2012-03

Mejoras al aprovechamiento de dispositivos multiconectados

Cortés Martín, Alberto

Universidad Carlos III de Madrid

TESIS DOCTORAL

Mejoras al aprovechamiento de dispositivos multiconectados

Autor: **Alberto Cortés Martín**
Ingeniero de Telecomunicación

Director: **Carlos García Rubio**
Dr. Ingeniero de Telecomunicación

Escuela Politécnica Superior
Departamento de Ingeniería Telemática

Leganés, 16 de marzo de 2012

Mejoras al aprovechamiento de dispositivos multiconectados

Autor: Alberto Cortés Martín
Director: Prof. Dr. Carlos García Rubio

Tribunal nombrado por el Mgfco. y Excmo. Sr. Rector de la Universidad Carlos III de Madrid, el día __ de _____ de _____.

El tribunal calificador: Firma:

Presidente:

Vocal:

Vocal:

Vocal:

Secretario:

Calificación:

Leganés, __ de _____ de _____.

Resumen

En esta tesis abordamos la problemática de cómo aprovechar las múltiples interfaces de red disponibles en los dispositivos de usuario modernos, de forma que se puedan traspasar los flujos de datos a las interfaces más apropiadas en cada momento.

Los principales problemas que hemos identificado y que hacen que sea difícil sacarle partido a los dispositivos de usuario con varias interfaces de red son:

- Las dificultades que presentan algunos protocolos de transporte para sobrevivir a un cambio de la interfaz de red.
- Las limitaciones de los sistemas operativos para facilitar la identificación de las interfaces más apropiadas para cada comunicación, en cada momento.

En cuanto a los problemas técnicos que presentan los protocolos de comunicaciones, pensamos que SCTP puede solucionar muchos de ellos, principalmente porque está diseñado, desde su concepción, para dispositivos con varias interfaces de red (es decir, para dispositivos *multiconectados*).

Sin embargo, el tiempo que tarda SCTP en detectar fallos de ruta por una interfaz y pasar a utilizar una de las alternativas es excesivamente largo para algunas aplicaciones.

Por eso presentamos una modificación a SCTP que permite ajustar la duración de la detección del fallo de ruta a las necesidades de las aplicaciones, de forma que la velocidad con la que el protocolo reacciona ante fallos de ruta deje de ser un problema. Esto mejora las prestaciones de SCTP como solución de transporte en dispositivos multiconectados.

Nuestra propuesta se basa en el envío de sondas activas a las rutas sospechosas de sufrir un fallo y abre la posibilidad de utilizar diferentes estrategias de planificación temporal de estas sondas.

Hemos basado la elección de la planificación temporal de las sondas en un estudio matemático de las propiedades de las planificaciones temporales, en concreto, hemos elegido la planificación con el menor tiempo medio de detección de fallos de ruta temporales, porque diferenciar rápidamente entre fallos de ruta temporales y permanentes permite mejorar las prestaciones del protocolo, sobre todo en entornos inalámbricos.

Pero los fallos de transmisión en la ruta no son el único criterio por el que se debería realizar un cambio de interfaz de las comunicaciones, existen otros, en su mayoría externos al propio protocolo que también se deberían considerar.

Hemos recopilado una lista, que pensamos completa, de los posibles criterios de selección de interfaces de red, con el objeto de complementar a nuestras modificaciones de SCTP en la correcta elección de la interfaz.

También hemos implementado un nuevo servicio del sistema operativo para publicar y consultar algunos de estos criterios, de forma que los demonios que recopilan información sobre estos criterios puedan desacoplarse de las aplicaciones que se benefician de estos datos, gracias a una interfaz estándar de comunicaciones entre ambos.

Hemos puesto en práctica nuestras contribuciones en varios proyectos de investigación, lo que nos ha demostrado que se complementan adecuadamente y permiten diseñar soluciones viables para los escenarios típicos que se plantean actualmente para el mejor aprovechamiento de dispositivos multiconectados.

A Emilia, Sara
y Sebastián.

Agradecimientos

Para la realización de esta tesis he contado con la inestimable ayuda de multitud de personas. Cada uno de ellos ha contribuido de diferentes formas: compartiendo sus extensos conocimientos técnicos, su amplia experiencia, manteniendo discusiones acaloradas sobre los más diversos temas, prestándome ayuda y apoyo dentro y fuera de lo estrictamente relacionado con mis investigaciones, haciéndome reflexionar sobre sus interesantes visiones del mundo, aguantando mis humores y ausencias o simplemente sabiendo distraerme cuando distracción era lo que más necesitaba. Algunas personas han sido capaces incluso de asumir varios de estos roles al mismo tiempo.

Por concretar más y en estricto orden alfabético: Andrés Domíngez, Andrés Marín, Carlos García, Celeste Campo, Dani Díaz, Esther, Eugenio, Gloria, Gorka, Juanjo, Kike, Luis de la Fuente, los miembros de Murlocs from Hell y Pablo Yavarone.

Para todos ellos mi más profundo agradecimiento.

Índice general

Índice de figuras

Índice de cuadros

Índice de listados

Parte I

Introducción y estado del arte

Capítulo 1

Introducción

1.1 Motivación

Hoy en día es habitual disponer de dispositivos finales de usuario equipados con múltiples interfaces de red, como portátiles y teléfonos móviles modernos, que suelen venir equipados con acceso a redes Wi-Fi, Ethernet, GPRS y UMTS.

Como consecuencia de ello, o quizá motivo de ello, es que empieza a ser fácil disponer de varios tipos de conexiones de red en todo momento, por ejemplo, en nuestro lugar de trabajo, rara es la ocasión en que no tenemos cobertura GPRS, Wi-Fi y Ethernet, simultáneamente.

La abundancia de redes de acceso disponibles se ha convertido en un nuevo recurso, que el sistema operativo debe gestionar si queremos liberar al usuario de lidiar con estos detalles por sí mismo.

Históricamente, los sistemas operativos han venido soportando el utilizar varias interfaces de red sin problemas. A pesar de ello, hoy en día sigue siendo difícil aprovecharse *inteligentemente* de esta multiconectividad. Por ejemplo, intente pedirle a su sistema operativo que encamine una aplicación P2P por una red pública y barata mientras mantiene una conversación de VoIP por la red corporativa de su empresa.

Incluso si usted, como usuario avanzado del sistema, consigue forzar un comportamiento como el propuesto, será mediante una solución relativamente estática. Intente solucionar esto mismo en un entorno un poco más dinámico, por ejemplo, pruebe a levantarse de su puesto de trabajo, salir del edificio y entrar en algún medio de transporte público, ¿qué interfaces debería utilizar ahora para sus aplicaciones de P2P y VoIP?

Durante su viaje en transporte público, su portátil acabará quedándose sin batería, cuando esto empiece a ser un problema, ¿no deberían reconducirse sus flujos de datos por las interfaces que consuman menos batería? y, en ese caso, ¿deberían reconducirse todos los flujos o sólo aquellos poco críticos y con bajas restricciones de calidad de servicio?

Éstos son sólo algunos ejemplos del tipo de gestión inteligente de las diversas

interfaces de red de las que disponemos hoy en día en nuestros dispositivos. Incluso los usuarios menos formados pueden imaginar escenarios similares e igualmente difíciles de solucionar.

Los retos a los que nos enfrentamos para llevar a cabo esta tarea son diversos: cambiar de interfaz de red implica cambiar de dirección IP, por lo que tenemos en primer lugar un problema de *movilidad*, cuya solución, típicamente, está ligada a los protocolos de comunicaciones involucrados.

La gestión automática y desatendida de esta movilidad implica que el sistema operativo debe ser consciente de los diferentes criterios para seleccionar una interfaz u otra, y debe ser capaz de medir y presentar datos y figuras de mérito sobre estos criterios, de forma que algún tipo de software de toma de decisiones elija la interfaz apropiada en cada momento para cada flujo de datos.

En esta tesis no pretendemos estudiar ni plantear las políticas de decisión concretas para seleccionar qué interfaz de red utilizar en cada caso, sino solucionar algunos de los problemas técnicos que ahora mismo hacen difícil llevar a la práctica estas hipotéticas políticas de decisión.

1.2 Historia de la tesis

A mediados de 2004 comenzamos a trabajar en el proyecto "Easy Wireless" ([1]), el cual abreviamos, familiarmente, como EW.

El proyecto EW estaba orientado a facilitar que los dispositivos móviles de usuario realizaran traspasos imperceptibles entre diferentes tecnologías de acceso, intentando sacarle el mayor partido posible a su multiconectividad.

Después de un par de años familiarizándonos con los problemas relacionados con la multiconectividad de dispositivos finales de usuario, estábamos en condiciones de plantear soluciones a diferentes niveles, que se complementaban para mejorar, de forma global, el aprovechamiento de la multiconectividad de dispositivos de usuario.

Fue entonces cuando resultó evidente realizar una tesis doctoral orientada a este tema particular.

Desde entonces hemos estado trabajando en ello, a lo largo del proyecto EW y posteriormente, durante el proyecto EW2 ([2]) cuya temática era continuación del anterior.

1.3 Objetivos

El objetivo de esta tesis es avanzar en la solución de los problemas relacionados con el mejor aprovechamiento de la multiplicidad de interfaces de red presentes en los dispositivos de usuario modernos.

Las diferentes contribuciones que presentamos en esta tesis están encaminadas a conseguir que un dispositivo multiconectado, manejado por un usuario sin conocimientos técnicos, pueda hacer un uso inteligente y desatendido de las diversas interfaces de red con las que cuenta su dispositivo.

En concreto, el tipo de problemas a los que proponemos soluciones en esta tesis son los siguientes:

- La dificultad de elegir automáticamente la interfaz más adecuada de entre las disponibles en cada momento.
- Una vez elegida, cómo realizar el traspaso de las comunicaciones a la interfaz en cuestión con el menor impacto posible.

El primero de los problemas es bastante amplio en su planteamiento; la elección de la interfaz adecuada depende de muchos factores: técnicos, pecuniarios, de ahorro de energía, estratégicos... y puede involucrar a diferentes actores, como el propio usuario y sus preferencias, el sistema operativo (a través de su conocimiento sobre los detalles técnicos de las interfaces), las aplicaciones y sus necesidades de comunicación y los operadores de las diferentes redes de acceso (en función del conocimiento interno del tráfico cursado por sus redes).

En el contexto del proyecto EW, teníamos claro que la responsabilidad de elegir la interfaz a utilizar debía recaer en el dispositivo, no en la red, de forma que la elección de la interfaz pudiera llevarse a cabo aún en ausencia de recomendaciones del operador de la red de acceso.

Tampoco resultaba adecuado cargar al usuario con la responsabilidad de la decisión, que en nuestra opinión no debería preocuparse de estos detalles en exceso (la computación ubicua es una de las especialidades de nuestro grupo de investigación).

Esto nos lleva directamente a orientar la solución del problema dentro del ámbito del sistema operativo del dispositivo de usuario; nuestro propósito es dotar al sistema operativo de las herramientas necesarias para tomar la decisión de qué interfaz de red utilizar para cada comunicación, teniendo en cuenta diferentes criterios, como las preferencias de los usuarios, las características técnicas de los interfaces, la calidad de servicio percibida por la aplicación y sus necesidades...

En este sentido nos fijamos dos objetivos principales:

O1. La identificación de todos los posibles criterios a tener en cuenta a la hora de elegir que interfaz de red utilizar para cada comunicación.

O2. El desarrollo de un nuevo servicio en el sistema operativo que permita elegir la interfaz más apropiada para cada comunicación teniendo en cuenta esos criterios.

El segundo de los problemas, sobre como minimizar el impacto de un traspaso de los flujos de red a una nueva interfaz, es un problema puramente técnico, y muy condicionado por la tecnología de movilidad utilizada para el traspaso y sus limitaciones intrínsecas.

Desde el primer momento decidimos utilizar SCTP como protocolo de transporte y como solución de movilidad. En concreto, SCTP, por si mismo, resuelve con bastante éxito el problema de la interrupción de las comunicaciones al producirse un cambio de interfaz bajo demanda.

Sin embargo SCTP presenta algunos otros problemas cuando el cambio de interfaz debe realizarse sin intervención externa, por ejemplo, ante un fallo de ruta. El protocolo de transporte resulta el más apropiado para detectar los fallos de ruta lo antes posible, ya que mantiene una comunicación directa con el otro extremo, por lo tanto resulta idóneo que sea el propio SCTP el que decida realizar un cambio de interfaz en esas circunstancias.

Sin embargo SCTP es especialmente lento en detectar un fallo de ruta, por lo menos para algunas aplicaciones de tipo interactivo con el usuario.

Para ser precisos, el problema no es tanto que SCTP resulte lento en su reacción ante fallos de ruta, si no que la rapidez de su reacción no puede ser configurada por la aplicación. Cada aplicación tendrá diferentes necesidades y requisitos en cuanto a la velocidad de reacción de su protocolo de transporte ante fallos de ruta y en SCTP, esto no puede configurarse.

Por otro lado, SCTP demora innecesariamente la transmisión de datos ante fallos de ruta temporales. Un fallo de ruta temporal es aquel que por su corta duración, no es deseable un cambio de interfaz, si no simplemente esperar a restituir las comunicaciones por la interfaz original, cuando el fallo desaparece.

Este tipo de fallos son habituales en redes inalámbricas, donde por pérdidas temporales de la cobertura, se producen interrupciones intermitentes en la transmisión de los datos.

Estos razonamientos nos han llevado a fijarnos dos nuevos objetivos en la realización de esta tesis:

O3. Modificar el protocolo SCTP para que la duración de la detección del fallo de ruta sea configurable por la aplicación.

O4. Modificar el protocolo SCTP para disminuir el impacto de los fallos de ruta temporales en la transmisión de los datos.

Creemos que con la combinación de un servicio en el sistema operativo para facilitar la correcta elección de la interfaz de red, la flexibilidad de poder ajustar la duración de la detección del fallo de ruta de SCTP y una disminución del impacto de los fallos de ruta temporales, estamos contribuyendo significativamente a aprovechar inteligentemente la multiplicidad de interfaces con la que vienen equipados los dispositivos de usuario modernos.

1.4 Estructura del documento

Hemos dividido este documento en cuatro partes:

- **La primera parte** consiste en esta introducción (capítulo 1), y en una revisión del estado del arte de las tecnologías más relacionadas con nuestras contribuciones (capítulo 2), que incluye una explicación de qué son los dispositivos multiconectados, la movilidad a nivel de protocolo y la movilidad a nivel de sistema operativo.

- **La segunda parte** contiene las contribuciones de esta tesis derivadas de nuestro trabajo con SCTP.

  Empieza con el capítulo 3, que consiste en una descripción detallada del soporte de movilidad en SCTP, necesaria para entender el siguiente capítulo y donde también identificamos los problemas que solucionamos.

  El capítulo 4 detalla nuestra propuesta de modificación de SCTP para hacer el protocolo más reactivo a los fallos de ruta. Incluimos los razonamientos que nos han llevado a plantear nuestras propuestas, simulaciones sobre su comportamiento y un análisis de la estabilidad del protocolo tras nuestros cambios.

  El capítulo 5 describe nuestro estudio de los fundamentos matemáticos para el análisis de estrategias de sondeo, así como su puesta en práctica con dos estrategias particularmente importantes para el caso de SCTP. También incluimos la demostración de cuál es la mejor estrategia posible para detectar sucesos uniformes en el menor tiempo posible y el estudio de como afecta una variación aleatoria a las propiedades de las estrategias de sondeo, ya que ambos resultan de utilidad directa en nuestra modificación de SCTP.

- **La tercera parte** contiene las contribuciones de esta tesis relacionadas con modificaciones al sistema operativo para mejorar el soporte de dispositivos multiconectados.

  El capítulo 6 incluye una lista de los criterios a tener en cuenta a la hora de seleccionar qué interfaz de red resulta más apropiada para cada comunicación.

  El capítulo 7 contiene la descripción y funcionamiento de un nuevo servicio para el sistema operativo que permite seleccionar la interfaz de red apropiada en dispositivos multiconectados. También incluye los detalles de nuestra implementación de referencia del servicio y lo aprendido al utilizarlo en combinación con las otras contribuciones de esta tesis, tanto por nosotros como por terceras personas.

  Esto último también sirve de resumen de lo aprendido en la puesta en práctica, en conjunto, de todas nuestras propuestas.

- **La cuarta parte** contiene nuestras conclusiones y vías de investigación futuras. En el apéndice A incluimos una descripción formal de nuestra propuesta de modificación de SCTP.

1.5 Convenciones tipográficas y de estilo

Aunque este documento ha sido íntegramente escrito por mi, Alberto Cortés, me referiré a mi trabajo y contribuciones en plural, utilizando fórmulas del estilo "*hemos* demostrado tal y cual cosa".

La razón principal para esto es que mi trabajo ha sido realizado siempre en colaboración de mis colegas de la universidad, cuyos méritos individuales pueden atribuirse atendiendo a la lista de autores de las publicaciones que respaldan esta tesis.

Las únicas secciones en que me referiré a mí en singular son en estos mismos párrafos y en los agradecimientos.

Como este documento está escrito en castellano, he traducido los términos ingleses que a diario utilizo en mis investigaciones, excepto en los casos en que su traducción me resulta forzada o dificulta entender el texto. En estos casos, mantengo la ortografía inglesa y utilizo un tipo de letra sin remates, ejemplo: router.

Para evitar confusiones a lectores de diferente formación académica, he evitado utilizar la palabra inglesa byte y en su lugar utilizo la voz castellana "octeto", que no deja lugar a dudas sobre la cantidad de bits que representa.

Utilizaré el símbolo tipográfico § para referirme a una sección o capítulo en particular, por ejemplo, esta es la §1.5, que debe leerse como "esta es la sección 5 del capítulo 1".

Para enfatizar un texto y para algunos ejemplos largos utilizo *el tipo de letra itálica*. Utilizo las **negritas** para resaltar visualmente que contenido estoy describiendo en un párrafo cuando en la misma página menciono varios temas.

Los ejemplos de interacción con el intérprete de comandos los indico como una figura rodeada de un fino marco negro y con números de línea en el margen izquierdo. El pie de figura se encabeza con las palabras "Listado". El prompt que he utilizado es ";" y aparece precediendo las entradas de comandos para distinguirlas de su salida. Ejemplo:

```
1 ; whoami
2 alcortes
```

Listado 1.1: Ejemplo de interacción con el intérprete de comandos

Donde sea apropiado ignorar parte de la respuesta a un comando, utilizaré el símbolo "`[...]`" para indicarlo.

Los nombres de comandos, funciones de programación y parámetros de los protocolos los muestro en un `tipo de letra monoespaciado` que simula el utilizado en la mayoría de terminales de texto. Para comandos o funciones que podemos encontrar en un sistema operativo tipo UNIX, el comando o función vendrá seguido de un número entre paréntesis, que indica la página de manual

donde se documenta. Por ejemplo `ifconfig(8)`. Aunque esto sólo lo haré las primeras veces que lo mencione.

Para la escritura matemática he seguido las instrucciones de [3] y me he inspirado en [4].

En §B incluyo un listado de las siglas que utilizo, acompañadas de su significado.

Tanto para el desarrollo de mis investigaciones como para la edición de este documento he utilizado exclusivamente software libre, excepto para generar el listado al final de §2.3.1 sobre como mostrar las interfaces de red en un sistema operativo Windows XP.

Capítulo 2

Estado del arte

2.1 Dispositivos multiconectados

2.1.1 Definición e historia

Un dispositivo multiconectado es un dispositivo con varias interfaces de red.

Históricamente, sólo unos pocos tipos de dispositivos han tenido varias interfaces de red, principalmente los routers, que por definición son dispositivos multiconectados.

Otros ejemplos de dispositivos multiconectados típicos han sido ordenadores fijos, de alta potencia, generalmente actuando como servidores, a los que se les dotaba de varias interfaces de red para que pudieran ser accesibles por rutas redundantes.

Es importante aclarar que, con frecuencia, estos dispositivos de alta disponibilidad, utilizan una técnica de multiconectividad *diferente*, comúnmente conocida como site multihoming ([5, 6]), consistente en dotar de multiconectividad a los dispositivos de una red sin necesidad de equiparles con varias interfaces, sino conectando esa red al exterior mediante varios proveedores de servicio.

Aunque tanto los dispositivos multiconectados como el uso de site multihoming comparte ciertos problemas y soluciones, los aspectos más técnicos de ambas metodologías difieren notablemente, ya que la problemática en dispositivos multiconectados admite soluciones extremo a extremo mientras que las soluciones al site multihoming tienen una fuerte componente de encaminamiento entre redes ([7]). En esta tesis nos ocupamos exclusivamente de aspectos de multiconectividad aplicables a dispositivos multiconectados, no a temas relacionados con site multihoming.

Desde aproximadamente el 2003, el escenario de dispositivos multiconectados ha cambiado notablemente: es habitual encontrar en el mercado dispositivos *de usuario* multiconectados, como portátiles y ordenadores de sobremesa y mas recientemente también teléfonos móviles ([8]).

Esto es debido principalmente al abaratamiento del hardware de red, la reducción de su tamaño y la amplia disponibilidad de redes de acceso inalámbricas

como Wi-Fi, GPRS o UMTS.

Otra manera de explicar este mismo fenómeno es por el éxito que están teniendo, desde hace aproximadamente 5 años, los dispositivos portátiles; este tipo de dispositivos pueden utilizarse en cualquier sitio o mientras uno se desplaza, por lo que no es razonable dotarles únicamente de una conexión cableada, tipo Ethernet por ejemplo, ya que se limitaría mucho su utilidad. Esto ha provocado que la cobertura Wi-Fi haya crecido notablemente estos últimos años, lo que a su vez ha hecho viable vender teléfonos móviles con interfaz Wi-Fi además de GPRS o UMTS.

Razones aparte, el hecho es que hoy en día existe un parque extenso de dispositivos de usuario multiconectados y esto plantea un nuevo reto: cómo aprovechar esta diversidad de conexiones para una mejor experiencia de usuario.

2.1.2 Ventajas de la multiconectividad

Las ventajas que ofrece un dispositivo de usuario multiconectado son múltiples ([9]):

1. **Cobertura extra**: un dispositivo con varias interfaces de red heterogéneas puede conectarse a diferentes tipos de redes y por tanto es más fácil que esté "conectado en todo momento".

2. **Tolerancia a fallos de red**: si se dispone de varias conexiones redundantes es más probable sobrevivir a un fallo de red en una de ellas, utilizando las otras.

3. **Movilidad**: disponer de varias interfaces de red puede ayudar a reducir el impacto de la movilidad en las comunicaciones, ya que es posible utilizar el solapamiento de coberturas de diferentes tecnologías de acceso para realizar traspasos de duración e impacto reducidos. El tema principal de esta tesis es mejorar la movilidad de dispositivos de usuario multiconectados, por lo que trataremos este punto con detalle en las secciones sucesivas.

4. **Elección óptima de la red de acceso**: Un dispositivo multiconectado puede elegir qué red de acceso utilizar para sus comunicaciones, basándose en criterios como el coste para el usuario, la calidad de servicio, el ahorro energético, etc. Por ejemplo, si cada tecnología de acceso tiene asociado un consumo de potencia por parte del dispositivo, disponer de alternativas puede ayudar a ahorrar energía utilizando en cada momento la interfaz con menor consumo de potencia.

5. **Distribución de carga**: Cuando se dispone de varias interfaces de red se puede distribuir el tráfico de diferentes aplicaciones entre todas ellas, lo que disminuye la carga individual de cada una. La idea es agregar la calidad de servicio y propiedades de transmisión de todas ellas para obtener un total superior al de cada una. Por ejemplo se puede agilizar una descarga de Internet utilizando simultáneamente el ancho de banda de varias redes de acceso.

6. **Transferencia simultánea**: Consiste en duplicar el tráfico por varias rutas, lo que aumenta el tráfico total cursado pero puede mejorar la calidad de servicio extremo a extremo. Por ejemplo, se pueden ignorar las pérdidas de paquetes por una de las interfaces ya que probablemente no coincidirán con las pérdidas sufridas en las otras interfaces.

Las cuatro primeras ventajas suelen enmarcarse bajo el término genérico de "movilidad de usuario", y constituyen el tema principal de esta tesis. Las dos últimas ventajas (distribución de carga y transferencia simultánea) quedan fuera del ámbito de estudio de esta tesis.

2.1.3 Retos de la multiconectividad

Las ventajas de la multiconectividad enumeradas en la sección anterior resultan evidentes hasta para usuarios sin formación telemática formal.

Sin embargo, a pesar de existir multitud de dispositivos multiconectados y de disponer a menudo de diversas redes de acceso, hoy en día sigue siendo difícil sacarle partido a la multiconectividad de una forma dinámica y sencilla para el usuario.

En lo que al protocolo IP respecta, tener varias interfaces de red implica tener varias direcciones IP. Los protocolos de transporte tradicionales, como TCP, no están pensados para sobrevivir a un cambio de las direcciones IP de sus interlocutores, por lo que es necesario reiniciar las conexiones si se quieren utilizar redes de acceso alternativas.

Ejemplo: intente desconectar su cable Ethernet mientras descarga un fichero de un servidor web o mantiene una conversación por Skype, aunque disponga de una conexión Wi-Fi adicional, esto no evitará que tenga que reiniciar la descarga o volver a arrancar la conversación. En §2.2 presentamos con más detalle los problemas relativos a la movilidad a nivel de protocolos y en concreto en redes IP.

La distribución de carga y la transferencia simultánea por varias interfaces, temas que no tratamos en esta tesis, presentan problemas similares, que hoy en día sólo pueden solucionarse haciendo una gestión directa de las conexiones de red desde las aplicaciones, lo que resulta inapropiado excepto para aplicaciones diseñadas específicamente para este propósito. A este respecto se están desarrollando protocolos específicos y modificaciones a protocolos ya existentes para solucionar estos temas por debajo del nivel de aplicación ([10, 11, 12, 13]).

En lo que respecta al sistema operativo, nos encontramos también con diversos problemas. Desde el principio, los sistemas operativos con soporte de red han sido capaces de utilizar varias interfaces de red sin problemas, pero aún hoy en día resulta casi imposible gestionar dicha variedad de una manera inteligente, sencilla y dinámica.

Ejemplo: intente utilizar una interfaz para el envío de datos de una aplicación y otra interfaz para el del resto de aplicaciones. En §2.3 presentamos un estudio más detallado de la gestión que los sistemas operativos modernos hacen de la multiconectividad.

Otro de los retos que plantea el aprovechamiento inteligente de la multiconectividad es que desde el momento en que tenemos la capacidad de elegir qué interfaz o interfaces utilizar, tenemos que disponer de políticas *automatizadas* para tomar estas decisiones ya que en entornos ubicuos no resulta realista cargar al usuario con tales decisiones.

Desde nuestro punto de vista, se puede mejorar notablemente el aprovechamiento de los dispositivos de usuario multiconectados resolviendo algunos de estos problemas.

2.2 Movilidad a nivel de protocolo

2.2.1 Definición y variantes

La movilidad es la capacidad de algunos dispositivos de cambiar su punto de conexión a la red durante su normal funcionamiento. Esto no implica que el dispositivo se desplace *físicamente*, pero, a menudo, éste es el caso.

