el menor de los propuestos en cada uno de los dos casos).

6.5. Conclusiones

Tras esta primera prueba, podemos ver cómo cada uno de los nodos se comporta según se espera, es decir, se sigue correctamente la evolución de la máquina de estados y por tanto las conexiones se establecen correctamente, tras lo cual, la visión del sistema por parte de cada uno de los nodos que lo conforman a través de su Loc_RIB va evolucionando correctamente a la vez que se van propagando las rutas preferidas por cada uno de ellos (el Nodo 2 tiene conocimiento del Nodo 3 gracias al Nodo 1 y viceversa). Además, hemos podido comprobar el correcto funcionamiento del proceso de eliminación de rutas al provocar el vencimiento del *Hold Timer* (al caer el Nodo 3, no responde a los mensajes del Nodo 1 y en este último vence el mencionado temporizador) y cómo la eliminación de rutas se propaga al resto de nodos del sistema (concretamente del Nodo 1 al Nodo 2 al caer el Nodo 3).

Finalmente, al ser esta la primera prueba veremos cómo es la secuencia de mensajes que lleva al establecimiento de una conexión:

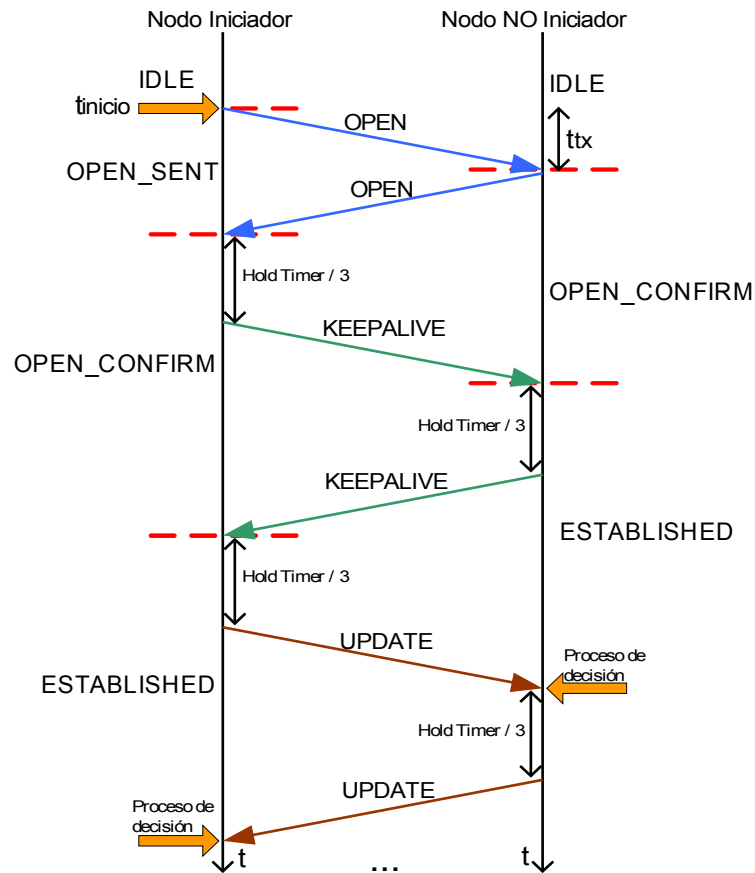


Figura 31. Establecimiento de una conexión.