Ejemplo: si nos movemos mientras mantenemos una conversación por un móvil GSM, es posible que a lo largo de esa conversación nuestro dispositivo vaya conectándose a diferentes estaciones base de telefonía, según salimos del radio de cobertura de unas y entramos en el de otras. La mayor parte de las veces, estos cambios de estación base son imperceptibles por el usuario, ya que lo protocolo GSM ofrece un buen soporte a la movilidad.

Para profundizar un poco más en los diferentes aspectos de la movilidad es conveniente introducir algo de nomenclatura ([14]) que nos permita tener un punto de vista más general:

Hardware involucrado

En cuanto a terminología general sobre el hardware involucrado en la movilidad, tenemos:

- **Dispositivo móvil**: un ordenador portátil, un teléfono móvil o cualquier otro hardware de usuario, de dimensiones y peso reducidos, que lo hacen susceptible de ser desplazado durante su normal funcionamiento, lo que implica un cambio en su punto de conexión a la red.

  En ocasiones otros dispositivos menos portátiles se consideran también dispositivos móviles, incluso si no se mueven, ya que un cambio de conexión a la red puede venir provocado por circunstancias ajenas al movimiento, como cuando desconectamos nuestro ordenador de sobremesa de una red cableada Ethernet, por dificultades técnicas por ejemplo, y pasamos a usar una red Wi-Fi disponible desde esa misma ubicación.

  Otras veces es la propia infraestructura de red la que se desplaza, por ejemplo, una femtocelda o un punto de acceso Wi-Fi embarcados en un tren en movimiento. En estos casos un dispositivo de usuario *inmóvil*, situado, por ejemplo, en un andén de una estación y que está conectado a

una de estas infraestructuras en movimiento, se considera un dispositivo móvil ya que tarde o temprano perderá cobertura con su punto de conexión embarcado y tendrá que conectarse a otro punto de conexión disponible.

Siguiendo con este mismo ejemplo, un dispositivo que viaje en el interior de uno de estos trenes, podría considerarse un dispositivo *no* móvil, a efectos de esta discusión, ya que mantendrá la conectividad con su punto de conexión a la red a lo largo de todo el trayecto. Piense en el término "dispositivo móvil" desde una perspectiva relativista.

Estas infraestructuras de red móviles quedan fuera del ámbito de esta tesis por lo que únicamente hablaremos de dispositivos móviles de usuario.

El término en inglés para designar a un dispositivo móvil de usuario es mobile host que abreviaremos como MH en este documento por ser la denominación más habitual en la literatura científica.

- **Punto de conexión a la red** es un dispositivo hardware que permite a un dispositivo de usuario conectarse a Internet o a cualquier otra red en particular. Por ejemplo un concentrador Ethernet con su latiguillo, un punto de acceso Wi-Fi o una estación base de telefonía GSM o UMTS.

  El descubrimiento de puntos de conexión disponibles, la configuración necesaria para usarlos y los protocolos de comunicaciones involucrados en este proceso no son el objeto de esta tesis y por lo general supondremos que, tanto los puntos de conexión a la red como los dispositivos de usuario utilizan tecnologías como DHCP ([15]), LLMNR ([16]) u otras tecnologías como las desarrolladas por el grupo de trabajo Zeroconf del IETF ([17]), que permiten solucionar estos detalles, aunque esto no resulte trivial en algunos casos.

  En general los puntos de conexión a la red son infraestructuras preinstaladas que no se desplazan. Aunque existen ejemplos de lo contrario: a veces estas infraestructuras se embarcan en vehículos, para dar cobertura en su interior (como el ejemplo anteriormente mencionado del punto de acceso Wi-Fi embarcado en un tren o en un crucero) o para dar cobertura en su exterior, como un vehículo de comunicaciones militar que permita a las fuerzas desplegadas en tierra mantener comunicación radio entre ellas.

  En ocasiones el punto de conexión a la red es consecuencia de una asociación temporal de dispositivos de usuario y no parte de una infraestructura preinstalada fija, como en el caso de las redes inalámbricas ad-hoc de varios saltos, como las redes vehiculares. En estos casos los dispositivos de usuario *vecinos* se convierten en los puntos de conexión a la red, pudiéndose dar el caso de ser estos más móviles y dinámicos que el propio dispositivo móvil.

- **Interfaz**: o interfaz de red, es un componente hardware, parte de un dispositivo de usuario, que le permite conectarse a un punto de conexión a la red, ya sea mediante un cable o mediante transmisión radioeléctrica en espacio libre. Ejemplos de interfaces de red son las tarjetas PCI Ethernet

o Wi-Fi, los drivers UMTS de radiofrecuencia de un teléfono móvil, el driver hardware Bluetooth de un portátil o un módem de telefonía fija tradicional.

Un dispositivo multiconectado tendrá varias interfaces de red y se dice que es *heterogéneo* si las interfaces son de tecnologías diferentes. Por ejemplo un portátil con interfaces de red Ethernet y Wi-Fi es un dispositivo multiconectado heterogéneo ya que permite conectarse a puntos de conexión de tecnologías Ethernet y Wi-Fi.

En el contexto de esta tesis, nos centraremos en dispositivos móviles de usuario multiconectados y heterogéneos, terminales tales como portátiles o teléfonos móviles de última generación.

Dado que nuestro campo de investigación gira entorno a protocolos para redes IP, los puntos de conexión a red que mejor se ajustan a nuestra investigación son aquellos que suelen utilizarse en redes IP como por ejemplo Ethernet, Wi-FI, GPRS, UMTS, PPP y LTE. Tecnologías con propósitos más específicos como Bluetooth o IrDa sólo tendrán cabida cuando se utilizan como transporte para datos IP.

Otras tecnologías como GSM, diseñadas para la transmisión de voz mediante protocolos fuera del mundo IP no tienen relación con esta tesis. A pesar de esto, utilizamos la tecnología GSM en algunos ejemplos, ya que es una tecnología ampliamente extendida, razonablemente conocida y que resuelve con bastante éxito el problema de la movilidad.

Traspasos

En todo proceso de movilidad está involucrado un *traspaso*, es decir, *el procedimiento mediante el cual un dispositivo móvil deja de usar un punto de conexión a la red y pasa a usar otro.*

Por ejemplo, si mantenemos una conversación telefónica por un terminal GSM mientras nos desplazamos, el MH irá conectándose a diversas estaciones base por el camino, realizando varios traspasos durante la conversación. El procedimiento para los traspasos se resume, a grandes rasgos, en los siguientes pasos:

- Paso 0: El MH está conectado a una estación base concreta, su punto de conexión a la red, por el que envía y recibe los datos de la conversación telefónica

- Paso 1: Conforme el MH se aleja de la estación base original, se pierde calidad en la señal radio hasta el punto que mejor sería conectarse a otra estación base diferente. La red de telefonía conoce estos detalles ya que tanto dispositivos móviles como estaciones base, mantienen a la red informada de la calidad con la que se reciben unos a otros. La red de telefonía decide que el MH debería cambiar de estación base e informa al MH, a la estación antigua y a la nueva, para prevenirles del inminente cambio.

- Paso 2: El MH cierra la conexión con la estación base antigua y se conecta a la nueva. Durante el tiempo que tarde en efectuarse esta operación, el MH permanece desconectado de la red telefónica, por lo que no puede transmitir los datos de voz de la conversación telefónica. Para evitar una degradación en la calidad percibida por los usuarios durante esta operación, se procura mantener a todas las partes informadas y sincronizadas, reduciéndose la duración del periodo en que el MH permanece desconectado.
- Paso 3: El MH ya está conectado a la nueva estación base y la conversación de voz continúa sin problemas por esta nueva ruta.

Los detalles de como se efectúa un traspaso en las diferentes tecnologías de movilidad varían de una o otra, pero todas ellas comparten algunos puntos clave, a saber:

- **Conocimiento de los puntos de conexión disponibles**: Algún elemento del sistema ha de ser consciente de los diversos puntos de conexión disponibles para el MH.

  Normalmente es el propio MH quien se encarga de esto, recopilando periódicamente indicios de la existencia de todos los puntos de conexión disponibles, mediante métodos de monitorización pasivos y/o activos.

  En tecnologías inalámbricas es habitual que los puntos de conexión ayuden al MH en esta tarea, anunciando su presencia y características a intervalos regulares mediante mensajes baliza (del inglés beacon).

- **Iniciador del traspaso**: es el elemento del sistema encargado de decidir en que momento se realiza el traspaso. Esta decisión se basa en los *criterios de activación del traspaso*, que suelen calcularse en función de medidas de calidad de conexión entre el MH y sus puntos de conexión. Estas medidas podrán ser recopiladas por diferentes elementos del sistema y habrán de ser transmitidas al iniciador del traspaso cada cierto tiempo.

- **Duración del traspaso**: Dependiendo de la tecnología de movilidad utilizada, el traspaso tendrá una duración determinada. Dependiendo también de la tecnología utilizada y las capacidades del terminal, la normal transmisión de los datos se podría ver afectada durante el traspaso, como en el ejemplo anterior sobre GSM, por lo que la duración del traspaso es un factor determinante a la hora de evaluar y comparar tecnologías de movilidad.

- **Señalización durante el traspaso**: Al menos el MH y los puntos de conexión antiguo y nuevo deben intercambiar algún tipo de señalización durante el traspaso mediante la cual se liberan los recursos de la conexión antigua y se reservan los de la nueva.

  En caso de sistemas de movilidad seguros, esta fase implica el intercambio de la información de autenticación y cifrado características de estos sistemas.

En infraestructuras comerciales, un traspaso puede implicar también un intercambio de información de tarificación.

Cuando el sistema de movilidad proporciona perfiles de calidad de servicio, esto debe ser gestionado también mediante información de señalización durante el traspaso.

El término en inglés británico para el traspaso es handover y en inglés americano es handoff. Organizaciones de los diferentes países utilizan uno u otro según su procedencia, por ejemplo el IETF, la ITU-T y el 3GPP suelen utilizar handover mientras que el IEEE suele utilizar handoff. En el mundo académico suele utilizarse el término británico handover.

Se pueden definir varios tipos de traspasos según el tipo de elementos involucrados (alcance del traspaso), la disponibilidad de los puntos de conexión a la red en el momento del traspaso y los efectos que el traspaso tendrá en los datos que están siendo transmitidos:

Alcance del traspaso

En cuanto al alcance del traspaso, es decir, a que nivel del modelo OSI afecta, tenemos:

- **Traspaso a nivel 2**: como el que se ha visto en el ejemplo anterior sobre GSM, o los traspasos que se producen entre puntos de acceso del mismo ESS en una red Wi-Fi utilizando protocolos como 802.11r ([18]) u 802.16e ([19]). Este tipo de traspaso es transparente a nivel IP o se percibe simplemente como una reconfiguración de las características de transmisión del nivel de enlace.

  Los criterios de activación de traspasos a nivel 2 no suelen definirse en los estándares de las tecnologías de movilidad, lo que permite a los fabricantes hardware competir entre ellos implementando criterios de activación de traspaso propios.

- **Traspaso a nivel 3**: como el que se produce entre dos puntos de acceso Wi-Fi de ESS diferentes o cuando pasamos de usar una Wi-Fi a usar GPRS en un teléfono móvil. Este tipo de traspasos implican un cambio en la dirección IP del MH, es decir, el MH deja de ser alcanzable por la dirección IP antigua y pasa a ser alcanzable por una nueva dirección IP propia de la nueva red a la que se ha conectado.

Esta tesis se centra exclusivamente en traspasos a nivel 3, es decir traspasos que implican un cambio de la dirección IP del MH. Las soluciones de movilidad a nivel 2 aplican únicamente a las tecnologías de acceso que las soportan, por lo que no permiten resolver el problema más general de la movilidad en dispositivos multiconectados con interfaces heterogéneas.

Iniciador del traspaso

También se pueden catalogar las tecnologías de movilidad según quien es el iniciador del traspaso. Existen dos casos bien diferenciados, cuando el traspaso es **iniciado por la red**, es decir por algún elemento del sistema ajeno al MH y cuando el traspaso es **iniciado por el dispositivo móvil**.

En general en esta tesis estaremos hablando de traspasos iniciados por el dispositivo móvil, ya que los traspasos iniciados por la red suelen presentar algunos inconvenientes, a saber:

- Dependencia de la infraestructura: Al menos el elemento de la red iniciador del traspaso debe estar presente y disponible, lo que genera una dependencia al MH de la infraestructura para poder ejercer la movilidad.

  Esto no es un problema en redes diseñadas con la movilidad en mente, como la red GSM. Sin embargo, para redes de propósito general como las redes IP, es preferible un sistema de movilidad con total independencia de la infraestructura de red, que ofrezca total autonomía al MH para ejercer la movilidad sin depender de elementos externos y sin que requiera ningún tipo de inteligencia de red. En esencia, es preferible preservar la naturaleza sin estado de la red, ubicando el soporte de movilidad en los terminales finales.

- Iniciador invisible: No todos los traspasos son predecibles, como en el ejemplo GSM al principio de esta sección. A veces un MH pierde la conexión con su punto de conexión abruptamente y sin avisar, lo que lo deja desconectado por completo de la red. En estas circunstancias es fundamental que el MH sea el propio iniciador del traspaso, ya que, aunque exista un soporte de movilidad en la red, esta no estará accesible desde el MH. Una vez más, la autonomía del MH para iniciar y llevar a cabo los traspasos es una característica fundamental.

Sin embargo, también existen algunas buenas razones para implementar sistemas de movilidad iniciados por la red, por ejemplo en redes comerciales puede resultar interesante que sea el operador de red el que inicie los traspasos, ya que éste tiene acceso a información sobre el estado global de la red que el MH generalmente ignora y puede utilizar la movilidad de los MH para gestionar mejor su tráfico interno y optimizar sus recursos.

Conexiones simultáneas

Una condición particularmente importante de un sistema de movilidad es si el MH puede conectarse al nuevo punto de conexión *antes* de empezar el traspaso. En el ejemplo anterior sobre GSM, el MH no podía conectarse a la nueva estación base hasta no haberse desconectado de la antigua, ya que los teléfonos móviles generalmente sólo disponen de una interfaz GSM, la cual sólo puede estar conectada a una estación base en cada momento.

En dispositivos multiconectados, existe la posibilidad de conectarse al nuevo punto de conexión antes de desconectarse del antiguo. Esto ayuda a reducir la

duración de los traspasos ya que el MH establece la nueva conexión e intercambia la señalización requerida antes de desconectarse del antiguo punto de conexión.

Idealmente, un dispositivo multiconectado podría realizar traspasos de duración cero si ha tenido tiempo suficiente para establecer la nueva conexión, de forma que los datos pueden dejar de enviarse por la interfaz antigua y pasar a enviarse por la nueva sin retraso alguno.

Según esto, se definen dos tipos de traspasos:

- **Traspasos brake-before-make**: en que la conexión al nuevo punto de conexión se realiza *después* de cerrar la antigua.

- **Traspasos make-before-brake**: en que la conexión al nuevo punto de conexión se realiza *antes* de cerrar la antigua.

En el contexto de esta tesis se tratan principalmente traspasos make-before-brake, ya que uno de los puntos de partida para esta tesis es aprovechar la creciente diversidad y disponibilidad de puntos de conexión que existen hoy en día, donde es habitual tener cobertura Wi-Fi y GPRS al mismo tiempo, por ejemplo.

Interfaces heterogéneas

Para poder aprovechar realmente la diversidad de tecnologías de acceso disponibles hoy en día, no basta con que un dispositivo sea multiconectado, es necesario que además sus interfaces sean *heterogéneas*, es decir, de tecnologías diferentes. Esto aumenta el número de puntos de conexión accesibles en cada momento y facilita poder realizar traspasos make-before-brake de duraciones reducidas.

A este respecto, se definen dos tipos de traspasos:

- **Traspasos verticales**: o entre tecnologías heterogéneas, donde el traspaso provoca que los datos pasen a transmitirse por interfaces de tecnologías diferentes

- **Traspasos horizontales**: donde el traspaso hace que los datos se transmitan por interfaces de la misma tecnología.

En el contexto de esta tesis nos interesan fundamentalmente los traspasos verticales ya que son los que mejor aprovechan la diversidad de puntos de conexión disponibles hoy en día. Por otro lado, los traspasos horizontales suelen resolverse mejor como traspasos a nivel 2, involucrando exclusivamente mecanismos de movilidad de la tecnología de acceso en cuestión y, en algunos casos, no requieren de la intervención de sistemas de movilidad implementados a nivel 3 o superiores.

Traspasos imperceptibles

El término inglés seamless handover, que traducimos por traspasos imperceptibles, se utiliza para calificar aquellos traspasos en los cuales no se aprecia un

cambio en la calidad del servicio, en su seguridad o sus prestaciones, tanto durante el traspaso como una que vez el dispositivo ya está conectado a la nueva red.

La mayoría de los sistemas de movilidad se diseñan con el objetivo de permitir traspasos imperceptibles o de aproximarse lo más posible a ello.

Nótese que la definición del término es bastante vaga. Dependiendo de la aplicación y el usuario, un traspaso puede resultar imperceptible o no. Por ejemplo, un sistema de movilidad que permita realizar traspasos de diez segundos de duración puede resultar imperceptible para una aplicación de descarga de correo electrónico en segundo plano, pero puede resultar inapropiado para una aplicación de VoIP.

2.2.2 Movilidad en redes IP

En el ámbito de estudio de esta tesis nos hemos centrado en sistemas de movilidad para redes IP.

Historia de la movilidad en redes IP

Cuando se diseñó el modelo OSI para la interconexión de dispositivos entre redes, la movilidad era un problema desconocido: los dispositivos no eran portátiles dado su gran tamaño y peso. Pocos dispositivos eran multiconectados, salvo los routers y servidores de alta disponibilidad.

Tanto los routers como los dispositivos con conexiones redundantes hacían un uso bastante estático de las interfaces de que disponían, de forma que no era habitual tener que cambiar de interfaz "en caliente". Además estos dispositivos estaban gestionados por administradores de red que se encargaban manualmente de realizar los cambios de interfaz, y resultaba admisible reiniciar las conexiones de una máquina para utilizar sus conexiones redundantes, de hecho en muchos casos era necesario reiniciar el propio dispositivo. En el caso concreto de los routers existen protocolos específicos, como BGP ([20]) por ejemplo, para modificar qué interfaces se encargan de encaminar qué datos.

Hoy en día existen dispositivos de tamaño y peso reducido, disponen de varias interfaces heterogéneas, y rara es la ocasión en que no tenemos cobertura de al menos dos tipos diferentes de redes inalámbricas. Por otro lado, estos dispositivos móviles ya no están administrados por expertos en redes y los traspasos pueden ser tan habituales que reiniciar las conexiones de red cada vez que se realiza un traspaso ya no resulta viable.

La mayoría de estos dispositivos ejecutan aplicaciones que utilizan el protocolo IP para comunicarse, sin embargo el protocolo IP no tiene soporte de movilidad.

Existen millones de usuarios con dispositivos perfectamente móviles, ejecutando aplicaciones IP en entornos con abundancia de redes de acceso y sin embargo no tenemos resuelto el problema de la movilidad a nivel IP.

Retos y problemas de la movilidad en redes IP

El protocolo IP permite la transmisión de datos entre dispositivos conectados a diferentes redes. Para ello, a las interfaces de cada dispositivo en una red IP se les asigna un identificador único, su dirección IP. En esencia, el protocolo IP establece un espacio de nombres para todas las interfaces, a las que les asigna una dirección IP única.

Por cuestiones de escalabilidad, la red IP global de Internet ha sido organizada según una topología jerárquica, al estilo de los prefijos telefónicos de la antigua red telefónica conmutada o de los códigos postales del sistema de correo tradicional. *Esto quiere decir que una dirección IP no sólo es un identificador único para las interfaces de los dispositivos IP, sino que también realiza la función de localizador topológico* (o geográfico si se prefiere, aunque no sea la palabra más adecuada).

Si un dispositivo IP realiza un traspaso a una nueva red, pasará a estar en una localización topológica de la red IP diferente, por lo que será necesario asignarle a su interfaz una dirección IP diferente.

Los dispositivos IP multiconectados pueden estar conectados simultáneamente a varias redes diferentes, por lo que cada una de sus interfaces se ubica topológicamente en lugares distintos de la red IP y por tanto cada una de sus interfaces tendrán direcciones IP diferentes, no sólo en los últimos bits, también en los que corresponden a los prefijos de red.

Es decir, *la movilidad de dispositivos a nivel IP implica un cambio en la dirección IP de las interfaces del dispositivo móvil*. Dado que la red IP encamina los datos según la dirección IP destino, es necesario que cuando un receptor IP sea móvil, el emisor (o algún punto intermedio de redirección) se mantenga al tanto de los cambios de dirección IP de su interlocutor. A esta argumentación se la conoce como el problema de la localización en redes IP móviles.

Cuando un dispositivo no es móvil, un protocolo como DNS ([21]) es suficiente para localizar una de sus direcciones IP. Sin embargo, cuando un dispositivo puede cambiar sus direcciones IP en cualquier momento, no basta para localizarlo una consulta a una base de datos estática, es necesario una suscripción a un servicio que notifique la nueva dirección IP cada vez que esta cambia.

Es así como la mayoría de protocolos de movilidad están intentando solucionar el problema de la localización. Estas notificaciones pueden tomar su tiempo y esto hace que las trasmisiones de datos a dispositivos móviles sufran retrasos más allá de la duración de sus traspasos. En nuestro trabajo en esta tesis supondremos que están resueltos todos estos problemas de localización.

La localización no es el único problema de la movilidad en redes IP, ni el más grave. Curiosamente, el problema más acuciante de la movilidad en redes IP proviene de un nivel mucho más alto en la torre de protocolos, del nivel de sesión, o más bien de su ausencia, y provoca que las aplicaciones IP vean desconectadas sus conexiones de transporte al realizarse un traspaso.

Cierre de conexiones de transporte durante los traspasos

Los protocolos no orientados a conexión suelen utilizarse para transacciones de datos de corta duración o para datos cuya utilidad expira con el tiempo.

Por contra, en el contexto de esta tesis, estamos interesados en el estudio de la movilidad en protocolos orientados a conexión: las transacciones de datos suelen ser de larga duración (pudiendo realizarse varios traspasos) y los datos retrasados no suelen ser descartables ni prescindibles (por lo que es importante realizar traspasos cortos y sin pérdidas).

El protocolo TCP pone en comunicación aplicaciones ejecutando en dispositivos IP. Para ello establece un nuevo espacio de nombres para las aplicaciones que ejecutan en un dispositivo, asignándoles uno o varios *números de puerto*. Para la identificación del dispositivo donde ejecutan estas aplicaciones, TCP utiliza la dirección IP de una de sus interfaces.

Nótese que cuando hablamos de movilidad, la asociación entre localizadores de interfaz y dispositivos es puramente temporal y volátil. Esto quiere decir que una conexión TCP no puede sobrevivir a un cambio en la dirección IP de una interfaz, ya que las direcciones IP de las interfaces involucradas son precisamente uno de los elementos identificativos de una conexión TCP. Dicho en otras palabras, una conexión TCP no se establece entre dos dispositivos sino entre los identificadores temporales de las interfaces de dos dispositivos. El concepto de dispositivo en si mismo es ajeno a TCP y a IP.

A nivel de aplicación, un usuario quiere contactar con otro usuario o con un servicio sin importarle demasiado el dispositivo en que reside o si éste cambia de red de acceso.

El eslabón que falta entre aplicaciones de usuario conversando entre ellas (usuarios con usuarios) y conexiones TCP entre identificadores temporales de interfaces (o direcciones IP si se prefiere) es precisamente el nivel de sesión, que por desgracia es casi inexistente hoy en día.

En situaciones de no-movilidad y con dispositivos no-multiconectados, es razonable pensar que la dirección IP de la interfaz del dispositivo destino identifica unívocamente a ese dispositivo, por lo que el nivel de sesión es innecesario. Una aplicación IP puede abrir un socket TCP contra la dirección IP de su destino y estará comunicándose con el dispositivo destino. Esto resulta económico y conveniente para el programador y así se ha hecho tradicionalmente.

Sin embargo, en escenarios de movilidad, donde un dispositivo cambia las direcciones IP de sus, en ocasiones múltiples, interfaces, no hay una relación unívoca y estable entre el dispositivo y la dirección IP temporal de una de sus interfaces.

En resumen, el programador de aplicaciones ha estado tradicionalmente asignando una tercera funcionalidad a la dirección IP: la de identificador único de dispositivo, aparte de las ya conocidas: identificador único de interfaces y localizador topológico de interfaces.

El problema es grave: las aplicaciones tradicionales TCP/IP no pueden funcionar en condiciones de movilidad. La base de aplicaciones TCP/IP es tan grande que reescribirlas todas sobre un nivel de sesión resulta inviable.

Se han propuesto numerosas soluciones a este problema, a todos los niveles de la torre de protocolos:

- Las soluciones por debajo del nivel de transporte intentan ocultar la movilidad a los niveles superiores, lo que puede dar buenos resultados siempre y cuando no se quiera optimizar el rendimiento de los protocolos superiores teniendo en cuenta la movilidad.

- Las soluciones por encima del nivel de transporte intentan gestionar la movilidad mediante el establecimiento de nuevas conexiones cuando se realiza un traspaso e introducen un nuevo espacio de nombres que asocia los dispositivos con sus direcciones IP temporales.

- Algunas soluciones utilizan infraestructura externa que redirecciona los paquetes entre el origen y el destino móvil, de forma que no es necesario que el origen se mantenga al tanto de los desplazamientos del destino.

- Otras soluciones se las apañan para modificar las conexiones, reconfigurándolas para la nueva dirección IP del destino, en lugar de cerrar la conexión antigua y establecer una nueva.

Veamos como funcionan, a grandes rasgos, algunos de los principales protocolos de movilidad IP.

2.2.3 Soluciones de movilidad en redes IP

MIH

Media-Independent Handover es una tecnología para dar soporte a la movilidad, estandarizada como 802.21 ([22]) por el IEEE en 2008.

El objetivo principal de esta tecnología es facilitar traspasos imperceptibles entre redes de acceso de unas tecnologías concretas: Ethernet, Wi-Fi, Bluetooth, WiMAX, GPRS, UMTS y CDMA2000.

El nivel que ocupa el protocolo MIH en la torre de protocolos es el 2,5, es decir por encima del nivel de enlace y las tecnologías de acceso y por debajo del nivel 3, y de las tecnologías de movilidad para los traspasos verticales.

El punto de partida de esta estandarización, según [23], era que los dispositivos móviles multiconectados heterogéneos presentan una incapacidad manifiesta para conectarse a una red de acceso adecuada a sus necesidades, a pesar de la existencia de protocolos de movilidad IP como los que veremos en las siguientes páginas.

Ejemplo: conectarse a la red de acceso por la que se recibe la mejor relación señal a ruido no asegura que la calidad de servicio asociada a esta nueva red permita un traspaso imperceptible ni el mantenimiento de la calidad percibida por el usuario.

Según esto, la propuesta de MIH es estandarizar los mecanismos por los que los diferentes protocolos de nivel 2 y de nivel 3 intercambian notificaciones de

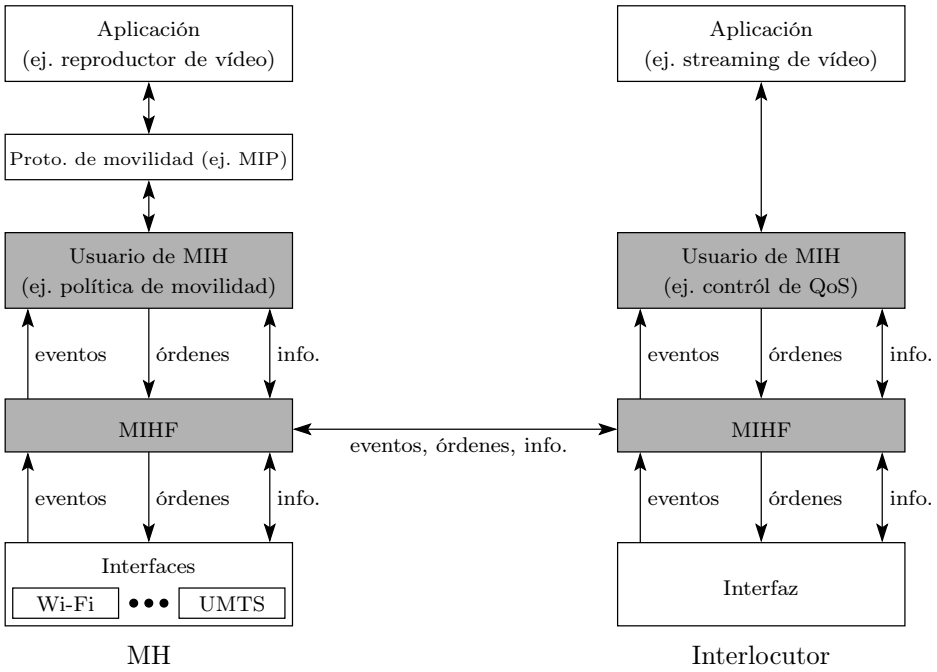

Figura 2.1: Elementos de una arquitectura MIH (fondo gris), resto de elementos del sistema (fondo blanco) y el tipo de mensajes que se intercambian.

eventos, envían e interpretan órdenes o se informan unos a otros sobre aspectos relevantes para la movilidad.

Los mensajes necesarios para transmitir información entre los niveles 2 y 3 de los dispositivos móviles y con los otros elementos de red involucrados en un traspaso se gestionan mediante el Media Independent Handover Function, o MIHF. Tanto los módulos de las tecnologías de acceso como los de los niveles superiores deben ser capaces de mantener diálogos MIHF entre ellos para anunciar la información de la que disponen sobre los traspasos y para aprovechar la de otros módulos, de forma que la movilidad resulte más efectiva.

La figura 2.1 ilustra las relaciones entre los diferentes niveles y dispositivos en MIH.

En este sentido, el ámbito de actuación de MIH es la iniciación y preparación para el traspaso, no la realización del traspaso en si mismo. Así, las tareas propias de MIH son el descubrimiento de las redes de acceso disponibles, la selección de la red más apropiada, la activación de la conexión a la red seleccionada y la obtención de la información necesaria a nivel IP para usar la nueva red de acceso. Los mecanismos para efectuar el traspaso se delegan en protocolos de movilidad IP propiamente dichos.

Inicialmente, MIH se centraba exclusivamente en redes inalámbricas tipo 802 como Wi-Fi o Bluetooth, pero tan pronto como se empezó a pensar en casos

de uso, enseguida se detectó la necesidad de ampliar la propuesta para incluir redes cableadas como Ethernet y redes de telefonía móvil como GPRS y UMTS, lo que obligó al IEEE a interactuar con otros organismos como el 3GPP.

Así mismo, el análisis de los casos de uso enseguida hizo evidente la necesidad de colaboración entre el dispositivo y la red, por varias razones: En algunos casos la red puede hacer recomendaciones a los dispositivos sobre cómo y cuándo realizar los traspasos, basándose en la información agregada de todos los dispositivos que alberga. En otros casos, la red es la que decide sobre los traspasos, enviando órdenes directas a los dispositivos, lo que permite implementar traspasos iniciados por la red.

MIP

Mobile IP es un protocolo, estandarizado por el IETF en el 2002 ([24, 25]), que permite a un dispositivo IP móvil ser contactado a través de su dirección IP original, aunque adquiera direcciones IP alternativas al moverse de una red a otra. También existe una versión de Mobile IP para IPv6, conocida como MIPv6 ([26]).

Mobile IP es un protocolo de nivel 3, que basa su solución de movilidad en el uso de un agente redirector. Para entender cómo funciona este agente redirector es necesario explicar antes algunos conceptos básicos de MIP:

- Red hogar o Home network: se supone que el dispositivo móvil pertenece a una red concreta (virtual o física) en la que tiene asignado una dirección IP permanente, que se conoce como su *dirección en el hogar* o Home address.

- Red visitada o Visited network: es el nombre que recibe la red a la que está conectado el MH cuando éste no se encuentra en su red hogar.

- Care-of-address: es la dirección IP que obtiene el MH mientras está conectado a la red visitada. Esta dirección IP cambia cada vez que el MH se conecta a una nueva red visitada.

- Agente en el hogar o Home agent: es un dispositivo que reside en la red hogar, que bien podría ser la propia puerta de enlace de dicha red, por ejemplo, si tuviese soporte MIP. Este dispositivo es el agente redirector.

El agente en el hogar se mantiene al tanto de los cambios de dirección IP del MH cuando éste se mueve de una red a otra, puede interceptar los datagramas dirigidos a la dirección en el hogar del MH y reenvía dichos datagramas a la care-of-addres del MH.

De esta manera un dispositivo que mantenga un intercambio de datos con el MH mandará los datagramas a la dirección en el hogar del MH, sin percatarse de si el MH está en una red diferente. El agente en el hogar se encargará de capturar los datagramas dirigidos al MH y reenviarlos mediante un túnel IP a la care-of-address del MH.

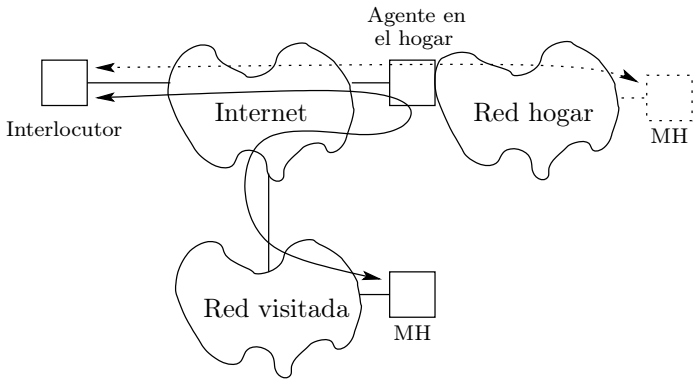

Figura 2.2: MIP: Rutas que siguen los datos entre el MH y su interlocutor, cuando el MH está en su red hogar (líneas punteadas) y cuando se ha desplazado a una red visitada (líneas continuas). El agente en el hogar hace de redirector cuando el MH está en una red visitada.

El MH a su vez utiliza su dirección en el hogar como dirección origen de sus respuestas a su interlocutor y las envía por un túnel IP a su agente en el hogar, que se encargará de extraerlas del túnel y reenviarlas al interlocutor del MH.

De hecho, el interlocutor del MH no necesita soportar MIP para mantener sus conexiones con el MH, ya que para él, todo el proceso y la infraestructura de MIP en la red hogar del MH son invisibles. La figura 2.2 ilustra el funcionamiento de MIP.

En caso de que la red visitada tenga soporte de MIP, el MH puede utilizar una variante del procedimiento anterior, en la que un dispositivo residente en la red visitada, el agente en la red visitada, se encarga de mantener el túnel con el agente en el hogar.

El MH debe mantener informado a su agente en el hogar sobre la care-of-address que se le asigna en cada red visitada cada vez que cambia de red, bien sea mediante comunicación directa con su agente en el hogar o bien a través del agente en la red visitada, si existiera.

El encaminamiento triangular de los datos que viajan al MH desde su interlocutor es uno de los problemas principales de MIP. No hay soluciones suficientemente seguras para evitar este problema, pero MIPv6 ya dispone de una optimización de ruta que permite al MH comunicar su cambio de dirección de forma segura a su interlocutor. Esto permite que ambos extremos pasen a comunicarse directamente, sin necesidad de utilizar el agente redirector en el hogar, con el único inconveniente de que se hace necesario un soporte de MIPv6 en el interlocutor del MH.

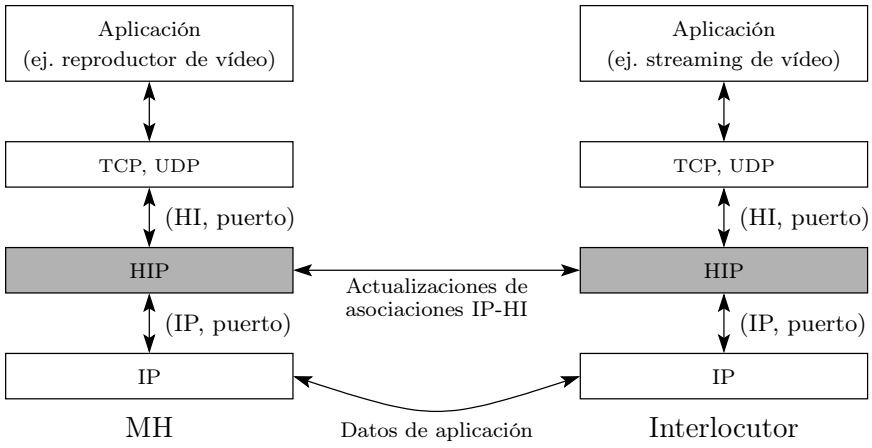

Figura 2.3: Arquitectura de movilidad en HIP. Componentes HIP en fondo gris, otros componentes en fondo blanco y los mensajes que se intercambian.

HIP

Host Identity Protocol es un protocolo experimental propuesto por el IETF en el 2008 ([27, 28]). El objetivo de HIP es evitar que a las direcciones IP se les atribuya esa tercera funcionalidad como identificadores de dispositivo (ver §2.2.2).

Para ello, HIP implanta un nuevo espacio de nombres global, que asigna un identificador único a cada dispositivo, denominado host identities, HI o identificadores de dispositivo. Estos identificadores de dispositivo se asocian a las múltiples y cambiantes direcciones IP de sus interfaces, desacoplando la funcionalidad de las direcciones IP como identificador de dispositivo y de interfaz, mientras se mantiene la funcionalidad de la dirección IP como localizador topológico.

El protocolo HIP se ubica al nivel 3,5 de la torre de protocolos, justo por encima del protocolo IP y debajo de los protocolos de transporte, como TCP.

Cuando se usa el protocolo HIP, el nivel de transporte ya no utiliza direcciones IP para identificar sus conexiones, sino que utiliza, sin saberlo, identificadores de dispositivo.

Los identificadores de dispositivo son identificadores criptográficos únicos para cada dispositivo, que no cambian con la movilidad ni dependen del número de interfaces disponibles en el dispositivo. El protocolo HIP se mantiene al tanto de las direcciones IP por las que es alcanzable un dispositivo y traduce los identificadores de dispositivo que usa el nivel de transporte a las direcciones IP que usa el protocolo IP.

Los identificadores de dispositivo tienen el mismo tamaño que una dirección IP de forma que para los protocolos de transporte son indistinguibles de una dirección IP. Esto permite que se puedan usar los protocolos de transporte

tradicionales como TCP o UDP sin que sea necesario modificarlos.

Cuando un dispositivo móvil cambia de red de acceso, y por tanto cambian sus direcciones IP, su pila HIP y la de su interlocutor intercambian la señalización necesaria para que se actualicen las asociaciones entre su identificador de dispositivo y las nuevas direcciones IP. Los protocolos de transporte ni siquiera se percatan de este intercambio y siguen utilizando los identificadores de dispositivo, que permanecen inalterados. La figura 2.3 ilustra el funcionamiento de HIP.

SCTP

Stream Control Transmission Protocol es un protocolo de transporte con soporte nativo de multiconectividad, estandarizado por el IETF en el 2000 ([29, 30]).

El protocolo SCTP tiene funcionalidades similares a TCP y UDP y comparte muchos de sus mecanismos y algoritmos. Es un protocolo unicast orientado a conexión, fiable y full duplex, con un control de flujo y de congestion TCP-friendly y orientado a mensajes. La API de sockets para SCTP ([31]) ha sido diseñada con la idea de facilitar la portabilidad de aplicaciones a SCTP y tiene notables similitudes con la API de sockets de TCP y UDP. SCTP soporta el uso de IP versión 4 y 6.

El motivo original del diseño de SCTP fue resolver los problemas que presentaba TCP como protocolo de transporte en sistemas multiconectados donde la tolerancia a fallos de red fuera crítica, como la señalización en redes de VoIP y los sistemas de autenticación, autorización y tarificación (AAA) en telefonía tradicional.

La capacidad de SCTP de soportar dispositivos multiconectados ayuda a aprovechar la redundancia de enlaces típicamente disponibles en este tipo de sistemas que requieren alta fiabilidad. En TCP, las conexiones tienen que restablecerse por alguna de las rutas redundantes si la principal sufre un fallo, mientras que en SCTP este traspaso es automático y las conexiones no se rompen.

La reciente abundancia de dispositivos finales de usuario multiconectados ha abierto las puertas a explorar los beneficios de usar SCTP en estos entornos, mucho más dinámicos que las redes entre servidores de las compañías telefónicas.

Las conexiones SCTP se denominan *asociaciones* y se establecen entre un conjunto de direcciones IP del emisor y un conjunto de direcciones IP del receptor, siendo posible transmitir los datos entre cualquiera de las combinaciones posibles de direcciones IP origen y destino, sin que sea necesaria la intervención del usuario o la aplicación. La figura 2.4 ilustra esta funcionalidad.

Utilizar un conjunto de direcciones IP para definir cada uno de los extremos de una asociación le da a SCTP un enorme potencial como protocolo de transporte para dispositivos multiconectados: para las aplicaciones es transparente, si así se desea, cuál de las rutas entre emisor y receptor se está utilizando en cada momento.

Los conjuntos de direcciones IP origen-destino que definen una asociación SCTP se deciden y anuncian en el establecimiento de la asociación, es decir,

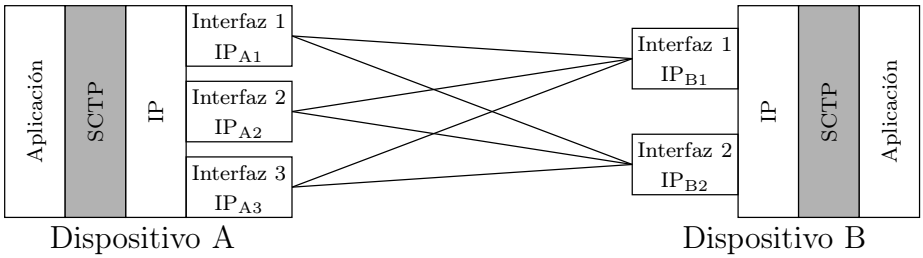

Figura 2.4: Ejemplo de las posibles rutas para los datos en una asociación SCTP entre dos dispositivos (de 3 y 2 interfaces respectivamente). Cualquiera de las rutas puede utilizarse para el envío de los datos, sin que los protocolos de niveles superiores tengan que preocuparse sobre qué ruta utiliza cada segmento de datos.

antes de la transmisión de datos. Existe una extensión a SCTP, conocida como extensión de movilidad o addip ([32]) que permite renegociar estos conjuntos de direcciones en una asociación ya establecida. Esto permite añadir y eliminar direcciones a la asociación dinámicamente, según el dispositivo se traslada de una red a otra, lo que ofrece a SCTP un enorme potencial como solución de movilidad para traspasos make-before-brake.

La extensión de movilidad de SCTP es de amplia implantación entre las implementaciones de SCTP. De ahora en adelante, cuándo nos refiramos a SCTP, supondremos implícitamente que estamos hablando de SCTP con su extensión de movilidad, lo que generalmente se conoce como mSCTP, las siglas en inglés de mobile SCTP.

El protocolo SCTP incorpora un mecanismo de supervisión de ruta mediante el cual es capaz de detectar cuándo las rutas sufren un fallo. De ahora en adelante nos referiremos al mecanismo de detección de fallo de ruta de SCTP como PFD, las siglas en inglés de path failure detection.

Cuando se detecta uno de estos fallos de ruta, SCTP comienza a utilizar una de las rutas alternativas para la transmisión de los datos, con la esperanza de que la nueva ruta esquive el segmento de red que está dando problemas. En terminología de movilidad, el PFD realiza la función de activador de los traspasos. Más adelante, en §3.3 explicamos con detalle el mecanismo de PFD.

En §4 proponemos una modificación de SCTP que permite ajustar la duración del PFD a los requerimientos de las aplicaciones.

SIP

Session Initiation Protocol es un protocolo de señalización del nivel de aplicación para crear, modificar y finalizar sesiones multimedia entre varios participantes, por ejemplo, llamadas telefónicas por internet o multiconferencias. Fue estandarizado por el IETF en 1999 ([33]).

Es un protocolo textual que recuerda e incorpora elementos de protocolos como HTTP ([34]) y SMTP ([35]) y está diseñado para ser independiente del

protocolo de transporte, siendo posible utilizar TCP, SCTP o UDP. También soporta las versiones 4 y 6 de IP.

Se entiende por sesión SIP el intercambio de información entre los participantes en una comunicación, que puede incluir datos de gestión más allá de la pura transmisión de los datos de usuario, como por ejemplo:

- La localización actual de cada participante, sus movimientos a otros terminales u otras redes de acceso o el conjunto de direcciones desde las que cada uno es accesible, en caso de ser participantes multiconectados.
- La disponibilidad de los participantes, si por ejemplo algunos se han desconectado.
- Las capacidades de los participantes, por ejemplo en términos de calidad de servicio de las aplicaciones multimedia implicadas en la visualización de los datos que se están transmitiendo entre los participantes.

El protocolo SIP introduce un nuevo espacio de nombres para los usuarios, a los que asocia un identificador SIP único, que tiene una forma similar a las direcciones de correo electrónico. Un servidor SIP, en algún punto de la red, se mantiene al tanto de la localización de cada uno de sus usuarios. Cuando se va a establecer una sesión SIP con un usuario, se consulta a su servidor para localizar al usuario y mediante un sistema de registro, los demás participantes se mantienen al tanto de los cambios de localización y direcciones IP de sus interlocutores.

Dependiendo del protocolo de transporte que se utilice para la transmisión de los datos, se puede gestionar la movilidad de usuarios estableciendo nuevas conexiones de transporte a sus direcciones más recientes, si se usa TCP por ejemplo, o actualizando las direcciones a utilizar en las asociaciones entre usuarios, si se utiliza SCTP por ejemplo. Este sistema permite traspasos brake-before-make, suponiendo que los dispositivos pueden actualizar su localización con su servidor SIP desde la nueva red de acceso a la que se han desplazado.

2.2.4 Comparativa y conclusiones

El protocolo IP permite comunicar dispositivos entre sí, sin importar donde se encuentren. Esto es gracias a que identifica unívocamente las interfaces de los dispositivos mediante su localización topológica en la red. En este sentido, resulta un protocolo útil en escenarios de movilidad, ya que permite que los datos alcancen su destino independientemente de la localización de éste. Si añadimos a esto que IP es un protocolo ampliamente implantado, resulta el protocolo natural para una solución global de movilidad.

Sin embargo, el resto de protocolos que colaboran con IP no están preparados para resolver el problema de la movilidad de forma apropiada. El soporte de la movilidad no es el objetivo de ninguno de los niveles de la torre de protocolos y ninguno de los protocolos clásicos lo han tenido en cuenta. Esto genera diversos problemas en escenarios de movilidad:

Cierre de conexiones de transporte

En situaciones de movilidad, las conexiones TCP se rompen al cambiar las direcciones IP de los dispositivos, lo que provoca pérdidas de datos.

Los protocolos de movilidad por encima del nivel de transporte pueden gestionar estas conexiones para cerrarlas apropiadamente y reestablecer la comunicación abriendo nuevas conexiones a los nuevos destinos (como hace SIP).

Los protocolos a nivel de transporte pueden aplicar esta misma solución de una forma más eficiente, al ser capaces de olvidar las rutas antiguas y comenzar a utilizar las nuevas, sin llegar a romper sus conexiones (como hace SCTP y hasta cierto punto MIPv6 con optimización de ruta).

Tanto SIP como SCTP presentan el problema de no ser compatibles con el gran parque de aplicaciones ya existentes, que en su mayoría utilizan directamente un protocolo de transporte como TCP. Portar estas aplicaciones a otros protocolos de transporte o insertar un nivel de sesión en ellas puede resultar complicado y caro y es un proceso propenso a generar nuevos errores de programación y eliminar los beneficios derivados de años de optimización al protocolo de transporte original.

Los protocolos de movilidad por debajo del nivel de transporte pueden *ocultar* el cambio de direcciones IP a los protocolos de transporte, lo que evita el cierre de las conexiones.

Para ello se pueden utilizar infraestructuras redirectoras (como los agentes en el hogar de MIP) o bien realizar una gestión directa extremo a extremo de las nuevas direcciones, mientras se mantiene el mismo identificador de conexión (como en el caso de HIP).

Ocultar la movilidad de los dispositivos al nivel de transporte tiene algunos inconvenientes: los protocolos de transporte suelen incorporar mecanismos para optimizar la transmisión de los datos a las características de la ruta, como por ejemplo el control de congestión de TCP y SCTP. Estos mecanismos suelen basar sus decisiones en la historia previa de la conexión.

Si los protocolos de transporte no son conscientes del cambio de ruta, estarán aplicando estos mecanismos a la nueva ruta creyendo que aún se está usando la antigua, lo que puede llevar a comportamientos subóptimos hasta que se recopilen suficientes evidencias por la ruta nueva como para anular los históricos acumulados por la ruta antigua.

Ineficiencia en las rutas

Los protocolos que utilizan infraestructuras redirectoras presentan el problema de que las conexiones entre los extremos de una comunicación han de pasar por los elementos redirectores, como en el caso de MIP.

Las rutas que siguen los datos al tener que pasar por estos elementos redirectores pueden no ser las rutas óptimas entre los extremos de la comunicación, lo que alarga innecesariamente el trayecto de los datos, introduce problemas derivados de rutados asimétricos y los hace vulnerables a las restricciones de calidad de servicio de los segmentos de red próximos a los elementos redirectores.

Cuando en lugar de utilizarse elementos redirectores, son los extremos de la comunicación los que se mantienen al tanto de los traspasos de sus interlocutores, las rutas que siguen los datos resultan más eficientes. Protocolos como MIPv6 con optimización de ruta, HIP, SCTP y SIP mantienen los extremos de sus conexiones al tanto de los traspasos de sus interlocutores y por tanto utilizan rutas eficientes.

Localización previa a la conexión

En caso de querer establecer conexiones con dispositivos que ya estén en movimiento, es necesario localizarlos allá donde estén. El sistema tradicional de DNS en colaboración con DNS dynamic updates (dDNS, [36]) es una solución a este problema de localización.

Aún así, no todos los dispositivos de usuario disponen de un nombre de dominio. Sería necesario, por tanto, identificar a los dispositivos de forma independiente de su localización y mantener un servicio de localización basado en ello lo suficientemente dinámico.

Resulta inevitable utilizar infraestructuras de red preinstaladas para este propósito, ya que al menos los dispositivos proveedores del servicio de localización han de ser localizables por todos los dispositivos.

Como alternativa a dDNS algunos protocolos de movilidad proporcionan sus propias infraestructuras para la localización como los agentes en el hogar de MIP o los servidores SIP.

Sea como fuere, estas infraestructuras y servicios deben ser notificados por el MH al realizar los traspasos para que su información se mantenga actualizada.

En el contexto de esta tesis no tratamos de resolver los problemas de localización asociados a la movilidad y se presupone el uso de protocolos como dDNS o SIP como complemento a aquellos protocolos de movilidad que no ofrecen un servicio de localización propio, como SCTP y HIP.

Necesidad de infraestructura

Algunos de los protocolos explicados hasta el momento requieren de la existencia y colaboración de elementos de red distintos a los extremos de la comunicación, ya sean estos infraestructuras físicas como el caso de los agentes en el hogar de MIP, o servicios software como es el caso de los servidores de SIP.

Otros protocolos de movilidad requieren únicamente de la participación y colaboración de los extremos de la comunicación, como es el caso de SCTP y HIP.

En nuestra opinión, es preferible una solución de movilidad que no requiera de elementos adicionales ajenos a los extremos de la comunicación, libre de ningún tipo de infraestructura de red preinstalada e independiente de la existencia de cualquier inteligencia en la red. Esto permite la movilidad de los usuarios finales en todo momento y mantiene la red libre de información de estado, preservando el principio extremo a extremo de su diseño ([37]).

2.3 Movilidad en el sistema operativo

Los sistemas operativos de propósito general ofrecen un soporte limitado para aprovechar la multiplicidad de interfaces de red. Cuando entorno a 1990 empezaron a popularizarse y abaratarse las redes locales, los ordenadores personales comenzaron a venir equipados con las primeras interfaces de red.

Los sistemas operativos de propósito general que controlaban este hardware de consumo adoptaron las primeras APIs de programación en red, e incorporaban los siguientes elementos para manejar múltiples interfaces de red:

- Una API de programación de red, **la API de sockets**, que oculta a las aplicaciones el número y naturaleza de las interfaces disponibles. En este sentido se dice que la API hace las interfaces de red transparentes (o invisibles) a las aplicaciones.

- **La tabla de rutas del núcleo**, que resuelve, en tiempo de ejecución, qué interfaz se encargará de enviar cada datagrama. En este sentido realiza la misma tarea que un router. De hecho, la tabla de rutas del núcleo es el primer router que atraviesa todo datagrama.

Vamos a ver con un poco de detalle el funcionamiento básico de estos sistemas.

2.3.1 Nombrado de interfaces

En los sistemas operativos se asigna un nombre a cada interfaz de red. Estos nombres se introducen por conveniencia para el usuario, ya que, internamente, el sistema operativo utiliza normalmente un número, que actúa como índice en una tabla de interfaces.

En diferentes sistemas operativos se utilizan diferentes estrategias para el nombrado de las interfaces de red, pero todas ellas derivan de la que se utilizaba originalmente en UNIX, que además sigue siendo la que se utiliza hoy en día en multitud de dispositivos de usuario multiconectados, como por ejemplo en algunas tablets y teléfonos móviles.

En UNIX/Linux el nombre de una interfaz de red suele consistir en una palabra que identifica el tipo de interfaz, su tecnología o alguna de sus características, seguido de un número por si hay más de una interfaz de ese tipo.

Ejemplos habituales de estos nombres son: `lo` para la interfaz de loopback, `ppp0` para la primera interfaz lógica del tipo Point-to-Point-Protocol, `eth0` para la primera interfaz Ethernet o `ath2` para la tercera tarjeta inalámbrica tipo Atheros.

Tradicionalmente el nombrado de interfaces lo realiza el núcleo de forma automática, aunque recientemente y gracias a `udev(8)`, el gestor dinámico de dispositivos, el administrador puede ocuparse personalmente de elegir los nombres para cada interfaz.

El nombre de cada interfaz de red se almacena en el núcleo como una cadena de caracteres de tamaño máximo `IFNAMSIZ`. En Linux 2.6 y anteriores `IFNAMSIZ` son 16 octetos.

La API que permite a las aplicaciones acceder a las interfaces y sus nombres se describe en `netdevice(8)`.

Para listar las interfaces de red activas junto con algunos de sus detalles puede usarse el comando `ifconfig(8)`:

```
; ifconfig -a
lo        Link encap:Local Loopback
          inet addr:127.0.0.1 Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING MTU:16436 Metric:1
          RX packets:8 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:1104 (1.0 KiB) TX bytes:1104 (1.0 KiB)

eth0      Link encap:Ethernet HWaddr 00:16:6f:65:d1:56
          inet addr:192.168.1.120 Bcast:192.168.1.255 Mask
              :255.255.255.0
          inet6 addr: fe80::216:6fff:fe65:d156/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
          RX packets:141 errors:1 dropped:1 overruns:0 frame:0
          TX packets:30 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1564567 (1.4 MiB) TX bytes:370698 (362.0 KiB)
          Interrupt:5 Base address:0xa000 Memory:dfdf9000-dfdf9fff

eth1      Link encap:Ethernet HWaddr 00:14:22:d7:c6:3b
          BROADCAST MULTICAST MTU:1500 Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)
          Interrupt:9
```

Listado 2.1: Mostrar las interfaces de red con el comando `ifconfig` en Linux

Otros sistemas operativos utilizan aproximaciones similares; en el siguiente ejemplo hemos utilizado el comando `ipconfig` en un sistema operativo Windows XP con una única interfaz llamada `Local Area Network`:

```
C:\>ipconfig

Windows IP Configuration

Ethernet adapter Local Area Network:

        Connection-specific DNS Suffix  . :
        IP Address. . . . . . . . . . . . : 192.168.1.60
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . : 192.168.1.1
```

Listado 2.2: Mostrar las interfaces de red con el comando `ipconfig` en Windows XP

2.3.2 La API de sockets

Las APIs de red de los sistemas operativos de propósito general están basadas en la API conocida como Berkeley sockets.

Esta API abstrae el número y naturaleza de las interfaces de red disponibles en el dispositivo, ocultando sus detalles al programador. De esta manera, las aplicaciones son independientes del hardware de red en que ejecutan y pueden ser programadas sin tener en cuenta los detalles de cada interfaz de red.

El principio detrás de este diseño es explotar la naturaleza por capas de la torre de protocolos OSI. Según esto, una aplicación no necesita ser programada teniendo en cuenta las particularidades de cada NIC y su programador no necesita saber nada sobre la configuración de hardware concreta en la que ejecutará su software.

En un SO de propósito general es deseable este aislamiento entre las aplicaciones y el hardware. En este sentido se dice que el hardware de red es transparente para las aplicaciones.

Si las aplicaciones no deciden cómo se encaminarán los datagramas que envíen, es necesario un servicio en el sistema operativo que se encargue de ello y este servicio es la tabla de rutas del núcleo. Veamos como funciona:

2.3.3 La tabla de rutas del núcleo

La tabla de rutas del núcleo es un servicio software del sistema operativo para el encaminamiento de los datagramas basado en la dirección destino de los mismos.

Los detalles concretos sobre el funcionamiento de las tablas de rutas del núcleo difieren para cada sistema operativo, incluso dentro del mismo sistema operativo, encontramos diferencias de una versión a otra. Pero detalles aparte, el funcionamiento básico es el mismo.

La tabla de rutas del núcleo resuelve, en tiempo de ejecución, qué interfaz de red se encargará de dar salida a cada datagrama que las aplicaciones quieren enviar.

Para ello, la tabla de rutas almacena una serie de *reglas*, las cuales especifican qué subredes destino son alcanzables por cada interfaz de red. Las tablas de rutas suelen soportar diferentes protocolos de nivel 3. Por ejemplo la tabla de rutas del sistema Linux en su versión 2.6.29 soporta los siguientes protocolos:

```
1 ; route --help
2 [...]
3   List of possible address families (which support routing):
4     inet (DARPA Internet) inet6 (IPv6) ax25 (AMPR AX.25)
5     netrom (AMPR NET/ROM) ipx (Novell IPX) ddp (Appletalk DDP)
6     x25 (CCITT X.25)
7 [...]
```

Listado 2.3: Extracto de la ayuda del comando `route(8)` donde se muestran los protocolos soportados por la tabla de rutas del núcleo

De ahora en adelante nos limitaremos a utilizar el protocolo IP en las explicaciones y ejemplos, por ser el más utilizado y con el que estamos más familiarizados.

La información para rellenar la tabla de rutas del núcleo se extrae de la configuración de cada interfaz en el momento de activarse. Los datos relevantes para ello son la dirección IP de la interfaz, la máscara de red y la dirección de la puerta de enlace, si existe.

Interpretación de la tabla de rutas

El listado 2.4 muestra la tabla de rutas del núcleo del dispositivo donde se está escribiendo este documento (un Linux 2.6.29). En él tenemos dos interfaces activas: `wlan0` y `eth0`. Ambas están conectadas a Internet por sus propias redes locales mediante sendas puertas de enlace; es un típico escenario de dispositivo multiconectado con conexión a Internet.

```
1 ; route
2 Kernel IP routing table
3 Destination  Gateway      Genmask         Flags Metric Use Iface
4 192.168.1.0  *            255.255.255.0   U     0      0   wlan0
5 default      192.168.1.1  0.0.0.0         UG    0      0   wlan0
6 10.0.0.0     *            255.255.255.0   U     0      0   eth0
7 default      10.0.0.2     0.0.0.0         UG    0      0   eth0
```

Listado 2.4: Estado de la tabla de rutas del núcleo en un dispositivo multiconectado a dos redes con acceso a Internet.

La elección de la interfaz de salida se realiza datagrama a datagrama, buscando una regla apropiada por la que alcanzar el destino[1].

Por ejemplo, en el caso de la tabla de rutas del listado 2.4:

- La línea 4 establece que los datagramas dirigidos a la subred `192.168.1.0/24` han de encaminarse directamente por la interfaz `wlan0`. Es decir son datagramas destinados a nodos de la propia red local a la que está conectada la interfaz.
- La línea 5 establece que cualquier otro datagrama se encamine a la puerta de enlace `192.168.1.1`, que como es parte de la subred indicada en la regla anterior, se encaminará según dicha regla, es decir por la interfaz `wlan0`.
- La interpretación combinada de las reglas de las líneas 4 y 5 establece que cualquier datagrama destinado a la red local de la interfaz `wlan0` se encamine directamente por esa interfaz, y que cualquier otro datagrama se encamine también por esa misma interfaz hacia su puerta de enlace.

  Estas dos reglas, por si mismas, resuelven el problema del encaminamiento para cualquier destino posible.
- Las líneas 6 y 7 forman el mismo patrón que las dos anteriores pero para la subred `10.0.0.0/24` a la que está conectada la interfaz `eth0`.
- El orden de las diferentes reglas, tal y como aparecen en la tabla de rutas del núcleo, corresponde al orden en que las interfaces se han ido activando, en el caso del ejemplo, primero se activó la interfaz `eth0` y luego la `wlan0`.

En general, cuando un dispositivo multiconectado dispone de múltiples interfaces de red, existe la posibilidad de que varias de ellas sean válidas a nivel 3 para encaminar los datagramas. Cada sistema operativo resuelve este problema de maneras diferentes, como veremos en la siguiente sección.

[1] Para ser justos, hay que decir que Linux tiene varias tablas de rutas, tal y como se explica en la §2.3.5. De momento, ignoraremos este hecho para no enturbiar la explicación.

Entradas en la tabla de rutas

Existen tres tipos de entradas en la tabla de rutas, dependiendo de a cuántos destinos representan:

- **Rutas individuales** (Host routes) o rutas punto a punto; identifican qué interfaz utilizar para alcanzar una dirección IP concreta.

- **Rutas de subred** (Subnetwork routes); asocian una interfaz de salida a todos los destinos de una determinada subred, utilizando su prefijo IP para identificarla. Ejemplo: líneas 4 y 6 del listado 2.4.

- **Rutas por omisión** (Default routes); asocian una interfaz de salida a cualquier destino que no esté siendo contemplado ya por otro tipo de rutas en la tabla de rutas. Ejemplo: líneas 5 y 7 del listado 2.4.

Cada vez que se manda un datagrama desde el dispositivo se ha de buscar en la tabla de rutas cuál es su interfaz de salida correspondiente, por ello se trata de mantener la tabla de rutas a un tamaño manejable y reducido: cuando la topología de la red lo permite, se prefiere utilizar unas pocas rutas de subred o incluso una ruta por defecto en vez de un número elevado de rutas individuales. En este sentido, las tablas de rutas del sistema operativo recuerdan a las tablas de rutas tradicionales de los routers IP.

En los dispositivos multiconectados de usuario es habitual que todas las interfaces disponibles estén conectadas a Internet, por lo que todas ellas son candidatas para alcanzar un destino IP público. En el caso de que más de una entrada en la tabla de rutas contemple un mismo destino, la mayoría de los sistemas operativos decidirán cual de ellas utilizar atendiendo a uno de los siguientes criterios:

- Utilizar la entrada con la *métrica* más pequeña. Las entradas en la tabla de rutas incluyen una métrica[2] que caracteriza cuan *lejos* queda el destino (ej. el número de saltos al destino). Esta es la aproximación utilizada antiguamente en UNIX y la que se usa hoy en día en Windows (Windows 2000 y sus sucesores, [38]).

  El concepto de métrica sólo tiene sentido en rutas individuales, ya que cada destino en una ruta de subred puede estar a un número diferente de saltos del origen. Ésta es la razón por la que el concepto de métrica ha ido cambiando con el tiempo, hasta que se ha convertido en un indicador *difuso* que refleja el grado de preferencia que tiene el administrador de la máquina para utilizar una interfaz u otra a la hora de encaminar datagramas a una subred determinada.

  Algunos demonios Linux todavía utilizan soluciones de este tipo para reflejar lo que han averiguado de la topología de red ([39]).

[2] Quinta columna en el listado 2.4.

- Utilizar la entrada más específica al destino, es decir, elegir las entradas individuales mejor que las de subred, o entre las de subred, elegir aquellas que representen menos direcciones. Esto es lo que se hace en los sistemas UNIX modernos y en Linux ([39, 40]). Este sistema puede verse como un sistema basado en métrica donde la métrica que se utiliza es el tamaño de la máscara de red de las entradas y se intenta maximizar esta métrica.

- Usar la primera entrada que se encuentre en la tabla, ignorando el resto de candidatas. Esto es lo que hace Windows NT ([41]). Nótese que el orden de ocurrencia de las entradas en la tabla depende normalmente del orden en que se introdujeron, por lo que este mecanismo favorece la utilización de interfaces que se han configurado recientemente.

Cualquiera de estas tres aproximaciones al problema es demasiado inocente para dar un uso inteligente a las interfaces disponibles en un dispositivo multiconectado. Sería interesante elegir la ruta teniendo en cuenta otros criterios como el precio asociado a usar una interfaz, su consumo energético, su seguridad, la calidad de servicio que ofrece...

Desgraciadamente, la tabla de rutas carece de la potencia expresiva suficiente para expresar estos criterios y sus combinaciones. En §6 proponemos una lista completa de criterios a considerar a la hora de seleccionar la interfaz de salida para cada datagrama.

2.3.4 Alternativas a la tabla de rutas

Las alternativas a la tabla de rutas propuestas hasta la fecha, como las que se explican en [8, 42, 43, 44, 45], comparten un esquema de funcionamiento similar:

P1) Miden y recopilan los datos y evidencias relevantes, como el ancho de banda o las pérdidas de cada interfaz inalámbrica en el dispositivo.

P2) Procesan estas medidas y calculan figuras de mérito asociadas a ellas con suficiente poder expresivo y potencial de comparación como para tomar decisiones sobre ellas, como por ejemplo la calidad percibida por el usuario en un streaming de vídeo.

P3) Manejan eventos externos que permiten algún tipo de conocimiento del contexto, como saber si el dispositivo está operando en modo batería o cuándo se le vuelve a conectar a una fuente de alimentación externa.

P4) Interpretan algún tipo de *lenguaje de descripción de políticas* que es capaz de expresar cómo se relacionan los elementos anteriores a la hora de tomar una decisión.

P5) Utilizan un sistema experto que aplica las políticas expresadas en el punto P4 a los datos recogidos en los puntos P1, P2 y P3.

Los servicios del sistema operativo son especialmente aptos para almacenar los datos y evidencias recopilados en el punto P1, y también resultan muy apropiados para transportar las notificaciones de los eventos a considerar en el punto P3.

Sin embargo, los sistemas operativos dominantes carecen de servicios para almacenar los datos del punto P1 y no suelen disponer de mecanismos homogéneos de notificaciones de los eventos relevantes al punto P3. Esto provoca que los middlewares de movilidad deben construir sus propios servicios de recopilación de datos y eventos sin la ayuda del sistema operativo. Los programas que miden estos datos quedan ligados al middleware que disemina sus resultados y las aplicaciones que usan los datos no son portables a otros middlewares.

Una aproximación más modular a base de servicios del sistema operativo permitiría acelerar el desarrollo de aplicaciones, haría el middleware más portable y evitaría la necesidad de soluciones ad-hoc en espacio de usuario. En §7 proponemos un nuevo servicio del sistema operativo para la recopilación y diseminación de los criterios de selección de interfaz.

2.4 Otros trabajos relacionados

A lo largo de los últimos 15 años han aparecido numerosos trabajos que abordan la problemática general de esta tesis desde otros puntos de vista o que proponen diferentes soluciones a las que hemos revisado en §2.2.3 y §2.3.

Por diferentes razones estos trabajos no han tenido un impacto tan notable como los mencionados anteriormente, pero algunos de ellos bien merecen unos comentarios, pues ilustran diferentes soluciones con sus propios méritos y desventajas.

SLM, gestión de movilidad a nivel de sesión

Session Layer Management o SLM ([46]) es una propuesta de nivel de sesión que gestiona las conexiones TCP/IP entre dispositivos móviles, redireccionando los flujos de datos de las aplicaciones entre conexiones TCP que se van creando según necesidad. SLM es consciente de la multiconectividad de los dispositivos, lo que permite sacarle cierto provecho.

SLM consiste en dos partes, la gestión de las conexiones en si mismas y la gestión de la localización. La gestión de las conexiones se sitúa en los dispositivos finales. La gestión de la localización requiere de infraestructura de red adicional.

A la hora de gestionar las conexiones TCP, SLM puede tratarlas individualmente o en grupo. Por ejemplo, si una navegador de Internet está utilizando varias conexiones TCP para la descarga de una página web y las imágenes incluidas en ella, SLM puede tratar todas estas conexiones como un único grupo, de forma que a todas ellas se les apliquen las mismas reglas de gestión.

De forma complementaria, SLM permite manejar independiente flujos de datos que pertenecen a la misma aplicación. Por ejemplo, durante una videoconferencia con datos de audio y vídeo, los datos de audio pueden disfrutar de

una prioridad más alta, de forma que pueden moverse a una red más fiable en caso de pérdida de calidad independientemente de los datos de vídeo.

Para diferenciar este tipo de extremos, las aplicaciones han de caracterizar sus flujos mediante identificadores de sesión e identificadores de grupo (que los agregan).

Las conexiones en los dispositivos se controlan mediante una entidad de gestión de la sesión, o SM, que contiene toda la lógica necesaria para inicializar y mantener las sesiones en dispositivos multiconectados o en situaciones de movilidad. Los identificadores de sesión y de grupo establecen un nuevo espacio de nombres en el SM y se gestionan mediante una "tabla de sesión", que se mantiene sincronizada entre los extremos de la comunicación y que permite una visión individual de dispositivos multiconectados.

La infraestructura de red necesaria para utilizar SLM se encarga de localizar el dispositivo. Utiliza un nodo de red, denominado User Location Server o ULS. El ULS es una base de datos de las localizaciones de cada dispositivo, que se mantiene al tanto de los movimientos y cambios de direcciones de sus interfaces, de forma similar a como lo haría un agente en el hogar en Mobile IP.

La identificación del ULS de un dispositivo por su interlocutor puede realizarse de dos maneras:

- Mediante una consulta al DNS preguntando por un nuevo tipo de registro dedicado a tal efecto.
- Durante el establecimiento de sesión SLM, si el dispositivo es alcanzable por medios tradicionales, como una consulta estándar al DNS.

En [47] se muestra una comparativa de SLM y Mobile IP. SLM reduce la latencia durante el traspaso, utiliza menos tráfico de control durante el traspaso y no sufre de encaminamiento triangular. En el lado negativo, SLM produce tráfico a ráfagas en determinadas circunstancias y requiere modificar notablemente las aplicaciones ya existentes para poder utilizarse. Ambos protocolos necesitan de infraestructura externa.

Dado que SLM está situado muy arriba en la torre de protocolos tiene semánticas y funcionalidad para gestionar flujos de aplicación de maneras independientes, lo que abre la posibilidad de gestionar los flujos de formas relevantes para el usuario. Esto resulta extremadamente útil en la práctica para dispositivos multiconectados en entornos ubicuos, como se explicará más adelante en §6 y la aproximación de SLM a estos asuntos ha servido de inspiración para algunos de las propuestas de esta tesis.

Tesla y Migrates

Tesla ([48]) es una capa de sesión que proporciona un nivel de abstracción a las aplicaciones sobre la tradicional API de sockets. Permite a las aplicaciones gestionar flujos de datos, de forma similar a como lo hace SLM, pero con unos objetivos más amplios: Tesla ofrece servicios de cifrado, compresión y traspaso de flujos de datos.

Al contrario que en SLM, los flujos de datos en Tesla no los identifica la aplicación sino que están definidos por las direcciones origen y destino, sus puertos y el protocolo de transporte que utilizan. De forma similar a SLM, las aplicaciones pueden agregar los diferentes flujos con el propósito de facilitar la gestión de los mismos cuando tienen características similares.

La gestión de los traspasos en Tesla se basa en Migrate ([49]), una iniciativa del MIT para explorar las posibilidades de los entornos ubicuos.

En Migrate se mantiene la conectividad extremo a extremo en los traspasos de forma similar a SLM: se ocultan los traspasos a las aplicaciones mediante el uso de un nuevo espacio de nombres y generando nuevas conexiones a nivel de transporte cuando cambian las direcciones IP de los extremos.

El proyecto Migrate hace hincapié en varios aspectos avanzados sobre los traspasos:

- Espacio de nombres global para la localización de los dispositivos o los servicios, mediante infraestructura de red adicional.
- Traspasos make-before-brake, para acortar la duración de los traspasos.
- Detección rápida de desconexiones, que permite a las aplicaciones saber cuando un fallo de red está afectando a las comunicaciones.
- Reconexión rápida, que permite a los dispositivos reestablecer las comunicaciones de forma rápida una vez se ha recuperado el contacto con los interlocutores.

Para llevar a cabo la detección rápida de desconexiones y la reconexión rápida, Migrate se basa en la información de conectividad que ofrecen los protocolos de transporte que utiliza, y cuando ésta no es suficiente para satisfacer las restricciones de tiempo de las aplicaciones, utiliza un servicio adicional, un agente de monitorización de conectividad, que mediante sondas activas vigila el estado de las rutas entre dispositivos. Esto ha servido de inspiración para algunas de nuestras propuestas (§4 y §5).

La gestión de las desconexiones se gobierna mediante un motor de implementación de políticas que permite la expresión de preferencias particulares para diferentes tipos de sesiones. Al mismo tiempo, Migrate permite la transferencia de información sobre las características de transmisión de los niveles de enlace a las aplicaciones, lo que permite a la aplicación colaborar en las decisiones del motor de políticas si lo desea.

La implementación de Migrate es completa y ha llevado a la modificación de algunas de las opciones TCP del núcleo de Linux 2.2 ([50]).

Tanto la manera de interactuar con el motor de políticas como algunos de los detalles de implementación en el núcleo de Linux[3] han servido también de inspiración para nuestra propuesta de un nuevo servicio del núcleo (§7).

[3]En especial las modificaciones a `/proc`.

TCP-R

TCP-R o TCP Redirection ([51]) es una extensión a TCP que permite que las conexiones sobrevivan al cambio en las direcciones IP de los extremos.

Este protocolo utiliza una aproximación similar a la de SCTP, evitando el uso de elementos redirectores.

Cuando se mantiene una conexión TCP entre dos extremos y uno de ellos realiza un traspaso, el dispositivo móvil anuncia su nueva dirección al otro extremo mediante un three-way handshake y la conexión TCP se actualiza a los nuevos valores de las direcciones IP. Evidentemente, el dispositivo móvil debe ser capaz de detectar los cambios en su dirección IP y la velocidad al hacerlo determinará en gran medida la duración del traspaso.

El intercambio de mensajes para la revisión de la conexión utiliza claves de autenticación y soporta diversos algoritmos de cifrado. Este intercambio es compatible hacia atrás con TCP, ya que los mensajes de TCP-R no afectan a la máquina de estados de una pila TCP estándar.

Gracias a este mecanismo criptográfico para la revisión de la conexión, TCP-R soporta con naturalidad traspasos brake-before-make, lo que resulta una ventaja notable sobre otros protocolos.

Cuando ambos extremos de la comunicación están al tanto de la nueva dirección del dispositivo móvil, se utiliza una estrategia tipo fast retransmit ([52]) para recuperar el tiempo perdido en el traspaso lo antes posible.

TCP-R no proporciona un servicio de localización previo a la conexión y debe ser complementado con DNS dinámico para sacarle partido a esta funcionalidad.

Este protocolo tampoco proporciona mecanismos propios para la detección del cambio de dirección IP y los autores proponen complementarlo con demonios en espacio de usuario para la notificación de dichos eventos.

Múltiples espacios de nombres de red

Linux, desde la versión 2.2, permite utilizar varias tablas de rutas simultáneamente. Un sistema de prioridades denominado RPDB (routing policy database) permite establecer el orden en que se deben consultar cada una de estas tablas.

Para decidir el orden de consulta de las tablas de rutas se pueden utilizar diversos criterios como la dirección de origen del paquete, su ToS[4], la interfaz por la que se recibió, el protocolo de aplicación que transporta, etc.

```
1 ; ip rule
2 0:      from all lookup local
3 32766:  from all lookup main
4 32767:  from all lookup default
```

Listado 2.5: Usando el comando `ip(8)` para mostrar las prioridades de las diferentes tablas de rutas del núcleo.

[4]El ToS o type of service es un campo de la cabecera IP que permite caracterizar cada paquete según unos valores preestablecidos. La interpretación concreta de esta cabecera ha ido cambiando con los años. La §22 de [13] hace un recorrido histórico sobre las diferentes interpretaciones de este campo.

El listado 2.5 muestra un ejemplo del sistema de prioridades de consulta de las diferentes tablas de rutas. En general, un usuario o administrador sólo ha de preocuparse de una única tabla, la tabla de rutas `main`, el resto se reservan para realizar tareas de encaminamiento avanzado. El conjunto de tablas de rutas creadas al arrancar la máquina se extraen de `/etc/iproute2/rt_tables`:

```
1 ; cat /etc/iproute2/rt_tables
2 255     local
3 254     main
4 253     default
5 0       unspec
```

Listado 2.6: Ejemplo del contenido de `/etc/iproute2/rt_tables`, donde se definen las tablas de rutas a utilizar en el arranque del dispositivo.

La ruta `unspec` ó `0` (definida en la línea 5 del listado 2.6), representa al conjunto de todas las demás, por ejemplo, se puede mostrar la totalidad de las reglas de todas las tablas de rutas mediante la invocación:

```
1  ; ip route show table 0
2  192.168.1.0/24 dev wifi proto kernel scope link src 192.168.1.120
3  default via 192.168.1.1 dev wifi
4  broadcast 192.168.1.0 dev wifi table local proto kernel scope link src 192.168.1.120
5  broadcast 127.255.255.255 dev lo table local proto kernel scope link src 127.0.0.1
6  broadcast 192.168.1.255 dev wifi table local proto kernel scope link src 192.168.1.120
7  local 192.168.1.120 dev wifi table local proto kernel scope host src 192.168.1.120
8  broadcast 127.0.0.0 dev lo table local proto kernel scope link src 127.0.0.1
9  local 127.0.0.1 dev lo table local proto kernel scope host src 127.0.0.1
10 local 127.0.0.0/8 dev lo table local proto kernel scope host src 127.0.0.1
11 fe80::/64 dev wifi proto kernel metric 256 mtu 1500 advmss 1440 hoplimit 4294967295
12 unreachable default dev lo table unspec proto none metric -1 error -101 hoplimit 255
13 local ::1 via :: dev lo table local proto none metric 0 mtu 16436 advmss 16376 hoplimit 4294967295
14 local fe80::216:6fff:fe65:d156 via :: dev lo table local proto none metric 0 mtu 16436 advmss
15   16376 hoplimit 4294967295
16 ff00::/8 dev wifi table local metric 256 mtu 1500 advmss 1440 hoplimit 4294967295
17 unreachable default dev lo table unspec proto none metric -1 error -101 hoplimit 255
```

Listado 2.7: Mostrar todas las reglas de todas las tablas de rutas usando la tabla virtual 0.

Todas las maneras de utilizar estas tablas virtuales están orientados a utilizar el núcleo de Linux como un router tradicional y no permiten usar criterios más en consonancia con lo que sería deseable en dispositivos móviles de usuario multiconectados.

2.5 Conclusiones

Hemos comenzado este capítulo, dedicado al estado del arte, con una revisión de qué son los dispositivos multiconectados, sus ventajas y los problemas generales a los que nos enfrentamos cuando tratamos con ellos.

A pesar de que los dispositivos multiconectados de usuario son cada vez más comunes y de la abundancia de redes de acceso, resulta difícil sacarle verdadero provecho a estas circunstancias.

Un dispositivo de usuario con varias interfaces de red heterogéneas nos permitiría disponer de cobertura casi permanentemente, posibilitando sobrevivir a fallos de conexión por interfaces individuales o desplazarnos, sin perder conectividad, por áreas geográficas con coberturas de diferentes tecnologías de acceso.

Otra ventaja importante de esta multiconectividad es poder seleccionar la interfaz de red más apropiada para cada una de nuestras comunicaciones según las necesidades de las aplicaciones, las condiciones externas y las capacidades de las redes de acceso disponibles.

Proporcionar este tipo de funcionalidades básicas son nuestro principal objetivo al elaborar esta tesis.

Una vez seamos capaces de disfrutar de estas funcionalidades básicas, se puede ir más allá, implementando sistemas para la distribución de tráficos individuales entre varias interfaces simultáneamente o la utilización de varios enlaces de forma redundante. Estos aspectos más complejos de la multiconectividad han quedado fuera del objetivo de nuestras investigaciones.

En nuestra opinión, los problemas fundamentales para aprovechar la multiconectividad de dispositivos de usuario se encuentran claramente localizados:

- Por un lado los protocolos de comunicaciones más utilizados no ofrecen un buen soporte de la movilidad para las aplicaciones de usuario más interactivas.
- Por otro lado, la ausencia de servicios en el sistema operativo para gestionar nuestras interfaces de forma automática y eficiente obstaculiza el desarrollo de soluciones generales y promueve el uso de soluciones diseñadas para casos específicos, de difícil aprovechamiento en escenarios más genéricos.

En cuanto a lo que a movilidad se refiere, a lo largo de §2.2 hemos introducido la nomenclatura básica de movilidad, dando una visión general del tipo de escenarios a los que nos enfrentamos y las estrategias más comunes para afrontarlos.

También hemos revisado algunos de los protocolos más comunes diseñados para solventar el problema de la movilidad, a diferentes niveles de la torre de protocolos y que son representativos de las estrategias más comunes. Hemos indicado sus ventajas e inconvenientes en lo que respecta a su aplicación a dispositivos multiconectados y hemos identificado algunas características que nos resultan favorables para este problema en particular, a saber:

- Preferimos protocolos capaces de realizar traspasos a nivel 3, de forma que nuestras soluciones sean independientes de las tecnologías de acceso.
- No resulta recomendable la dependencia de una infraestructura de movilidad externa a nuestro dispositivo, ya que nos orientamos a soluciones genéricas de aplicabilidad global.
- Teniendo en cuenta la creciente disponibilidad de redes de acceso, resulta razonable centrarnos en protocolos capaces de implementar traspasos make-before-brake, que permiten realizar traspasos imperceptibles más fácilmente, aunque es importante no limitarse a ellos, dejando siempre la posibilidad de utilizar traspasos brake-before-make cuando no quede más remedio.

- Aunque resulte óptimo realizar traspasos horizontales siempre que sea posible, lo habitual será enfrentarnos a traspasos verticales, por lo que debemos focalizar nuestro trabajo en traspasos entre tecnologías de acceso heterogéneas.

- Es importante evitar la rotura de conexiones a nivel de transporte para evitar involucrar a las aplicaciones o a una capa de sesión en nuestras soluciones de movilidad.

- Se debe procurar disminuir la duración de los traspasos para que el impacto en la calidad de servicio percibida por los usuarios sea el menor posible.

- Es recomendable evitar soluciones que provoquen ineficiencias en el encaminamiento, siendo ideal que las rutas antes y después de un traspaso sean las mismas que se utilizarían, bajo las mismas circunstancias, en ausencia de movilidad.

- En escenarios de movilidad no es viable confiar en soluciones de localización tradicionales como DNS, y se deben utilizar soluciones de localización dinámica. Algunos protocolos de movilidad no implementen su propia solución de localización, pues es necesario cierta dependencia de infraestructura adicional para ello. Estos protocolo deben ser complementados con una solución independiente de localización. Dado que la naturaleza técnica de los retos que plantea la localización dinámica se aleja bastante de los temas que tratamos en esta tesis, hemos decidido no abordar este problema y suponer que la localización se resuelve por mecanismos independientes a nuestras propuestas.

En lo que respecta a cómo se gestiona la multiconectividad desde el sistema operativo, en §2.3 hemos revisado la manera en que los sistemas operativos exponen las interfaces de red del dispositivo y cómo se resuelve el problema del encaminamiento de las comunicaciones cuando existen varias interfaces viables.

El problema principal en este sentido es que ni las APIs de sockets, ni la tabla de rutas, tienen el potencial suficiente para sacarle provecho a la multiconectividad de los dispositivos de usuario modernos.

En los próximos capítulos describimos las contribuciones, que tanto en el campo de los protocolos de movilidad como en el de los servicios del sistema operativo, proponemos para mejorar el aprovechamiento de la multiconectividad de dispositivos de usuario.

Parte II

Contribuciones a nivel de protocolo

Capítulo 3

SCTP como protocolo de movilidad

A pesar de que este capítulo está incluido en una de las partes dedica a nuestras contribuciones, el único contenido original de este capítulo es §3.3.2, donde interpretamos el mecanismo de detección de fallo de ruta bajo una nueva perspectiva muy personal, que ha condicionado el diseño de las propuestas que presentaremos en los dos siguientes capítulos.

El resto del capítulo consiste en una explicación de cómo funciona SCTP en situaciones de movilidad, con el detalle suficiente para que los capítulos posteriores puedan entenderse con facilidad.

3.1 ¿Por qué SCTP?

La elección de SCTP como protocolo de movilidad precede a la decisión de elaborar esta tesis doctoral y data de hace 7 años, cuando empezamos a trabajar en el proyecto de investigación EW.

El objetivo del proyecto EW era mejorar el soporte de traspasos imperceptibles para dispositivos móviles de usuario entre redes heterogéneas. El proyecto no estaba centrado en ninguna tecnología de movilidad concreta e incluso se afrontó desde el punto de vista de comparar soluciones para diversas tecnologías.

Durante las primeras fases del proyecto, realizamos una evaluación de las diferentes tecnologías de movilidad y desde el primer momento SCTP destacó como nuestra tecnología ideal:

- Es un protocolo de la capa de transporte, que es una de las especialidades de nuestro grupo de trabajo, tanto en nuestra faceta investigadora como docente.
- Cuando comenzamos la tesis, SCTP empezaba a ganar importancia. Hacía poco que se había incluido una implementación bastante completa en el

núcleo de Linux y era un tema de constantes referencias en las listas de correo del TSVWG[1]. De hecho, ese grupo de trabajo está preparando un borrador ([54]) para solucionar uno de los problemas principales que abordamos en nuestras investigaciones: acelerar la detección de fallo de ruta, del que luego hablaremos con más detalle en §3.3.3.

- Los traspasos iniciados a petición de la aplicación se realizan en SCTP sin retardo alguno, lo que nos parece una ventaja fundamental sobre otros protocolos de movilidad que gozan de mayor popularidad, como Mobile IP.

- La aproximación general de SCTP nos pareció elegante, y su ubicación en la pila de protocolos a nivel de transporte nos resulta de las más apropiada para solucionar el problema de la movilidad. Nuestro grupo de trabajo tiene experiencia a nivel de aplicación, por lo que adaptar aplicaciones existentes o crear nuevas para este protocolo no resulta un problema para nosotros.

- Al contrario que otros protocolos de movilidad, como Mobile IP, todavía no cuenta con una extensa base de publicaciones, estudios y propuestas, lo que permite una entrada rápida y una fácil identificación de los problemas fundamentales.

Después de unos años trabajando y familiarizándonos con SCTP resultó evidente orientar una tesis doctoral a las contribuciones que estábamos realizando en este campo y desde entonces hemos estado utilizando SCTP en nuestras investigaciones.

3.2 Escenario básico de movilidad en SCTP

El comportamiento básico de SCTP en escenarios de movilidad se puede ilustrar con un sencillo ejemplo: tenemos un dispositivo móvil multiconectado, que mantiene una comunicación con un servidor, al que puede alcanzar por varias rutas posibles, una para cada una de las interfaces del dispositivo móvil, como se muestra en la figura 3.1. Probablemente estas rutas tendrán algunos tramos en común, en especial, según nos acercamos al servidor, que quizá ni siquiera sea multiconectado.

De las posibles rutas entre los extremos de una asociación, se escoge una, que se denomina *ruta primaria*. El resto de rutas se denominan *rutas alternativas*[2].

La ruta primaria es la que se utiliza para transmitir los datos en ausencia de problemas de red. Las rutas alternativas se reservan para ser utilizadas si la primaria presenta problemas de transmisión.

La aplicación puede solicitar un cambio de ruta primaria, lo que implicará un cambio en el camino que siguen los datos hasta el receptor, aún en ausencia

[1] IETF Transport Area Working Group, [53].

[2] En SCTP no se especifica qué criterios utilizar para escoger la ruta primaria de entre las disponibles.

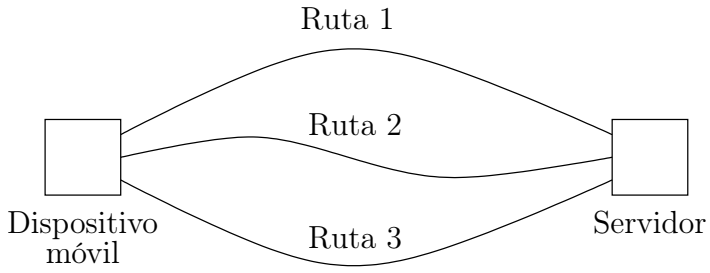

Figura 3.1: Ejemplo de escenario de movilidad en SCTP.

de problemas de red. Tanto si el cambio es automático, debido a problemas de red en la ruta primaria, como si es a petición de la aplicación, estamos ante un caso de traspaso.

Los traspasos que son consecuencia de una solicitud de la aplicación son prácticamente inmediatos, simplemente se pasa a utilizar una ruta diferente, y no es necesario notificarlo al otro extremo. Este tipo de traspasos no son el objetivo de esta tesis y el estudio de cómo las aplicaciones pueden optimizar la transmisión de los datos haciendo una gestión directa de las rutas queda fuera nuestro campo de estudio.

Los traspasos que son consecuencia de la detección de un fallo de ruta (de ahora en adelante PFD, del inglés path failure detection) son nuestra principal preocupación.

Por un lado, estos traspasos por PFD resultan muy interesantes para hacer una gestión de la movilidad transparente a las aplicaciones, que en la mayor parte de los casos desean mantenerse al margen de los detalles de encaminamiento de sus datos y de la movilidad.

Por otro lado, el tiempo necesario para detectar un fallo de ruta se suma directamente a la duración del traspaso, y si este tiempo es demasiado alto para una aplicación, puede resultar inviable utilizar SCTP como solución de movilidad.

3.3 Descripción detallada de un traspaso por PFD

Cuando se pide a SCTP que se transmita un mensaje por una asociación, lo primero que se hace es partirlo en *trozos*[3] o segmentos de un tamaño adecuado para el protocolo de red subyacente (IPv4 o IPv6). El tamaño de estos segmentos se conoce como PMTU, del inglés path maximum transfer unit y se obtiene utilizando los métodos descritos en [55, 56, 57].

A cada uno de estos trozos se le asigna un *número de secuencia de transmisión* (TSN). El receptor ordena los trozos recibidos según este TSN y los ensambla de nuevo para obtener una copia exacta del mensaje original.

[3]Traducción directa del término técnico chunk en inglés.

SCTP es un protocolo fiable, en el sentido de que si alguno de los segmentos se pierde por el camino hasta el receptor, será retransmitido por el emisor poco tiempo después, de forma que se asegure que el destinatario recibe al menos una copia de cada segmento enviado. Esto quiere decir que el emisor tiene que averiguar si algún segmento se ha perdido, y en ese caso, tendrá que retransmitirlo.

Para saber cuándo un segmento se ha perdido, todos los segmentos recibidos por el destinatario han de ser asentidos, mediante la transmisión de un segmento de asentimiento (ACK) de vuelta al emisor original de los datos.

Si un emisor no recibe el asentimiento a alguno de los segmentos de datos enviados en un periodo de tiempo determinado, se asume que el segmento de datos se ha perdido y se planifica su reenvío en un tiempo corto.

Esto sucederá tanto si se perdió el segmento de datos original, como si se perdió su asentimiento en el camino de vuelta o si simplemente alguno de los dos se ha retrasado más de lo esperado. Un fallo de ruta, por ejemplo, puede ser la causa de estas pérdidas.

Los receptores SCTP deben asentir cualquier segmento de datos recibido inmediatamente[4]. El tiempo que tarda un segmento de datos en llegar hasta el receptor y su asentimiento en volver al emisor se conoce como el *round trip time* de la ruta o RTT.

Generalmente el RTT cambiará con el tiempo a lo largo del tiempo de vida de una asociación, según los elementos transmisores de la ruta reaccionen a la diferente carga de tráfico que deben procesar en cada momento.

Un emisor SCTP estima el RTT de cada una de las rutas a su destino manteniéndose al tanto de cuánto tiempo pasa entre que envía un segmento por una ruta y le llega su correspondiente asentimiento. Estas medias se promedian en el tiempo, siguiendo la metodología original de TCP, que se explica en [59, 60]. Para eliminar las posibles ambigüedades derivadas de los asentimientos a retransmisiones, se ignoran las medidas de RTT de segmentos retransmitidos, según el el algoritmo de Karn/Partrige (ver [61] para más detalles).

3.3.1 Retransmisiones

El temporizador de retransmisión, o RTO (por sus siglas en inglés, retransmission time-out), se utiliza por un emisor SCTP para saber cuándo darse por vencido al esperar el asentimiento a un segmento recién enviado. Su funcionamiento se describe con detalle en [62].

El valor de RTO se calcula a partir del RTT estimado y el temporizador se reinicia en el momento de mandar un segmento de datos, de forma que si no llega su asentimiento antes de que expire, se considera que el segmento se ha perdido y se planifica su retransmisión[5]. Al igual que en TCP, los valores

[4] Aunque por eficiencia se suele esperar un tiempo corto antes de asentir un segmento de datos, ver sección 4.2 de [58].

[5] La mayoría de las implementaciones de protocolos que utilizan este tipo de temporizadores simplifican este proceso para evitar mantener un temporizador para cada segmento en curso. Una aproximación muy popular es mantener un único RTO para todos los segmentos en curso, reiniciándolo cada vez que se envía un nuevo segmento.

mínimos y máximos de RTO se limitan a unos valores prefijados por omisión a $\mathrm{RTO}_{\min} = 1$ segundo y $\mathrm{RTO}_{\max} = 60$ segundos.

Las retransmisiones se envían por rutas alternativas, en vez de por la misma ruta por la que se envió el segmento de datos original. Con esta medida se pretende aumentar las posibilidades de que la retransmisión evite el posible fallo de ruta o congestión que ha podido ocasionar la pérdida o retraso del segmento original. El éxito de esta medida depende de la topología de las rutas original y alternativas; lo ideal sería tener rutas con pocos tramos en común, aunque SCTP no dispone de ningún mecanismo para fomentar esto.

Como el receptor debe ordenar los segmentos de datos recibidos antes de pasarlos a la aplicación, el retraso de un solo segmento de datos fuerza al receptor a encolar todos los segmentos con un TSN posterior, hasta que el segmento retrasado finalmente es recibido.

En estos casos, para evitar acumular demasiados segmentos en el receptor, los emisores dan cierta preferencia a las retransmisiones sobre las transmisiones de nuevos datos. Esto significa que cuando un emisor tiene tanto nuevos datos para transmitir, como algunas retransmisiones planificadas, las transmisiones de nuevos datos se retrasarán en favor de algunas de las retransmisiones.

Ajuste del RTO por retransmisión

Cuando un temporizador de retransmisión expira, el RTO de la ruta se incrementa *artificialmente*, antes de retransmitir los datos por una ruta alternativa. Esta estrategia se conoce como *backoff exponencial binario* y es de vital importancia para mantener la estabilidad de la red en casos de congestión ([59]).

Esto significa que cada ráfaga sucesiva de retransmisiones se retrasará el doble de tiempo que la anterior, según el valor del RTO se va duplicando cada vez que expira. Si finalmente se recibe un asentimiento, se asume que el problema que afectaba a la ruta ha desaparecido: el backoff exponencial binario se interrumpe y el valor del RTO se fija de nuevo al valor del RTT estimado.

En caso de que la ruta no se recupere del fallo en un tiempo razonable, estaremos ante un fallo de ruta permanente y se debería dejar de usar esa ruta y pasar a mandar las transmisiones de nuevos datos por una ruta alternativa.

SCTP decide que se encuentra ante un fallo de ruta permanente cuando se alcanza cierta cantidad umbral de retransmisiones consecutivas, es decir cuando después de un número determinado de retransmisiones, sigue sin recibirse confirmación de su recepción.

Ajuste de la ventana de congestión por retransmisión

Cuando un temporizador de retransmisión expira, se asigna un valor a la ventana de congestión no superior a un PMTU ([52]). Esto quiere decir que cuando sea necesaria una retransmisión, la cantidad de datos que se pueden enviar a la ruta por la que se perdió el segmento original, se limita a un único segmento como mucho.

Este valor que se le asigna a la ventana de congestión al detectarse una retransmisión se conoce como *ventana de pérdidas*, del inglés loss window.

Conforme vayan llegando asentimientos, el valor de la ventana de congestión se incrementa en un segmento con cada segmento asentido, mediante el algoritmo conocido como slow-start ([52]), con la idea de sondear nuevamente la capacidad de la red.

Llegados a un punto en que se sospecha que estamos próximos a la capacidad disponible, se ralentiza aún más el crecimiento de la ventana de congestión, hasta un crecimiento lineal con cada RTT, según el algoritmo de congestion avoidance, con el mismo propósito de intentar utilizar el máximo de la capacidad disponible sin llegar a congestionar la red en exceso.

En resumen: el vencimiento de un temporizador de retransmisión tiene como consecuencia que se reduzca al mínimo la velocidad con la que se envían datos a la ruta en cuestión, y que se vaya incrementando esta velocidad progresivamente si no son necesarias más retransmisiones.

El razonamiento tras este comportamiento es que en SCTP (y en TCP) se atribuye la pérdida de un segmento a una ruta congestionada, por lo que se reacciona en consecuencia, reduciendo bruscamente el envío de datos por la ruta en cuestión para aliviar la congestión y aumentando progresivamente la tasa de envío de datos si parece que la congestión ha desaparecido.

Fallos de ruta permanentes

Cuando nos encontramos ante un caso de fallo de ruta permanente, todos los segmentos de datos y/o sus asentimientos se perderán.

En este caso, no sólo es necesario realizar las retransmisiones correspondientes de los segmentos no asentidos, será conveniente también realizar un traspaso a una ruta alternativa de forma que las nuevas transmisiones puedan evitar el fallo de red que afecta a la ruta primaria.

Para ello, cuando SCTP está seguro de que se encuentra ante un fallo de ruta permanente, se designa una de las rutas alternativas como la nueva ruta primaria y se desactiva la ruta primaria original. Las posibilidades de evitar el fallo de ruta mediante esta estrategia dependen directamente de cuan diferentes son las rutas entre si.

Evidentemente, la pérdida de un solo segmento de datos o de su asentimiento no es un identificador definitivo de un fallo de ruta permanente. El segmento original o su asentimiento pueden haberse perdido o retrasado por un router temporalmente sobrecargado o quizá uno de los tramos de la ruta está sufriendo un periodo de interferencias, del que se recuperará en breve.

Para diferenciar entre este tipo de problemas temporales en la transmisión y un fallo de ruta permanente, SCTP cuenta el número de retransmisiones *sucesivas* que está realizando, hasta que llega a un determinado umbral, conocido como `Path.Max.Retrans` o PMR. El valor por omisión de PMR es 5, es decir, cuando es necesaria una sexta retransmisión consecutiva al enviar datos por una ruta, SCTP asume que el fallo de ruta es permanente.

Este sencillo mecanismo resulta conveniente para el programador de una pila SCTP, sin embargo, resulta poco flexible para las aplicaciones, ya que cada aplicación tiene sus propias ideas acerca de qué se debe considerar como un fallo

de ruta permanente.

Cuando una pila SCTP detecta un fallo de ruta permanente, se realiza el traspaso a una de las rutas alternativas, los nuevos segmentos de datos se transmiten por una nueva ruta primaria y si esto evita el punto de fallo de red, la transmisión de los datos podrá completarse normalmente, sin necesidad de intervención por parte del usuario ni la aplicación.

La siguiente desigualdad establece un límite inferior a la duración de la detección de fallo de ruta, T_{PFD}:

$$T_{PFD} > \sum_{i=0}^{PMR} \text{mín}\left(RTO_{max}, 2^i * RTO_{fail}\right) \tag{3.1}$$

La desigualdad (3.1) sólo considera los mecanismos más relevantes que afectan a T_{PFD}, que son los que hemos explicado en los párrafos anteriores[6].

RTO_{fail} es el valor de RTO en el momento del fallo de ruta. Evaluando (3.1) para los valores por omisión de RTO_{min}, RTO_{max} y PMR (1 segundo, 60 segundos y 5 retransmisiones respectivamente), obtenemos $T_{PFD} > 63$ segundos para un $RTO_{fail} = RTO_{min}$ y $T_{PFD} > 6$ minutos para $RTO_{fail} = RTO_{max}$.

La figura 3.2, en la página siguiente, muestra los TSNs de los segmentos de datos transmitidos por un emisor en el caso de un fallo de ruta permanente. Los datos han sido recopilados utilizando la herramienta de simulación OMNeT++[7].

El emisor y el receptor de la simulación están conectados mediante dos rutas redundantes. El emisor quiere transmitir 200 segmentos de datos al receptor a una velocidad similar al ancho de banda disponible entre ellos por cualquiera de las rutas. En el segundo 3 de la simulación la ruta primaria sufre un fallo de ruta permanente que evita que se transmitan por ella tanto los segmentos de datos como sus asentimientos.

Entre los segundos 0 y 3, el emisor está transmitiendo los datos normalmente por la ruta primaria y los asentimientos a estos datos (que no se muestran en la figura) le están llegando con regularidad.

En el segundo 3, la ruta primaria sufre un fallo de ruta permanente, lo que evitará que se transmitan paquetes en ambos sentidos por esa ruta. El RTO en este momento es aproximadamente 2 segundos, por lo tanto, el receptor dejará de recibir segmentos de datos enviados posteriormente al segundo 3 y el emisor dejará de recibir los asentimientos a los segmentos mandados hace 2 segundos, es decir, los que corresponden a segmentos de datos enviados posteriormente al segundo 1, que en la figura son aquellos segmentos de datos con un TSN superior a aproximadamente 60.

Es decir, no se recibirán los asentimientos de los TSN mayores que 60 y tendrán que ser retransmitidos.

El emisor esperará los asentimientos a esos TSNs durante un periodo de RTO segundos (2 segundos) a partir del último envío de datos (poco después del segundo 3, lo que tarda la ventana de congestión en agotarse).

[6] En [63] se detalla la ecuación exacta para T_{PFD}.

[7] En concreto, hemos utilizado la versión 4.1 de OMNeT++ y la versión del 10 de junio del 2011 del paquete INET.

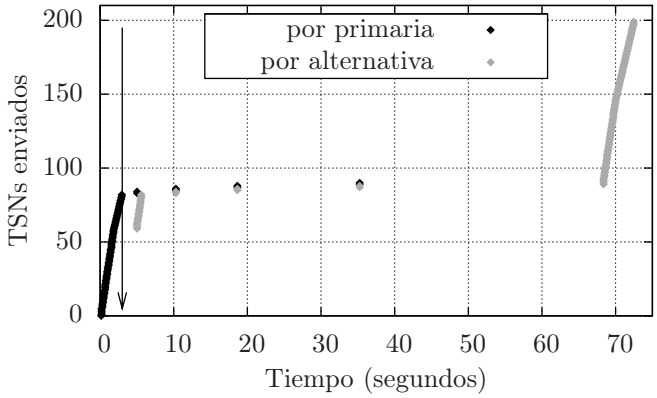

Figura 3.2: Un ejemplo de los TSNs transmitidos por un emisor SCTP. El origen quiere mandar 200 segmentos de datos al destino, a una velocidad similar al ancho de banda disponible. Emisor y receptor están conectados por dos rutas redundantes y completamente independientes, de similares características de transmisión. Los segmentos enviados por la ruta primaria se muestran como puntos negros y los enviados por la ruta alternativa se muestran como puntos grises. La ruta primaria sufre un fallo de ruta permanente en el segundo 3, como sugiere la flecha vertical hacia abajo. Hemos reducido artificialmente a 4 el PMR para mejorar la legibilidad.

Aproximadamente en el segundo 5, el RTO expira. El valor del RTO se multiplica por 2, según la estrategia de backoff exponencial binario, por lo que queda en un valor de 4 segundos. Las retransmisiones pendientes (TSNs de más de 60) son enviadas por la ruta alternativa y algunas nuevas transmisiones se intentan enviar por la ruta primaria, aunque pocas, ya que ahora mismo la ventana de congestión de la ruta primaria tiene el valor mínimo de la ventana de pérdidas.

Los asentimientos a los mensajes retransmitidos serán recibidos por el emisor normalmente por la ruta alternativa (tampoco se muestran en la figura), pero los asentimientos a los nuevos datos transmitidos por la ruta primaria no llegarán nunca, debido al fallo de ruta permanente.

Por lo tanto, al cabo de aproximadamente 4 segundos (valor actual del RTO en la ruta primaria), el temporizador de retransmisión volverá a expirar, y se duplica de nuevo su valor hasta 8 segundos. Las retransmisiones pendientes se envían por la ruta alternativa y algunas nuevas transmisiones vuelven a intentarse por la primaria.

En esta simulación redujimos artificialmente el PMR del emisor a 4, para mantener la figura legible. Consecuentemente, el proceso de backoff exponencial binario irá duplicando el RTO antes de intentar las nuevas transmisiones de datos hasta que la quinta retransmisión consecutiva es necesaria, aproximadamente en el segundo 69. No se preocupe si los tiempos no encajan exactamente con lo explicado anteriormente, existen mecanismos adicionales a los que hemos explicado que modifican ligeramente la planificación de los envíos y retransmi-

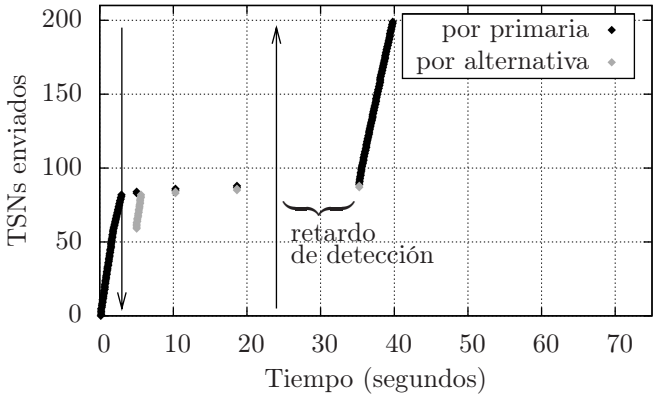

Figura 3.3: TSNs enviados por el emisor. La misma situación que en la figura 3.2 pero ahora la ruta primaria se recupera del fallo de ruta en el segundo 24, como sugiere la flecha hacia arriba. El eje de abscisas de esta figura se ha mantenido al mismo tamaño y escala que el de la figura 3.2 para facilitar las comparaciones entre ambas.

siones y que hemos omitido por motivos pedagógicos. Consulte [63] para un análisis completo de los mecanismos involucrados en la planificación.

A partir del segundo 69, habiéndose planificado la quinta retransmisión, SCTP asume que la ruta primaria ha sufrido un fallo de ruta permanente, por lo que esta ruta se desactiva y se designa la ruta alternativa como la nueva ruta primaria, por lo que se realiza un traspaso. A partir de ahora todas las nuevas transmisiones serán realizadas por esta nueva ruta primaria, libre de fallos de red y serán recibidas sin problemas por el destino.

SCTP ha sobrevivido al fallo de ruta y, en el segundo 72, la transmisión de los 200 segmentos de datos concluye con éxito. SCTP ha sufrido un retraso en el envío de los datos de unos 66 segundos aproximadamente, mientras trataba de decidir si los problemas de transmisión que experimentaba eran un problema temporal o un fallo de ruta permanente.

Fallos de ruta temporales

La figura 3.3 muestra un ejemplo del comportamiento de SCTP cuando la ruta primaria sufre un fallo de ruta temporal, de 21 segundos de duración.

En este caso, los nuevos segmentos de datos enviados por la ruta primaria después de la cuarta retransmisión, llegarán al receptor sin problemas por la ruta primaria y sus asentimientos también llegarán al emisor sin problemas por esta misma ruta. El contador de retransmisiones se reinicia al llegar el primer asentimiento después del segundo 24, y la comunicación puede continuar por la ruta primaria sin mayores problemas.

SCTP ha podido detectar la recuperación de la ruta primaria y ha conseguido terminar la transmisión de los 200 segmentos de datos, en su mayoría usando la ruta primaria. El *retardo de detección* es el tiempo que ha pasado entre que la ruta se recupera del fallo (en el segundo 24) y el reestablecimiento de las

transmisiones por esa misma ruta (en el segundo 36). Por tanto el retardo de detección en este ejemplo es de aproximadamente 12 segundos.

Intuitivamente, el retardo de detección puede reducirse, por ejemplo, si se intentaran transmisiones de datos por la ruta primaria más a menudo durante la detección del fallo de ruta.

Si comparamos el comportamiento de SCTP ante un fallo de ruta permanente (figura 3.2) y su comportamiento ante un fallo de ruta temporal (figura 3.3) observamos que en ambos casos SCTP es capaz de terminar la transmisión de los datos satisfactoriamente, y que el retraso sufrido en el caso de un fallo de ruta temporal será menor o igual que en el caso de un fallo de ruta permanente, como es razonable.

Reducir el retraso sufrido, tanto en situaciones de fallo permanente como de fallos temporales es una de las principales contribuciones de esta tesis. En §4 proponemos una modificación a SCTP para reducir estos retrasos y en §5 demostramos por qué la estrategia propuesta es óptima para fallos de ruta temporales.

3.3.2 PFD interpretado como mecanismo de sondeo

Una comparación entre la figura §3.2 y la §3.3 sugiere que el mecanismo de PFD de SCTP es *un mecanismo de sondeo de la recuperación de la ruta primaria, mientras intentan descartarse otras posibles causas de problemas*, como una estimación a la baja del RTT o eventuales pérdidas en la transmisión.

Esta interpretación del PFD como mecanismo de sondeo de la recuperación de la ruta se hace evidente desde la siguiente perspectiva: cuando expira por primera vez un RTO, SCTP sospecha que un fallo de ruta permanente ha ocurrido, pero no tiene evidencias para confirmarlo. Se envían algunos segmentos más de datos para sondear la ruta, a intervalos cada vez mayores, para aliviar la posible congestión y si no se obtiene ningún asentimiento después de unos cuantos intentos, se asume que nos encontramos ante un fallo de ruta permanente.

En este sentido, el valor de PMR representa el número de sondéos a realizar, antes de darnos por vencidos y declarar la ruta principal como inactiva. Un valor alto de PMR permite realizar un sondeo más fiable, ya que se reduce la posibilidad de falsos positivos en la detección del fallo de ruta, pero alarga su duración en proporción geométrica.

Efectivamente, la planificación temporal de las sondas es geométrica, ya que las ráfagas de transmisiones que actúan como sondas deben esperar a las retransmisiones pendientes, que a su vez, se retrasan un periodo de tiempo cada vez mayor debido al mecanismo de backoff exponencial binario.

Esta interpretación personal es el punto de partida de nuestras propuestas para agilizar la detección de fallos de ruta permanentes y la recuperación ante fallos temporales, que se explican mas adelante en §4 y §5.

3.3.3 Reducción de la duración del PFD

Como hemos explicado en §3.3, la detección de fallo de ruta en SCTP dura aproximadamente entre 1 y 6 minutos cuando se utilizan los valores por omisión de los parámetros del protocolo. Esta duración resulta demasiado larga para lo que algunas aplicaciones necesitan y se han propuesto diversas soluciones al respecto.

Algunas de estas soluciones pasan por ajustar los valores de los parámetros de SCTP que influyen en la duración del PFD:

- **Reducir el valor de PMR**: se aprecia en la desigualdad (3.1) que el valor de PMR afecta de forma significativa a la duración del PFD.

  En [64] se afirma que para tasas de pérdidas entre el 1 % y el 8 %, un valor de PMR entre 0 y 5 es suficiente para detectar un fallo de ruta. También se muestra cómo un valor de PMR de 0 mejora el goodput con respecto a valores de PMR mayores que 0. Los autores encontraron que los traspasos espurios no degradan la eficiencia en dichas circunstancias.

  Sus conclusiones muestran que reducir el PMR es un método simple y eficaz para acelerar el mecanismo de PFD cuando los traspasos frecuentes no sean un problema, por ejemplo en escenarios en que se puede contar con rutas redundantes, diseñadas, desplegadas y controladas con el propósito de funcionar como alternativas de las rutas principales, por ejemplo en redes militares o en compañías con su propia infraestructura replicada.

  En redes con tasas de error más altas o ráfagas de errores, como en Wi-Fi, o en escenarios donde existe una ruta preferente, por precio o calidad de servicio, este mecanismo de reducir el valor de PMR no resulta tan atractivo.

- **Reducir el retraso de los asentimientos**: Los asentimientos en SCTP no se mandan inmediatamente, se retrasan y acumulan para formar *asentimientos selectivos* o SACK, tratando de reducir la cantidad de tráfico de control generada por el protocolo.

  La influencia del retraso de los SACKs en el PFD no se muestra en (3.1), pero puede ser calculada de las ecuaciones proporcionadas en [63]. En [65] se estudia el impacto de reducir el retraso de los SACKs en el PFD. Sus conclusiones son que el efecto es mayor en redes lentas. Su análisis de mejor caso, utilizando un retraso nulo para los SACKs, muestra que el tiempo de PFD se reduce en menos de un 10 % mientras que el tráfico de control generado por los SACKs se duplica. Los mismos autores proporcionan estudios más detallados sobre este mismo tema en [66].

- **Reducir $\mathbf{RTO_{min}}$** : La reducción del valor mínimo de RTO puede acelerar la detección del fallo de ruta drásticamente cuando el RTT está por debajo del valor por omisión de $\mathrm{RTO_{min}}$. Nótese como $\mathrm{RTO_{min}}$ fija una duración mínima a cada elemento del sumatorio en la desigualdad (3.1).

En [54] no se recomienda modificar los valores de estos parámetros si no se tiene suficiente conocimiento sobre las características de transmisión de las rutas. En [67] se citan algunos escenarios en los que una modificación de estos parámetros puede resultar inofensiva y por tanto interesante para reducir la duración del PFD.

Otras soluciones requieren modificar el normal comportamiento de SCTP:

- **Latidos después de la primera transmisión**: En [64] se analizan diversos algoritmos de retransmisión. Sus experimentos demuestran que mandar un latido justo después de la primera retransmisión reduce la duración del PFD aproximadamente a la mitad, ya que ese latido, si no es asentido, cuenta como una pérdida más en el contador de retransmisiones. En cierto sentido, el efecto de esta sonda sería similar a reducir en una unidad el PMR, pero manteniendo el número de sondas original.

  Ese latido adicional reduce la duración de la detección de fallo de ruta a aproximadamente la mitad, sin embargo, cuando hay nuevas transmisiones pendientes, se manda prácticamente al mismo tiempo que la siguiente transmisión por lo que resulta redundante.

- **Usar la ruta con el menor RTO**: Los autores de [68] modifican el mecanismo de PFD para que siempre se elija como dirección primaria la de menor RTO. La propuesta es novedosa y sus experimentos producen resultados emocionantes. Sin embargo, esta aproximación no encaja en escenarios en los que existe una ruta preferente por razones externas al protocolo, ya que todas las rutas gozan de la misma categoría y se cambia de una a otra según sus características de transmisión en cada momento.

- **Transferencia multiruta simultánea**: (en inglés Concurrent Multipath Transfer). En [69] se proponen soluciones para utilizar SCTP en escenarios donde los datos pueden viajar simultáneamente por varias rutas, lo que ayuda a reducir los retrasos de transmisión introducidos por un fallo de ruta gracias a que los datos están llegando al receptor por varias rutas simultáneamente. La aplicabilidad de esta solución se limita a escenarios donde se puede asumir el coste de utilizar redundantemente las rutas al receptor.

- **Introducir un nuevo estado de fallo potencial**: A lo largo de los últimos dos años, se ha discutido activamente sobre nuevos mecanismos para evitar los inconvenientes de un PFD lento, tanto en la lista de correo electrónico del TSVWG como durante las reuniones presenciales de dicho grupo[8].

  Como consecuencia de ello, se está escribiendo un nuevo borrador sobre este tema ([54]). En él, los autores proponen la introducción de un nuevo

[8]Nuestra participación en estas discusiones se ha limitado a la lista de correo, no hemos asistido a las reuniones presenciales.

estado en SCTP, denominado *estado de fallo potencial*, o PF, que es un nuevo estado intermedio para las rutas, a medio camino entre los estados de "activo" e "inactivo".

El estado PF indica que una ruta sigue activa pero puede volverse inactiva pronto. Mientras una ruta está en estado PF, las transmisiones se envían a rutas alternativas y se recomienda vigilar el estado de la ruta en estado PF para decantarse finalmente por volverla inactiva o recuperarla al estado activo.

Durante la elaboración de esta tesis hemos mantenido el contacto con el TSVWG, proponiendo mejoras al borrador e intentando colaborar para que se incluyese nuestra propuesta (ver capítulo §4) en él. Sin embargo, nuestros esfuerzos de colaboración han sido rechazados sin explicaciones.

Es importante darse cuenta que la propuesta del nuevo estado PF no va dirigida a acelerar la detección de fallo de ruta, sino más bien a utilizar las rutas alternativas *antes* de la detección del fallo. Utilizar las rutas alternativas antes de la detección del fallo tiene los inconvenientes ya explicados, pero tiene sentido en escenarios preparados para ello, como los que se explican en la propuesta original ([70]) en la que se basa el borrador. Es posible que esta diferencia fundamental entre el contenido del borrador y nuestra propuesta haya sido la causa principal de que se hayan denegado nuestros intentos de colaboración en el borrador.

Todas estas propuestas han servido de inspiración para nuestro trabajo modificando SCTP (§4).

3.4 Identificación de los problemas de SCTP y un adelanto de nuestra propuesta

Creemos firmemente que el principal causante de la lentitud del mecanismo de detección de fallo de ruta en SCTP es la planificación geométrica de las sondas.

Nuestra propuesta (§4) consiste en reducir la duración del mecanismo de PFD mediante la utilización de una planificación temporal de las sondas diferente, configurable por la aplicación para ajustarla a sus necesidades y óptima en la detección de fallos temporales.

Nuestra propuesta se basa en dejar de utilizar las transmisiones de nuevos datos como sondas para la ruta y en su lugar utilizar latidos, de forma que desacoplamos el mecanismo de control de congestión del mecanismo de detección de fallo de ruta, lo que nos libera de utilizar sondas sujetas a una política de backoff exponencial binario. Con los ajustes adecuados en la ventana de pérdidas podemos asegurar la estabilidad de red de nuestra propuesta, evitando enviar más tráfico a la ruta en cuestión del que se enviaría en una implementación estándar de SCTP.

Capítulo 4

Propuesta de modificación de SCTP: SCTP-AP

4.1 Resumen

Cuando un dispositivo SCTP está enviado datos a un interlocutor multiconectado y la ruta primaria sufre un fallo de ruta, el mecanismo de detección de fallo de ruta (PFD) de SCTP automáticamente se hace cargo de la situación, desactivando la ruta primaria y permitiendo que la comunicación continúe por una de las rutas alternativas sin necesidad de intervención del usuario o la aplicación.

El tiempo que tarda una implementación estándar de SCTP en detectar este fallo de ruta es superior a 1 minuto en el mejor de los casos, lo que resulta excesivo para algunas aplicaciones en Internet.

Durante el proceso de detección de fallo de ruta, se observa:

- Retraso en las comunicaciones: ya que las transmisiones están siendo demoradas por los intentos de retransmisión.
- Baja tasa de envío: ya que la ventana de congestión de la ruta sobre la que se realiza el PFD se mantiene al valor mínimo de ventana de pérdidas por la falta de asentimientos a los datos enviados por la ruta fallida. Esto reduce la tasa de envío de las transmisiones por esa ruta, para satisfacer las restricciones de control de congestión.

  Las retransmisiones por la ruta alternativa tampoco contribuyen de manera significativa a aumentar la tasa de envío por la misma razón: se intentan muy pocas transmisiones durante el PFD, por lo que el número de retransmisiones es muy bajo.

Por tanto es importante reducir la duración del PFD para que afecte lo menos posible a la comunicación de los datos.

Creemos que la definición de qué es un fallo de ruta debe ser proporcionada por la aplicación. Por ejemplo, una interrupción de un minuto puede resultar inaceptable en un videojuego interactivo, pero puede no presentar mayor

problema para una aplicación que realice una copia de seguridad de datos periódicamente cada noche.

Las aplicaciones pueden reducir este tiempo de detección de fallo de ruta ajustando diversos parámetros del protocolo, pero desgraciadamente esto tiene repercusiones, a veces indeseables, en otros aspectos del comportamiento del protocolo (§3.3.3).

Por ejemplo, las aplicaciones pueden reducir el tiempo de detección de fallo de ruta modificando parámetros de control de congestión, como el valor de $\mathrm{RTO}_{\mathrm{min}}$, pero esto puede acarrear efectos indeseados en el propio mecanismo de control de congestión.

También se puede reducir el tiempo de detección reduciendo un parámetro propio del mecanismo de PFD, el PMR. Sin embargo esto aumenta la probabilidad de falsos positivos en la detección de fallos de red, lo que puede ser inaceptable. Otros intentos de reducir la duración de la detección de fallo de ruta conllevan compromisos similares.

En este capítulo, proponemos SCTP-AP, una modificación a SCTP que permite ajustar la duración de la detección de fallo de ruta, manteniendo una estrategia conservadora del mecanismo de control de congestión del protocolo y sin aumentar la probabilidad de falsos positivos en la detección.

SCTP-AP sondea la ruta bajo sospecha mediante latidos SCTP estándar. Este mecanismo de sondeo activo puede ser configurado por la aplicación para ajustarse a sus necesidades. *En general, si una aplicación define un fallo de ruta como la incapacidad de enviar datos durante un periodo determinado de tiempo, un mecanismo de PFD que tarde más que eso en detectar los fallos de ruta será inútil.*

En SCTP, las retransmisiones son lentas para satisfacer al mecanismo de control de congestión, sin embargo algunas aplicaciones necesitan detecciones de fallo de ruta rápidas. Para evitar este compromiso proponemos desacoplar ambos mecanismos, evitando sincronizar los sondeos de la ruta con las retransmisiones:

Proponemos un mecanismo de sondeo activo que utilice latidos SCTP para realizar detecciones de fallo de ruta rápidas, independientemente de la planificación temporal de las retransmisiones.

En §2.2.3 ya hicimos una breve introducción a SCTP y en §3 proporcionábamos una descripción en detalle del mencanismo de PFD en SCTP.

La idea inicial de nuestra propuesta fue publicada en [71]. En la siguiente sección presentamos una versión más completa y detallada del material publicado.

En §4.3 presentamos los resultados de nuestras simulaciones con SCTP-AP ante fallos de ruta de diferentes duraciones y en §4.6 los resultados de simulaciones ante flujos competidores.

4.2 Descripción del protocolo

La siguiente descripción del protocolo está escrita para facilitar su implementación y hemos utilizado un lenguaje y estilo similares al que se utiliza en la

especificación original de SCTP ([29]), por lo que puede resultar ininteligible para quien no esté familiarizado con ese documento en concreto. No pasa lo mismo con §4.2.1 y posteriores, que sí están dirigidas a un público más general.

Los siguientes párrafos describen un nuevo estado para las direcciones de destino SCTP, llamado *sondeo activo* o AP (del inglés active probing).

Una dirección de destino entrará en el estado AP si:

1. La dirección no está ya en el estado AP.
2. Y es una dirección activa.
3. Y la asociación está en el estado `ESTABLISHED`.
4. Y el temporizador `T3-rtx` (temporizador de retransmisión) para esa dirección expira.

Una dirección de destino saldrá del estado AP si:

1. Se obtiene una nueva medida del RTT. Esta manera de salir del estado AP se denomina *recuperación de ruta*. Esto sucede cuando se recibe un asentimiento por la ruta, con permiso del algoritmo de Karn.
2. O la dirección pasa a estar inactiva. Esta manera de salir del estado AP se denomina *retransmisión rápida*.
3. O la dirección primaria cambia a una dirección activa diferente a la que nos ocupa.
4. O la asociación entra en el estado `SHUTDOWN-ACK-SENT`.
5. O la asociación entra en el estado `CLOSED`.

Cuando una dirección de destino está en el estado AP, la pila SCTP-AP le mandará una cierta cantidad de latidos.

Los parámetros `ActiveProbing.Burst` y `ActiveProbing.GiveUp` son dos nuevos parámetros del protocolo que pueden ser configurados por el nivel superior, normalmente por la aplicación. El valor por defecto de `ActiveProbing.Burst` es PMR y el valor por defecto de `ActiveProbing.GiveUp` es $63 * \mathrm{RTO}_{\min}$, por las razones que se explican posteriormente en §4.2.1, en el apartado "Valores por omisión de los nuevos parámetros del protocolo".

El retardo entre cada latido (T_{HB}) tiene un valor de

$$T_{HB} = \frac{\texttt{ActiveProbing.GiveUp}}{\texttt{ActiveProbing.Burst}} * \mathrm{unif}(1, 2) \tag{4.1}$$

El primer latido se manda tan pronto como la dirección de destino entra en el estado AP.

Cuando una dirección de destino sale del estado AP, todas las transmisiones y retransmisiones pendientes se envían siguiendo la estrategia normal de SCTP.

Los latidos enviados como consecuencia de estar en el estado AP deben consumir octetos de la ventana de pérdidas, lo que reduce la transmisión de nuevos datos por la ruta.

La cantidad necesaria de octetos para enviar los latidos *debe ser reservada en la ventana de pérdidas al entrar en el estado AP* y no puede ser consumida por otros mecanismos del protocolo. Si al entrar en el estado AP, la ventana de pérdidas no tuviera los octetos necesarios para la reserva, la dirección no entrará en el estado AP.

Dado que los latidos son mensajes cortos, su eficiencia con respecto al tamaño de las cabeceras IP y SCTP es baja, por eso, a la hora de reservar espacio en la ventana de pérdidas ha de tenerse en cuenta la reserva de las cabeceras SCTP que envuelven a estos latidos, preservando el carácter TCP-Friendly de la comunicación ([72]).

4.2.1 Razonamientos

Latidos como sondas

Es cómodo reutilizar los latidos SCTP como sondas, ya que deben ser asentidos inmediatamente. Otro punto a su favor es que al no transportar ningún tipo de dato del nivel superior, no tienen por qué ser retransmitidos.

Normalmente los latidos ocupan pocos octetos. Su tamaño depende de la implementación, ya que su contenido no está especificado en las RFCs. El tamaño mínimo del trozo que transporta un latido son 8 octetos. En Linux 2.6.32, los trozos que transportan latidos ocupan 48 octetos, por lo que el datagrama IP que los transporta tendrá un payload de 60 octetos. Lo mismo se puede decir de los asentimientos a los latidos, ya que se limitan a transportar lo que quiera que contuviese el latido al que se refieren.

Para asegurar el principio de conservación de paquetes [59] se debe reservar suficientes octetos en la ventana de pérdidas como para mandar todos los latidos necesarios. Por tanto, latidos más pequeños tendrán un impacto menor en la ventana de congestión (mas información sobre este particular en §4.5).

En ausencia de tráfico de datos, SCTP envía latidos periódicamente para monitorizar el estado de las rutas al destino. Ésta es una aproximación similar a lo que SCTP-AP propone. De hecho, SCTP permite que la capa superior envíe latidos a placer, por lo que el comportamiento de SCTP-AP puede ser simulado por una aplicación que use una pila SCTP estándar, pero que a base de solicitar latidos en momentos concretos, imite el tráfico que generaría una pila SCTP-AP.

Actualmente no parece haber diferencia en cómo se encaminan por Internet los mensajes de control y los de datos en SCTP[1]. La diferencia de tamaño entre los típicos segmentos de datos y los latidos puede ser significativa, dependiendo del PMTU, pero eso no parece condicionar el encaminamiento. Si en el futuro se popularizaran discriminantes de este tipo, los latidos pequeños podrían volverse inapropiados para sondear el camino de los segmentos de datos.

[1] No podemos proporcionar referencias que confirmen esta afirmación.

Distribución temporal de las sondas

La planificación temporal del sondeo de la ruta debe ser escogida para detectar los fallos de ruta y sus recuperaciones de forma óptima. La estimación de la duración de los fallos de ruta es complicada y poco fiable en un protocolo de propósito general, por lo que hemos optado por una aproximación que minimice el retardo medio de detección, esto es, presuponer una distribución uniforme de la duración de los fallos de ruta y planificar las sondas periódicamente. El capítulo 5 está dedicado íntegramente a explicar y justificar esta decisión.

SCTP-AP podría utilizar otras planificaciones temporales de las sondas. Por ejemplo, si se sabe que los fallos de ruta van a estar motivados por ráfagas de pérdidas de duración prolongada en un enlace radio, se podrían utilizar planificaciones temporales de las sondas que aprovecharan las idiosincrasias de ese segmento de ruta.

Como el uso principal de SCTP-AP es el de un protocolo de propósito general, la planificación temporal de las sondas está optimizada para el caso más general posible, en el que no se puede presuponer que los fallos sigan patrones temporales predecibles. La adaptación de la planificación de las sondas a situaciones más concretas es un tema de investigación abierto que probablemente abordaremos en el futuro.

Valores por omisión de los nuevos parámetros del protocolo

Como hemos explicado en la sección anterior, SCTP-AP introduce dos nuevos parámetros a SCTP: `ActiveProbing.Burst` y `ActiveProbing.GiveUp`.

Hemos escogido los valores por omisión de estos parámetros de forma que el comportamiento de SCTP-AP sea lo más parecido posible al de SCTP en escenarios con RTT bajos:

- La duración del periodo de sondeo es la misma que lo que dura una detección de fallo de ruta en SCTP en el mejor de los casos, es decir `ActiveProbing.GiveUp` $= 63 * \mathrm{RTO}_{\min}$.
- El número de sondas a enviar también es el mismo que el número de ráfagas de transmisiones que realiza SCTP en su configuración estándar durante el PFD, es decir `ActiveProbing.Burst` $= 5$.

El motivo de esta decisión es minimizar la sorpresa de utilizar SCTP-AP en lugar de SCTP, en los escenarios más comunes en Internet.

Las aplicaciones deberían cambiar estos parámetros a su gusto si realmente quieren aprovecharse de las ventajas que ofrece SCTP-AP.

Recuperación de ruta y retransmisión rápida

En el contexto de SCTP-AP denominamos “recuperación de ruta” a la situación en que una pila SCTP-AP, habiendo entrado en el estado AP, recibe un asentimiento de su interlocutor.

En esta situación, se considera que la ruta se ha recuperado por completo y se restablece el normal envío de las transmisiones por la ruta "recuperada".

Cuando una ruta se encuentra en el estado AP durante más tiempo del definido por el parámetro `ActiveProbing.GiveUp`, y no se ha recibido asentimiento alguno por ella, decimos que estamos ante una retransmisión rápida: se desactiva la ruta y se restablecen las transmisiones y retransmisiones pendientes por una de las rutas alternativas.

Hemos utilizado el adjetivo 'rápida", para evidenciar que en este caso las retransmisiones se están enviando por la ruta alternativa *antes* de lo que el control de congestión aconsejaría para enviarlas por la ruta principal (ahora desactivada). Nótese que este mecanismo no tiene nada que ver con el popular mecanismo de "fast retransmit" de TCP y SCTP (§3.2 de [52]), a pesar de su similitud en el nombre de su traducción al castellano.

4.3 Simulaciones del funcionamiento básico

Hemos implementado SCTP-AP en el simulador OMNeT++ ([73]) versión 4.1. Nuestra implementación es una modificación de la pila SCTP que ya existía en el módulo INET Framework ([74]). Nuestra versión de INET Framework, que incluye lo necesario para utilizar SCTP-AP, está disponible en `https://github.com/alcortes-uc3m/inet`, rama SCTP-AP, etiqueta 1.0.

La figura 4.1 muestra la topología de la red utilizada en nuestras simulaciones. Tenemos dos dispositivos multiconectados: "origen" y "destino", que están conectados por dos rutas independientes y de idénticas características de transmisión. Cada una de estas rutas contiene dos routers.

El dispositivo origen va a enviar una cantidad fija de mensajes a destino usando una asociación SCTP-AP. Cuando el último mensaje sea recibido por destino, la asociación se cierra y la simulación termina. El mismo escenario lo

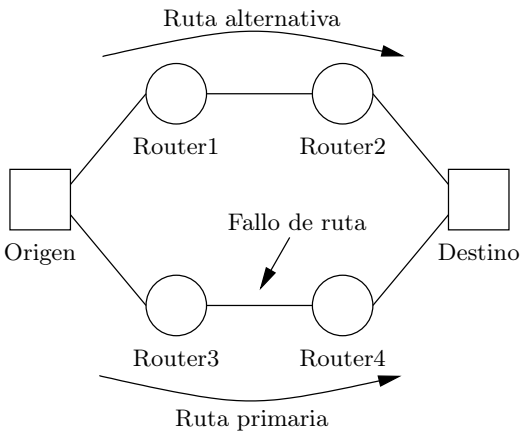

Figura 4.1: Topología de red de nuestras simulaciones.

repetimos con una pila SCTP estándar para poder comparar el comportamiento de ambos protocolos.

La aplicación ficticia que está enviando los datos tiene unos requerimientos tales que una interrupción de las comunicaciones de más de 10 segundos se considera un fallo de ruta. Hemos elegido este valor para mantener las gráficas de las simulaciones legibles.

La ruta primaria va a sufrir un fallo de ruta de diferentes duraciones, en el segmento de red entre `Router3` y `Router4`, lo que impedirá la transmisión de datagramas por ese segmento de red en ambos sentidos. Tenemos 4 configuraciones posibles:

- **Sin fallo de ruta**: En esta configuración, la ruta primaria no sufre fallo de ruta alguno y los datos se transmiten sin problemas. El propósito de esta configuración es mostrar que en ausencia de fallos de ruta, SCTP-AP y SCTP presentan el mismo comportamiento.

- **Fallo de ruta permanente**: La ruta primaria sufre un fallo en el segundo 3 de la transmisión y nunca más se recupera. En esta configuración las pilas SCTP y SCTP-AP tendrán que realizar un traspaso a la ruta alternativa, para poder terminar la transmisión de los datos con éxito.

- **Fallo de ruta de corta duración**: La ruta primaria sufre un fallo en el segundo 3, pero se recupera del mismo en el segundo 7, es decir, estamos ante un fallo de ruta de duración admisible para la aplicación (inferior a 10 segundos).

  La recuperación de la ruta tiene lugar antes de que tanto SCTP como SCTP-AP hayan decidido que se encuentran ante un fallo de ruta permanente, por lo que no se realiza un traspaso y la transmisión de los datos continúa por la ruta primaria una vez ésta se recupera del fallo.

- **Fallo de ruta de larga duración**: La ruta primaria sufre un fallo en el segundo 3 y se recupera del mismo en el segundo 18,5, es decir, estamos ante un fallo de ruta de una duración superior a lo que la aplicación considera aceptable (superior a 10 segundos).

  La duración de este fallo de ruta es demasiado corta como para que SCTP la considere un fallo de ruta permanente y la transmisión se completa por la ruta primaria una vez ésta se recupera, sin traspaso alguno. A nivel de aplicación, se apreciará un deterioro de la calidad de la comunicación evidente.

  La pila SCTP-AP está configurada para detectar fallos de ruta de 10 segundos de duración, por lo que se considerará este fallo de ruta como permanente y se realizará un traspaso para continuar enviando los datos por la ruta alternativa. A nivel de aplicación la interrupción de las comunicaciones se mantiene dentro de los límites aceptables.

Detalles de la simulación

Los enlaces que conectan los dispositivos a los routers representan la red de acceso, con un ancho de banda de 10 Mbps y un retardo de propagación de 25 ns.

Los enlaces entre los routers representan una red de comunicación pública, con un ancho de banda de 100 kbps y un retardo de propagación de 1 ms.

Todos los enlaces tienen una probabilidad de pérdidas de 0 %. Hemos simulado los enlaces usando modelos PPP. Todos los routers tienen colas FIFO con capacidad para 25 paquetes (poco más de 1 segundo de cola[2]).

El escenario que estamos simulando es la transferencia de un fichero de 107.200 octetos entre el origen y el destino: El dispositivo origen envía 200 mensajes al destino usando SCTP-AP o SCTP. Los mensajes de aplicación tienen una longitud de 536 octetos, que es un número suficientemente bajo como para asegurarnos que cualquier dispositivo receptor podrá manejar un segmento SCTP con los 536 octetos mas algún eventual trozo de control ([75]).

La ruta primaria para ambos dispositivos es aquella que pasa por `Router3` y `Router4`. Esta ruta sufrirá un fallo de ruta de diferentes duraciones en nuestras simulaciones, lo que interrumpirá la transmisión de datos y mensajes de control en ambos sentidos.

Simulamos los fallos de ruta estableciendo una probabilidad de pérdida de paquetes de 100 % durante la duración de los mismos en el enlace entre el `Router3` y `Router4`.

Las pilas SCTP/SCTP-AP de ambos dispositivos están configuradas con los valores estándar. En las simulaciones donde SCTP-AP está habilitado, `ActiveProbing.GiveUp` es 10 segundos y `ActiveProbing.Burst` es 10, es decir, una sonda activa por segundo. Hemos decidido usar estos valores de configuración de SCTP-AP porque facilitan la interpretación de las gráficas y los resultados de nuestras simulaciones, no son representativos de los valores óptimos que debería usar una aplicación concreta.

En estas simulaciones hemos planificado las sondas activas sin aplicarles la variación aleatoria que incluye la ecuación (4.1), es decir, el tiempo entre cada sonda activa, en estas simulaciones, pasa a estar determinado por la ecuación:

$$T_{HB} = \frac{\texttt{ActiveProbing.GiveUp}}{\texttt{ActiveProbing.Burst}} \tag{4.2}$$

La razón de eliminar la variación temporal aleatoria de las sondas en estas simulaciones es únicamente para simplificar la interpretación de las gráficas que presentamos en las secciones sucesivas. En §4.5.4 introducimos las razones para estas variaciones y en §5.8 presentamos un estudio detallado de los efectos que tienen sobre el protocolo.

[2]Hemos escogido estos valores para que nuestras modificaciones en el mecanismo de detección de fallo de ruta resulten sencillas de explicar e ilustrar en gráficas.

Acerca del parámetro PMR

La versión de la pila SCTP de INET Framework que se usó en las simulaciones tenía un bug de implementación, por el cual el mecanismo de detección de fallo de ruta decidía cambiar de ruta en la retransmisión PMR − 1 en lugar de en la retransmisión PMR.

Aunque hemos resuelto este bug posteriormente, las simulaciones que presentamos en este documento se han realizado con la versión antigua de la pila, ya que el parche que enviamos para solucionar el problema tardó unas semanas en aplicarse a la rama en producción del software.

Esto significa que el simulador va a detectar el fallo de ruta a la cuarta retransmisión en vez de a la quinta, pero esto no representa un problema, ni afecta a nuestras conclusiones. Antes al contrario, las gráficas son más claras de esta forma ya que el traspaso se realiza antes y el eje de tiempos es más corto.

Incluso si se hubiese aplicado el parche a tiempo, hubiésemos utilizado un valor de PMR de 4 retransmisiones en vez del valor estándar de 5 para acortar el eje de abscisas de las gráficas.

Para reproducir fielmente nuestras simulaciones en versiones modernas de la pila SCTP del simulador, sería necesario utilizar un valor de PMR de 4 retransmisiones en vez de 5.

Ajuste artificial de la tasa de envío

En nuestros primeras simulaciones la aplicación entregaba todos los mensajes a la pila SCTP/SCTP-AP al comienzo de la simulación, de forma que los 200 mensajes a enviar estaban disponibles para ser transmitidos desde el establecimiento de la asociación. El control de congestión de SCTP ajustaba la tasa de envío y después de unos segundos se alcanzaba una tasa de envío razonablemente estable y ajustada a las características de la ruta.

Sin embargo, la opinión generalizada de los revisores del WoWMoM 2011[3] era que este ajuste de la tasa de envío por parte del control de congestión, con sus retransmisiones, añadía un ruido innecesario a nuestra simulación y distraía la atención de los efectos y consecuencias del mecanismo de detección de fallo de ruta, que es lo queremos mostrar.

Haciéndonos eco de estas reclamaciones, hemos ajustado nuestra simulación de forma que la tasa de envío de datos sea, desde el principio, la tasa adecuada para las características del canal. De esta forma no es necesario la intervención del mecanismo de control de congestión y las gráficas quedan más claras.

Para ello establecimos un tamaño de la cola de envío en la pila SCTP/SCTP-AP del dispositivo origen tal que la tasa de envío fuese la adecuada para la ruta[4]. Este ajuste nos parece adecuado, ya que diferentes sistemas operativos utilizan diferentes tamaños de colas de envío para las pilas SCTP por lo que no hay un consenso generalizado para este parámetro y el valor que nosotros hemos utilizado tiene la propiedad de clarificar los resultados de nuestras simulaciones.

[3] IEEE International Symposium on a World of Wireless Mobile and Multimedia Networks, congreso al que enviamos un artículo sobre este tema, que finalmente fue rechazado.

[4] El efecto de este ajuste es similar a reducir la ventana de recepción en el destino, pero actuamos exclusivamente sobre el origen.

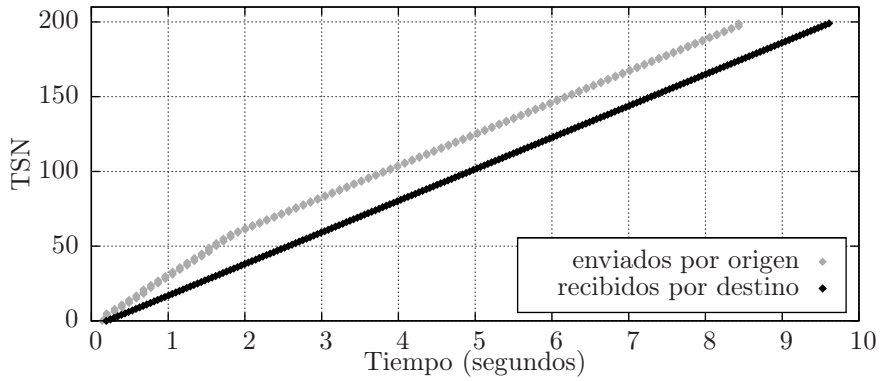

Figura 4.2: Sin fallos de ruta: TSNs enviados por el origen y recibidos por el destino por la ruta primaria.

Hemos realizado este ajuste de forma que el impacto en la simulación es mínimo, nuestros resultados en cuanto a tiempo de traspaso y otras medidas sobre el mecanismo de detección de fallo de ruta son los mismos antes y después del ajuste, pero las gráficas y explicaciones se simplifican notablemente después del mismo.

El valor concreto de cola de envío de la pila SCTP/SCTP-AP que utilizamos fue 13.401 octetos[5].

4.3.1 Resultados de las simulaciones

Simulación sin fallo de ruta

En la simulación sin fallo de ruta, todos los segmentos de datos son enviados por la ruta primaria y no se producen pérdidas de paquetes ni otros inconvenientes de transmisión. Tanto la pila SCTP como la pila SCTP-AP deberían comportarse de forma idéntica ya que el mecanismo de detección de fallo de ruta no entra en juego en toda la simulación.

La figura 4.2 muestra los números de secuencia de transmisión (TSNs) de los mensajes enviados por el origen y recibidos por el destino cuando la ruta primaria no sufre fallo de ruta alguno.

En ausencia de fallo de ruta, todos los mensajes de datos se envían sin problemas por la ruta principal (por eso en la figura no aparecen mensajes enviados a la ruta alternativa).

No hay diferencia alguna en el comportamiento de SCTP y de SCTP-AP en este caso. Ni el mecanismo de PFD de SCTP ni el de SCTP-AP se activan en este caso. Las gráficas para ambos protocolos son idénticas y por eso la figura sólo muestra el comportamiento de uno de ellos.

[5]Que es el tamaño de la cola de los routers + 1 octeto necesario por cuestiones de implementación de la pila SCTP en el simulador.

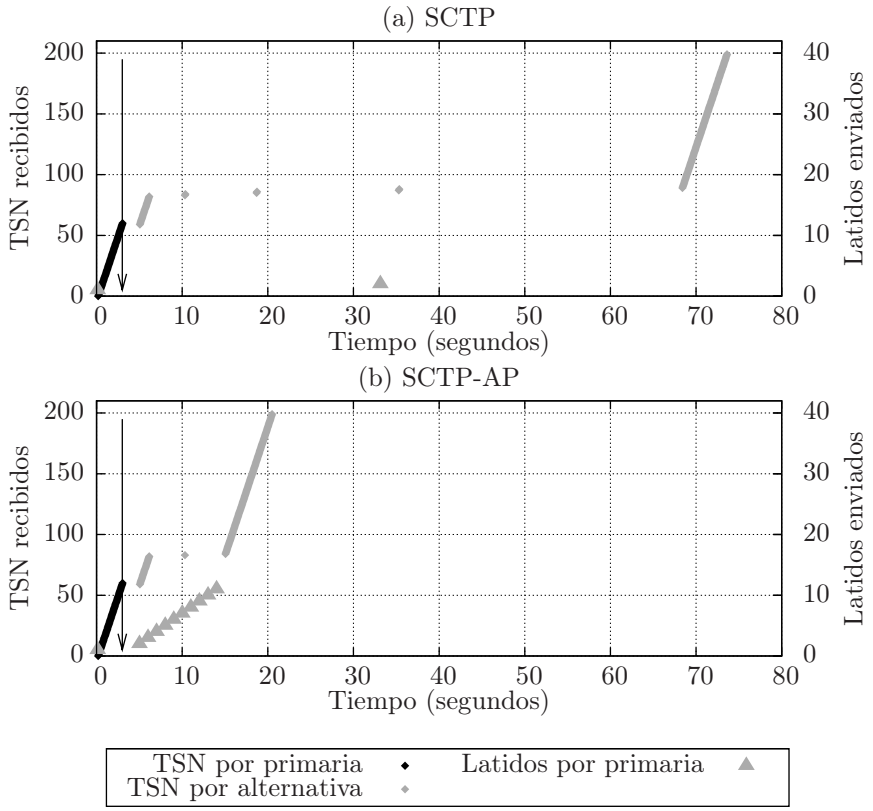

Figura 4.3: Fallo de ruta permanente: TSNs recibidos por el destino a través de la ruta primaria y la alternativa. El comportamiento de SCTP se muestra en la subfigura (a) y el de SCTP-AP en la (b). La flecha hacia abajo en el segundo 3 indica el momento del fallo de ruta. Los latidos enviados por el origen utilizan el eje de ordenadas mostrado a la derecha.

El tiempo necesario para entregar todos los segmentos de datos al destino es de aproximadamente 10 segundos, por lo que el `goodput` es de aproximadamente 85,7 Kbps.

Fallo de ruta permanente

En la simulación de un fallo de ruta permanente la ruta primaria sufre un fallo de ruta en el segundo 3. A partir de este momento ningún paquete se transmitirá con éxito en ninguno de los dos sentidos (ni de `Router3` a `Router4`, ni viceversa) y la transmisión de los segmentos de datos tendrá que completarse por la ruta alternativa, una vez las pilas SCTP y SCTP-AP se percaten de la caída del enlace y decidan utilizar la ruta alternativa. La figura 4.3 muestra los TSNs recibidos por el destino mediante SCTP y SCTP-AP.

La figura también muestra los latidos enviados por el origen a través de la ruta primaria. En la simulación SCTP-AP, los últimos 10 latidos son las sondas activas utilizadas para muestrear la ruta. Dado que la ruta primaria está caída, ninguno de estos latidos alcanzará al destino.

El valor de RTO_{fail} en el momento del fallo de ruta es 2,4 segundos. La duración de la detección del fallo de ruta en la simulación SCTP es de aproximadamente 66 segundos. La duración de la detección del fallo de ruta en la simulación SCTP-AP es de aproximadamente 12,4 segundos: 2,4 segundos hasta la primera retransmisión y otros 10 segundos correspondientes al valor de `ActiveProbing.GiveUp`.

Fallo de ruta corto

Con esta simulación pretendemos observar el comportamiento de SCTP y SCTP-AP ante un fallo de ruta temporal de corta duración. Con corta duración nos referimos a que la ruta se recupera del fallo antes de que tanto SCTP como SCTP-AP se den cuenta del fallo.

Hemos escogido un fallo de ruta de duración 5,5 segundos, que es inferior a los 12,4 segundos y a los 66 segundos que tardaban las pilas SCTP-AP y SCTP, respectivamente, en detectar el fallo de ruta en la simulación del fallo de ruta permanente.

En este caso, el resultado esperado es que la transmisión de los segmentos de datos se retome por la ruta primaria al poco tiempo de que esta se recupere, y por tanto la transmisión de los datos se complete por la ruta primaria sin mayor problema que un retraso derivado del tiempo en que la ruta ha estado caída y lo que ha tardado cada pila en detectar su recuperación.

La figura 4.4, en la página siguiente, muestra los TSNs recibidos por el destino. La pila SCTP se da cuenta de la recuperación de la ruta con la segunda ráfaga de retransmisiones, aproximadamente en el segundo 12. La pila SCTP-AP detecta la recuperación de la ruta antes, en el segundo 9, cuando el quinto latido es asentido por el destino.

En ambos casos, la recuperación de la ruta se ha detectado y se ha terminado la transmisión de los datos por la ruta primara, ya recuperada del fallo.

Fallo de ruta largo

Con esta simulación pretendemos observar el comportamiento de las pilas SCTP y SCTP-AP ante un fallo de ruta temporal de larga duración. Con larga duración nos referimos a un fallo de ruta más largo de lo que tarda SCTP-AP en detectarlo, pero más corto de lo que tarda SCTP.

El comportamiento esperado es que la pila SCTP-AP detectará el fallo de ruta en el mismo tiempo que en el caso de la simulación del fallo de ruta permanente y terminará por transmitir los datos por la ruta alternativa.

La pila SCTP, mas lenta en detectar el fallo de ruta, verá como la ruta se recupera antes de decidir que se encuentra ante un fallo de ruta y terminará por transmitir los datos por la ruta primaria una vez esta se reponga del fallo.

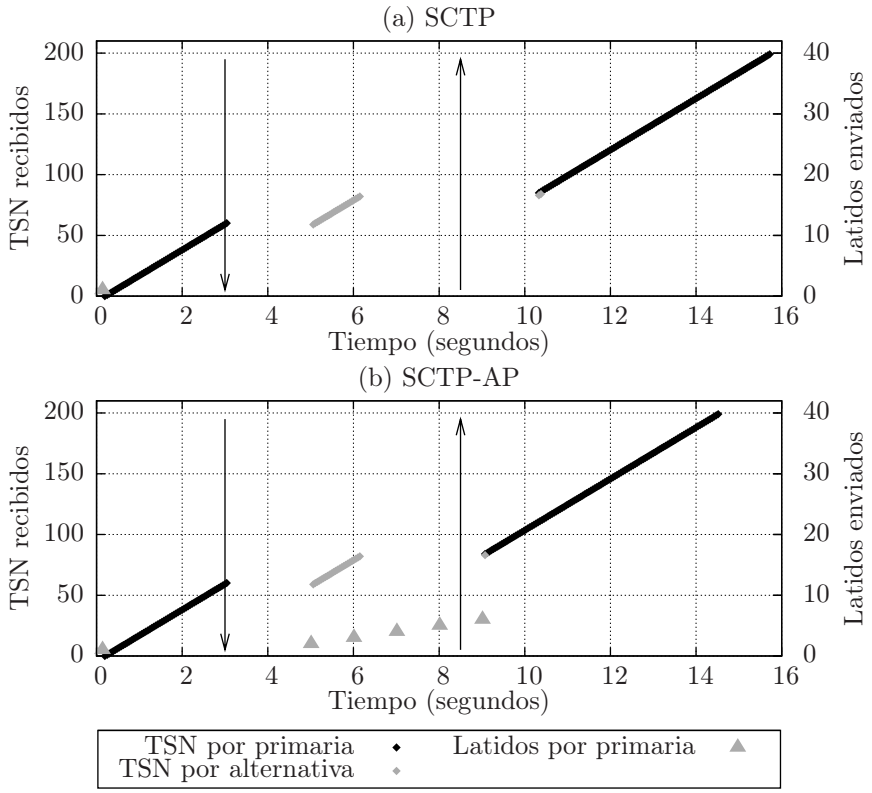

Figura 4.4: Fallo de ruta corto: TSNs recibidos por el destino por las rutas primaria y alternativa. Las flechas en el segundo 3 y en el segundo 8,5 indican la caída y recuperación de la ruta respectivamente. Los latidos enviados por el origen utilizan el eje de ordenadas mostrado a la derecha.

Hemos escogido un fallo de ruta de duración 15,5 segundos, que es más de los 12,4 segundos que tarda SCTP-AP en detectar el fallo pero menos de los 66 que tarda SCTP.

La figura 4.5, en la página siguiente, muestra el comportamiento de ambas pilas y se observa como cada una de ellas acaba transmitiendo los datos por rutas diferentes. La pila SCTP detecta la recuperación de la ruta con la transmisión correspondiente a la tercera retransmisión, en el segundo 19 y continúa la transmisión de los datos por la ruta recién recuperada. La pila SCTP-AP se da por vencida con el décimo sondeo, en el segundo 15, marca la ruta primaria como inactiva y continua la transmisión de los datos por la ruta alternativa.

Aunque la pila SCTP-AP proporciona un `goodput` superior a la pila SCTP, la recuperación de la ruta pasa inadvertida para ella.

Este comportamiento es el esperado, ya que si la aplicación ha establecido un tiempo de `ActiveProbing.GiveUp` de 10 segundos, es porque fallos de ruta

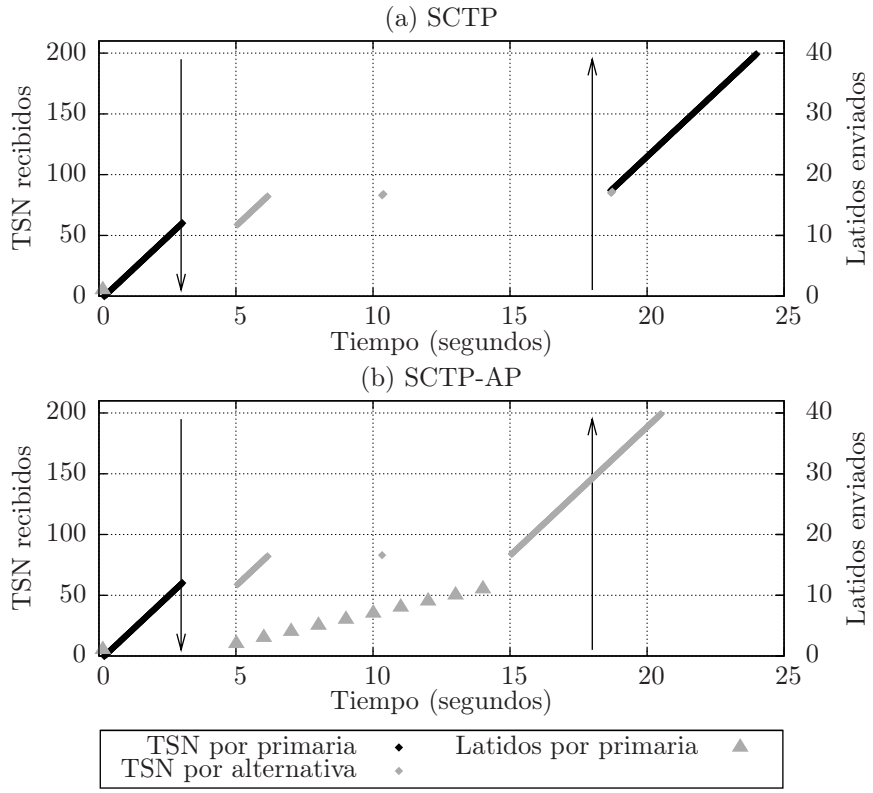

Figura 4.5: Fallo de ruta largo: TSNs recibidos por el destino por las rutas primaria y alternativa. El comportamiento de SCTP se muestra en la subfigura (a) y el de SCTP-AP en la (b). Las flechas en 3 y 18,5 segundos muestran los momentos en que sucede el fallo de ruta en la primaria y en el que se recupera, respectivamente. Los latidos usan el eje de ordenadas de la derecha.

temporales de duración superior a 10 segundos han de considerarse como fallos de ruta permanentes.

La recuperación de las rutas se consigue en SCTP (y en SCTP-AP) mediante la monitorización activa de las rutas que no están siendo utilizadas. Esta monitorización consiste en enviar un latido cada 30 segundos para ver si se obtiene respuesta del otro extremo. En caso de que se reciba respuesta a uno de estos latidos por la ruta primaria, se considera ésta como recuperada y se vuelve a utilizar para enviar los datos.

Si esta simulación fuese más larga, de forma que diese tiempo a la pila SCTP-AP a enviar un latido de monitorización por la ruta primaria, se restablecería la transmisión de los datos por dicha ruta. Este primer latido se enviaría 30 segundos después de marcar la ruta como inactiva (en el segundo 15), es decir en el segundo 45.

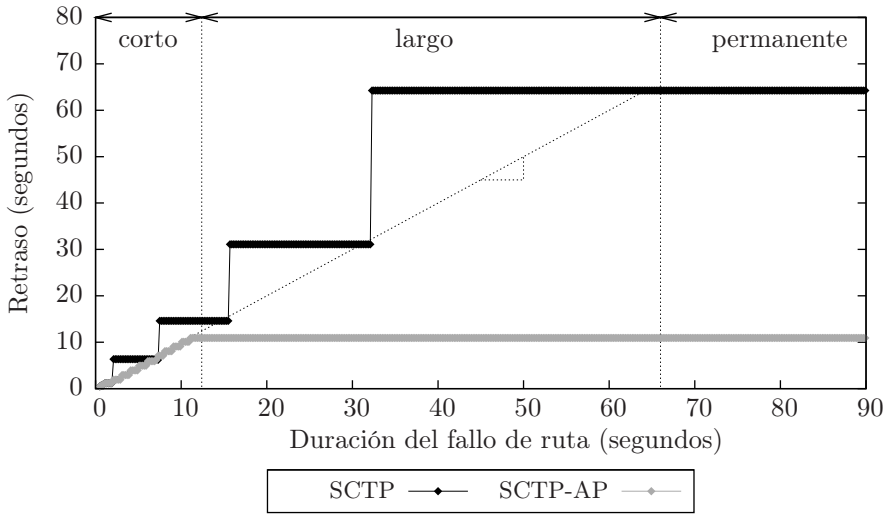

Figura 4.6: Retraso provocado por un fallo de ruta según su duración.

En caso de que quisiéramos acelerar la detección de la recuperación de la ruta después de un fallo de ruta temporal, habría que modificar el parámetro de sondeo de rutas inactivas a un valor menor de 30 segundos.

Reducir este valor aumenta el tráfico de control enviado a una ruta que podría estar sufriendo una sobrecarga importante. En [76] se propone una manera de estudiar el efecto general de modificar este valor a partir de la cantidad de tráfico adicional generado. En [77] se propone ajustar la frecuencia de sondeo de las rutas inactivas dependiendo de la frecuencia con la que se producen los fallos de ruta, de forma que el sondeo se realice más a menudo en las rutas que más fallos de ruta han sufrido.

4.4 Comparativa para diferentes duraciones del fallo de ruta

La figura 4.6 muestra el retraso en que se incurre en la transferencia de un fichero en el mismo escenario de las simulaciones anteriores para diferentes duraciones del fallo de ruta.

El retraso que se sufre en SCTP-AP es siempre menor o igual al de SCTP.

En el caso de un fallo corto, es decir, un fallo de duración inferior a aproximadamente `ActiveProbing.GiveUp`, el retraso por SCTP-AP presenta un comportamiento con escalones uniformes, tantos como sondas se envíen (esto es, según el valor de `ActiveProbing.Burst`).

Ante fallos más largos, SCTP-AP ofrece un retardo constante, con un valor aproximado de `ActiveProbing.GiveUp`. En estos casos, la transmisión se com-

pletará siempre en SCTP-AP por la interfaz alternativa, mientras que SCTP completará la transmisión por la primaria si el fallo de ruta no es permanente.

4.5 Consideraciones de estabilidad del protocolo

La crítica más habitual que recibe SCTP-AP es que presenta un comportamiento poco estable durante la detección del fallo de ruta: ante la sospecha de un fallo de ruta, que bien podría estar causado por un problema de congestión, SCTP-AP decide mandar más tráfico, en forma de sondas periódicas.

Es un comentario razonable, pero infundado. De hecho, SCTP-AP es más estable que SCTP, en especial en la fase de detección de fallo de ruta. En las siguientes secciones analizamos la estabilidad de SCTP-AP y la comparamos con la de SCTP.

Breve introducción a la estabilidad de protocolos

Desde un punto de vista algo inocente, la estabilidad de un protocolo podría resumirse de la siguiente forma: "el emisor no debe introducir un paquete en la red hasta estar seguro de que el receptor ya ha extraído el paquete anterior".

De esta manera se asegura que el tráfico ofrecido sea menor a la capacidad de la red y se elimina el riesgo de un colapso por congestión ([78, 79]).

Una interpretación literal de este principio nos lleva a algoritmos de transmisión tipo ping-pong con una ventana de congestión de tamaño 1, que aunque tienen una excelente capacidad para auto-adaptar la tasa de envío a las características de transmisión de la red, resultan extremadamente ineficientes en el aprovechamiento de la capacidad del canal.

En la práctica, sabemos que podemos hacer una transmisión más eficiente, sin arriesgarnos a un colapso por congestión, mediante la utilización de ventanas de congestión de tamaño variable, cuyo tamaño responda a la congestión *percibida* por los extremos y al uso de mecanismos de transmisión rápida como slow-start y fast-retransmit, si los utilizamos en los momentos apropiados ([52]).

La idea subyacente a estos mecanismos de transmisión es la siguiente: la capacidad del canal es desconocida por los extremos de una transmisión, pero son ellos los que deben asegurarse de no sobrepasarla. A estos efectos, los extremos deben inferir la capacidad disponible en cada momento, si se cree que estamos infrautilizando el canal, enviemos más rápidamente, si por contra tenemos indicios de que el canal está siendo saturado, enviemos más lentamente.

Además, estos mecanismos sitúan a los flujos que comparten una red en una situación de competencia unos con otros, que puede dar lugar a un reparto no equitativo de la capacidad total entre los diferentes flujos y por eso, dentro del análisis de estabilidad se suelen incluir también las características de fairness, aggressiveness y responsiveness [79, 80].

Estabilidad de protocolos multiruta

La estabilidad de red en escenarios multiconectados merece una especial atención:

- **Ecuanimidad del reparto de recursos**: Es un hecho bien conocido que los protocolos multiruta que realizan transferencia simultánea no son ecuánimes en el reparto de recursos a no ser que se realice un control de congestión de forma global a todas las rutas ([81]).

  SCTP-AP no es un protocolo que realice transferencia simultánea, y al compartir el mecanismo estándar de control de congestión de TCP, se puede concluir que la ecuanimidad en el reparto está asegurada, por lo menos en los mismos términos TCP-friendliness que SCTP ([82, 72]).

- **Colapso por congestión**: Una cuenta global de los paquetes introducidos en la red debe tener en consideración *la suma total de paquetes introducidos por todas las interfaces del dispositivo*.

  Sin embargo, si se consideran cada una de las interfaces por separado, puede ser que el protocolo sea *localmente inestable por una de las interfaces* aunque sea globalmente estable por la suma de todas ellas.

  Por ejemplo, pensemos en un protocolo para dispositivos multiconectados que no utilizara ningún tipo de control de congestión por una de las rutas, y que por las otras utilizará un estricto control de congestión que tuviese en cuenta los datos enviados por la primera. Globalmente el tráfico ofrecido a la red sería estable, pero localmente, por la primera ruta, sería inestable.

  El impacto de este hipotético protocolo en la congestión de la red sería bien diferente si el cuello de botella a congestionar es compartido por ambas rutas o si existieran cuellos de botella para cada una de ellas, probablemente con restricciones de capacidad bien diferentes.

  Así pues podemos hablar de dos tipos de estabilidad en escenarios multiconectados: estabilidad global y estabilidad local. Evidentemente, lo ideal es diseñar protocolos estables según ambas acepciones.

  No hemos encontrado estudios previos sobre este particular y nuestro trabajo tampoco está dirigido a profundizar en estos detalles. Aún así el análisis de estabilidad que aportamos en las siguientes secciones contempla un análisis de estabilidad local y otro global, ya que pensamos que la diferencia es importante.

4.5.1 Estabilidad local de SCTP-AP

Localmente, SCTP-AP introduce tráfico a menor velocidad que SCTP por la ruta en que se está realizando una detección de fallo de ruta, por lo que a pesar del sondeo activo, SCTP-AP resulta más estable localmente que SCTP.

Esto es debido a que los octetos extra introducidos por el sondeo activo se reservan previamente en la ventana de pérdidas de la ruta que está siendo sondeada, dando lugar a un menor envío de datos por esa interfaz, para dar cabida al tráfico de control adicional introducido por las sondas activas.

La figura 4.7 muestra el tráfico introducido por SCTP y SCTP-AP en la ruta sobre la que se está realizando la detección de fallo de ruta. Los datos están sacados de la simulación de fallo de ruta permanente en §4.3.1.

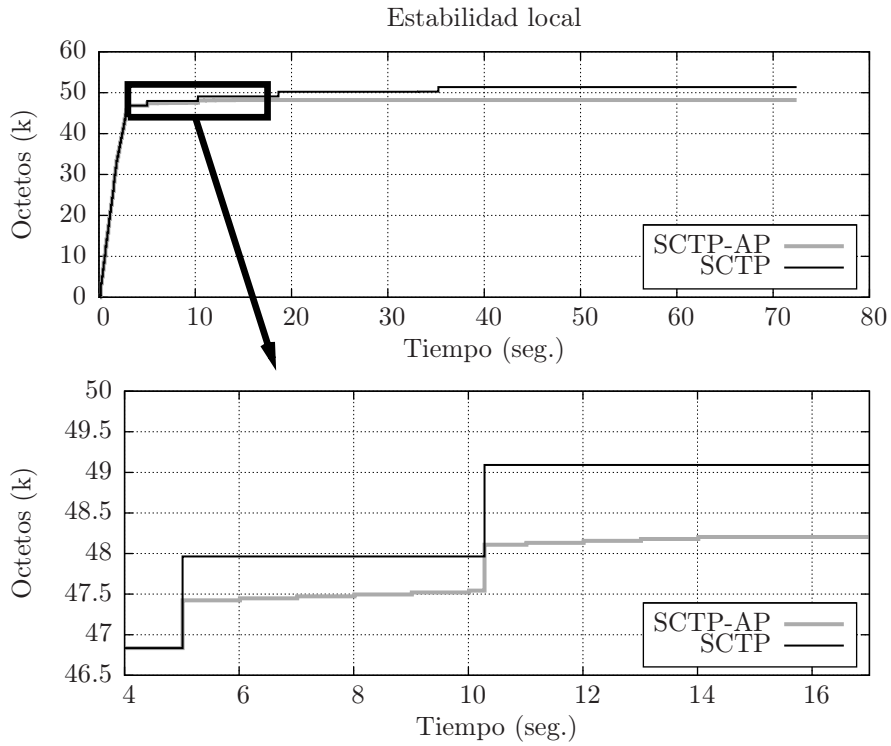

Figura 4.7: Tráfico introducido en la red, en octetos de carga útil IP, por la ruta sobre la que se está realizando una detección de fallo de ruta en SCTP y en SCTP-AP.

En esta simulación, la detección de fallo de ruta se realiza en SCTP-AP durante el periodo 5-15 segundos. Durante ese tiempo en SCTP se realiza la transmisión de 2 ráfagas de datos nuevos a la ruta, mientras que en SCTP-AP se realiza la transmisión de 2 ráfagas con *menos* datos nuevos y un sondeo activo de 10 latidos.

El tráfico extra introducido por el sondeo activo es de aproximadamente 240 octetos, correspondiente a la carga útil IP de 10 latidos en nuestros simulador.

La cantidad de octetos introducida en la red entre los segundos 5 y el 15 es inferior en SCTP-AP que en SCTP. En el segundo 5, en SCTP-AP se envían menos cantidad de datos, ya que la ventana de pérdidas es menor que en SCTP. A partir de ahí se aprecian el incremento de tráfico producido por las 10 sondas activas, formando 10 escalones de 24 octetos cada uno.

En el segundo 14, la cantidad de tráfico introducida por ambas implementaciones debería ser similar, siendo la única diferencia entre una y otra que en SCTP-AP el tráfico de datos total introducido es menor, debido tanto a la menor ventana de pérdidas en SCTP como al crecimiento más lento de su ventana de congestión, al partir ésta de un valor inicial inferior.

Detalle de implementación del simulador

La diferencia entre la cantidad de datos enviada por las pilas SCTP y SCTP-AP en nuestra simulación es aún mayor de lo que debería.

El simulador que estamos utilizado no implementa la fragmentación de mensajes de usuario en situaciones donde se pueden agregar varios de ellos en un solo segmento. Por ejemplo, si se pueden enviar dos mensajes de usuario y medio, en un solo segmento SCTP, el simulador solo enviará los dos primeros.

En nuestras simulaciones hemos reducido la ventana de pérdidas (de 1220 octetos) en la cantidad de octetos que consumen las sondas activas (240 octetos). El resultado es una ventana de pérdidas de 980 octetos, suficientes para enviar un segmento SCTP de datos con 1 mensaje completo de datos del usuario (536 octetos) más una porción del siguiente.

Como se aprecia en la figura, la cantidad de datos enviados con una ventana de pérdidas de 980 octetos es solamente de 536 octetos, ya que el simulador tan solo ha enviado un mensaje de usuario, en vez de enviar el primer mensaje y parte del siguiente.

Otras implementaciones de pilas SCTP pueden tener las mismas limitaciones, pero aún así esto no es un problema para el estudio de la estabilidad, ya que simplemente da lugar a un envío todavía mas conservador del que debería y por tanto más estable.

4.5.2 Estabilidad global de SCTP-AP

Globalmente SCTP-AP introduce también tráfico a un ritmo más lento que SCTP durante la detección del fallo de ruta, por lo que SCTP-AP es más estable globalmente que SCTP.

La figura 4.8, en la página siguiente, muestra el tráfico total introducido por SCTP y SCTP-AP por todas las rutas. Los datos están sacados también de la simulación de fallo de ruta permanente en §4.3.1.

4.5.3 Consecuencias y limitaciones de la estabilidad

Utilizar una ventana de pérdidas menor de 1 MSS está expresamente permitido tanto por la especificación de SCTP (§7 de [29]) como por la especificación de los mecanismos de TCP de la que deriva (§3 de [52]).

La consecuencia fundamental de conseguir la estabilidad del protocolo mediante a disminución de la ventana de pérdidas es la disminución de la agresividad por la ruta. Esto deriva en una menor tasa de transmisión en los primeros instantes después de una retransmisión (como se deduce de [83, 84]).

Pensamos que dado que el sistema de detección de fallo de ruta de SCTP-AP detectará el fallo de ruta o su recuperación antes que SCTP, la menor agresividad en el envío de los datos será compensado con creces por un goodput superior evaluado en la totalidad de la comunicación.

Una de las limitaciones fundamentales de este mecanismo para estabilizar SCTP-AP es que la reserva de la ventana de pérdidas ha de ser menor que el

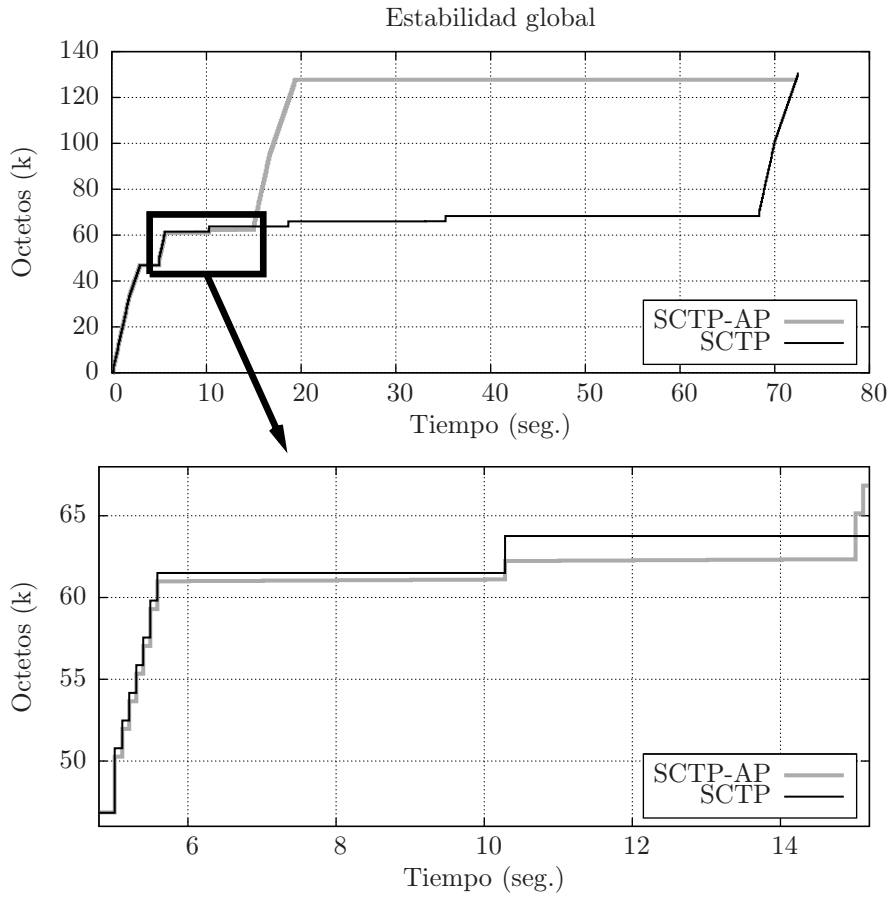

Figura 4.8: Tráfico total introducido en la red, como carga útil IP, por SCTP y SCTP-AP.

MSS. Por ello, la cantidad de los latidos a enviar y su tamaño han de limitarse a este valor:

$$\texttt{ActiveProbing.Burst} * \text{tamaño de los latidos} < \texttt{MSS} \tag{4.3}$$

Teniendo en cuenta que la cantidad de latidos a enviar raramente sobrepasará el valor de 5 (§5.9) y que el tamaño de los latidos en las diferentes implementaciones de SCTP no supera los 60 octetos, esto no debería ser un problema: la reserva en la ventana de congestión de pérdidas será generalmente menor que 300 octetos.

En caso de que la reserva en la ventana de congestión de pérdidas resulte demasiado grande o que se quiera aumentar la agresividad del protocolo sin llegar a hacerlo inestable, se puede recurrir a enviar latidos más pequeños. Como

el tamaño mínimo de los latidos SCTP es de 20 octetos de payload IP, la reserva será como mínimo de 100 octetos[6].

4.5.4 Sincronismo de los latidos

Otro aspecto importante de la estabilidad del protocolo es evitar el sincronismo entre asociaciones que comparten segmentos de una misma ruta.

Por ejemplo, si tenemos varias asociaciones SCTP-AP compartiendo un mismo segmento de red, que falla por algún motivo, cada una de ellas enviará sus propias sondas activas. Aunque estas asociaciones no pertenezcan al mismo dispositivo, pueden tener los mismos parámetros de configuración, por ejemplo, por que estén siendo utilizadas por la misma aplicación o por que utilicen los parámetros de configuración por omisión del protocolo.

Si esto es así, un fallo de ruta desencadenará la transmisión de las sondas activas de todas las asociaciones en los mismos momentos de tiempo, lo que puede ser perjudicial para la red.

Para ello, SCTP evita utilizar planificaciones temporales fijas, introduciendo una variación aleatoria del ±50 % del RTO en el envío de sondas para monitorizar rutas inactivas.

Una estrategia similar debe aplicarse a los latidos de sondeo de la ruta primaria de SCTP-AP para evitar el sincronismo de las sondas activas entre aplicaciones que usen los mismos parámetros de configuración. La ecuación (4.1) incluye un término aleatorio que implementa esta variación aleatoria en la planificación de los latidos, de una magnitud equivalente al ±50 % del RTO característico de una implementación SCTP estándar.

En §5.8 presentamos un estudio del efecto que tiene introducir variaciones aleatorias en la planificación de las sondas. En §5.9 resumimos los resultados del estudio: podemos adelantar que aplicar una variación a la temporización de las sondas *empeora las prestaciones en cuanto a detección de la recuperación de la ruta*. En lugar de aplicar una variación equivalente al ±50 % del RTO, se podría realizar un estudio más en profundidad de este compromiso. Este estudio es parte de nuestro trabajo futuro.

4.6 Simulaciones ante flujos competidores

En §4.3 hemos presentado los resultados de algunas simulaciones que ilustran el funcionamiento básico de SCTP-AP ante fallos de ruta de diferentes duraciones. En §4.5 mostramos simulaciones de la cantidad de datos enviados por SCTP-AP durante el traspaso, donde se aprecia la estabilidad de nuestra propuesta.

Todas esas simulaciones se han realizado en entornos especialmente diseñados para ilustrar los detalles del protocolo de la forma más sencilla posible.

En esta sección presentamos los resultados de simulaciones en entornos más realistas, donde el dispositivo móvil está conectado a Internet mediante una

[6] El MSS mínimo garantizado para TCP en IPv4 es de 536 octetos ([75]).

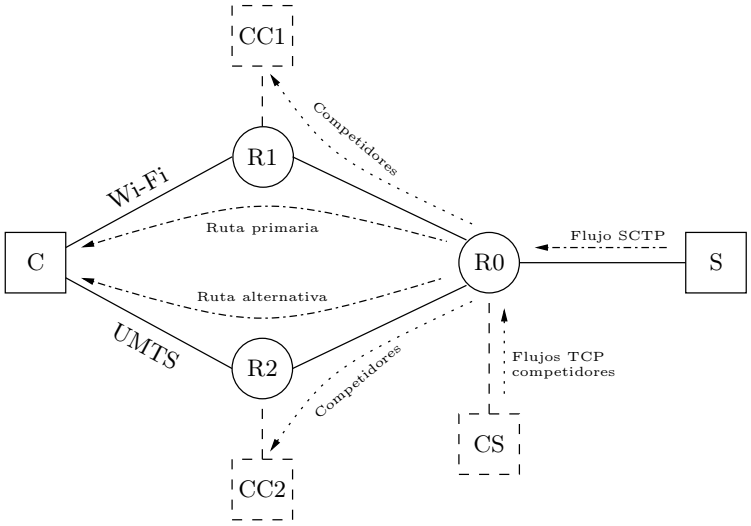

Figura 4.9: Topología de red de las simulaciones de competencia con otros flujos.

interfaz Wi-Fi y otra UMTS, con pérdidas, y donde nuestros flujos compiten por el ancho de banda con otros flujos TCP.

El propósito de estas simulaciones es observar el comportamiento de nuestra propuesta en un escenario más general y verificar que nuestras modificaciones no producen comportamientos inesperados cuando se combinan con el resto de mecanismos de SCTP que generalmente están involucrados en una transmisión real.

4.6.1 Topología

La figura 4.9 muestra la topología de red que hemos utilizado en las simulaciones:

- Un **cliente C**, que está descargando datos de un **servidor S**, mediante una asociación SCTP que utilizan las interfaces Wi-Fi y UMTS de C.
- Dos **clientes competidores CC1 y CC2**, que se están descargando datos de un **servidor competidor CS**, mediante varias conexiones TCP cada uno.

La red que conecta todos estos dispositivos está formada por los **routers R0, R1 y R2**:

- El enlace entre el cliente C y R1 simula una red Wi-Fi, a 54 Mbps con un RTT de 10 ms.
- El enlace entre el cliente C y R2 simula una red UMTS a 14,4 Mbps y un RTT de 35 ms.

- Los enlaces entre los routers simulan un backbone de Internet que ofrece a cada flujo TCP competidor un ancho de banda de 8 Mbps y un RTT de 60 ms por la ruta entre R0 y R1 y un ancho de banda de 7 Mbps y 30 ms de RTT por la ruta entre R0 y R2.
- El resto de enlaces simulan conexiones Ethernet a 100 Mbps con un retardo despreciable.
- El PMTU es 1492 octetos.
- El tamaño de las ventanas de recepción anunciadas por las pilas TCP y SCTP y los buffers de las tarjetas de red es suficiente para aprovechar el producto $BW * RTT$ por todos los enlaces.

Hemos elegido estos valores basándonos en medidas realizadas en nuestras propias casas. Como en las secciones anteriores, hemos utilizado el simulador OMNeT++. Los valores utilizados para la configuración de las pilas TCP y SCTP son los valores estándar del simulador y los protocolos.

En la pila SCTP-AP hemos utilizado un valor para `ActiveProbing.Burst` de 5 y un valor para `ActiveProbing.GiveUp` de 5 segundos.

Hemos elegido el valor de `ActiveProbing.Burst` basándonos en la figura 5.13; un valor de 5 es suficiente para que la mejora frente a la estrategia geométrica sea considerable y es coherente con el valor por omisión del PMR de SCTP.

Hemos elegido un valor de `ActiveProbing.GiveUp` de 5 segundos porque nos parece razonable como tiempo máximo de espera para una aplicación tipo vídeo en streaming, que podría encajar bien con el escenario propuesto para las pruebas.

El tamaño máximo de los segmentos de datos SCTP es 1400, para poder acompañarlos de algunos trozos de control cuando sea necesario dentro del PMTU escogido.

4.6.2 Impacto de un fallo de ruta

Para la simulación de un fallo de ruta en el enlace correspondiente a la interfaz Wi-Fi vamos a utilizar solo dos flujos TCP por cada cliente competidor de forma que las consecuencias del reparto del ancho de banda disponible sean más evidentes.

La asociación SCTP entre C y S se establece en un momento aleatorio entre el segundo 0 y 1 de la simulación. Se van a enviar suficientes datos de S a C (40 MB) como para que la descarga dure 60 segundos de media en condiciones de competencia con los otros flujos TCP.

Las conexiones TCP se inicializan también en un momento aleatorio entre el segundo 0 y 1 de la simulación. Por cada una de ellas se van a transmitir de CS a CC1 y CC2 una cantidad de datos tal, que cada una de ellas dure aproximadamente 3,5 minutos de media (170 MB). De esta manera nos aseguramos que las conexiones TCP no se cierran hasta haberse completado la transmisión SCTP en el peor de los casos (fallo de ruta en el enlace Wi-Fi).

		CC1	CC2	SCTP
Sin fallo	SCTP	238 ± 3	215 ± 8	66 ± 2
	AP	237 ± 3	217 ± 7	67 ± 2
Con fallo	SCTP	221 ± 1	252 ± 2	$\mathbf{226} \pm 6$
	AP	222 ± 2	252 ± 2	$\mathbf{171} \pm 7$

Cuadro 4.1: Tiempo que tardan en transmitirse los datos, en segundos, por las diferentes conexiones TCP y la asociación SCTP/SCTP-AP. Resultados redondeados a un segundo. Cada valor es la media muestral de diferentes simulaciones y los intervalos de confianza corresponden a un índice de confianza del 95 %. Los valores indicados para CC1 y CC2 corresponden a las medias muestrales de la media de los dos flujos TCP de cada uno de los clientes.

El fallo de ruta se va a producir por el enlace Wi-Fi en un momento aleatorio entre el segundo 20 y 40 de la simulación, es decir, entorno a la mitad de la duración de la transmisión de los datos por SCTP.

El cuadro 4.1 muestra el tiempo necesario para terminar todas las descargas. En general se observa que el tiempo necesario para la transmisión de los datos es el mismo tanto si usamos SCTP como si usamos SCTP-AP, excepto en el caso en que suceda un fallo de ruta (resaltado en negrita), donde la transmisión por SCTP-AP concluye unos 55 segundos de media antes que la transmisión por SCTP, tal y como se espera de una pila SCTP-AP configurada con un valor de `ActiveProbing.GiveUp` de 5 segundos, frente a una pila SCTP estándar con un tiempo de PFD de 63 segundos.

Nótese como en el caso de enfrentarnos a un fallo de ruta, los tiempos de transmisión de las conexiones TCP son iguales tanto si usamos SCTP como si usamos SCTP-AP, tal y como era nuestra intención al diseñar SCTP-AP para que tenga el mismo comportamiento TCP-friendly de SCTP.

Si comparamos la duración de las descargas por TCP cuando se sufre un fallo de ruta y cuando no, se observa que las descargas del CC1 terminan antes y que las de CC2 terminan más tarde. La razón de esto es que desde el momento en que sucede el fallo de ruta, el cliente CC1 disfruta de menos competencia y el cliente CC2 de más, al encaminarse los datos SCTP/SCTP-AP por la ruta alternativa.

4.6.3 Impacto de las pérdidas

Las pérdidas por una interfaz pueden hacer que el mecanismo de detección de fallo de ruta se active frecuentemente, lo que resulta ideal para poner de manifiesto inconvenientes imprevistos de nuestra propuesta.

La siguientes simulaciones pretenden verificar el correcto funcionamiento de SCTP-AP en enlaces con pérdidas. En lugar de someter a la interfaz Wi-Fi a un fallo de ruta, veremos como se comporta nuestra propuesta ante diferentes valores de la tasa de pérdidas por esa interfaz.

		CC1	CC2	SCTP
TP=0 %	SCTP	1221 ± 5	1220 ± 6	295 ± 9
	AP	1207 ± 6	1216 ± 6	288 ± 6
TP=0.1 %	SCTP	1225 ± 10	1222 ± 4	335 ± 5
	AP	1207 ± 11	1215 ± 7	331 ± 5
TP=0.5 %	SCTP	1222 ± 9	1221 ± 12	478 ± 10
	AP	1229 ± 15	1216 ± 5	474 ± 21
TP=1 %	SCTP	1226 ± 9	1218 ± 5	602 ± 10
	AP	1212 ± 13	1225 ± 3	611 ± 11
TP=3 %	SCTP	1190 ± 4	1219 ± 5	$\mathbf{1022} \pm 49$
	AP	1179 ± 8	1227 ± 6	$\mathbf{909} \pm 53$

Cuadro 4.2: Tiempo que tardan en transmitirse los datos, en segundos, por las diferentes conexiones TCP y la asociación SCTP/SCTP-AP. Resultados redondeados a segundos. Cada valor es la media muestral de diferentes simulaciones y los intervalos de confianza corresponden a un índice de confianza del 95 %. Los valores indicados para CC1 y CC2 corresponden a las medias muestrales de la media de los cuatro flujos TCP de cada uno de esos clientes.

Durante los proyectos de investigación EW e EW2 tuvimos ocasión de analizar las relaciones entre los diferentes esquemas de modulación Wi-Fi utilizados en algunas tarjetas concretas y la tasa de pérdidas sufridas por los datos (en adelante abreviada como TP). Todas las tarjetas analizadas cambiaban a un esquema de modulación más resistente a interferencias y con mayor sensibilidad al aproximarse el valor de TP al 3 %. Por eso, en estas simulaciones utilizaremos valores de TP por la interfaz Wi-Fi que no superan el 3 %.

El conjunto de valores utilizados en las simulaciones es igual que en el caso anterior, aunque ahora la cantidad de datos a enviar por las conexiones TCP competidoras es sensiblemente superior para acomodar la duración de las transmisiones al caso peor (TP = 3 % en la interfaz Wi-Fi), donde la asociación SCTP/SCTP-AP tarda más en completarse.

Para hacer la simulación más completa, hemos aumentado el número de flujos TCP competidores de los clientes CC1 y CC2 a cuatro por cada uno de ellos y hemos aumentado la duración general de la simulación para obtener datos más fiables en un entorno con elevadas pérdidas aleatorias.

El cuadro 4.2 muestra el tiempo necesario para terminar todas las descargas. Como en el caso anterior, se puede decir que los flujos TCP no se ven afectados por el hecho de competir contra un flujo SCTP o contra uno SCTP-AP, si bien es cierto que se los flujos de CC1 concluyen un poco antes conforme aumentan las pérdidas por la interfaz Wi-Fi, dado que la asociación SCTP/SCTP-AP no puede competir en igualdad de condiciones ante elevadas tasas de pérdidas por la interfaz Wi-Fi.

Dado que la asociación SCTP/SCTP-AP no llega a realizar ningún traspaso, los flujos del cliente CC2 presentan un comportamiento idéntico e independiente

del valor de la tasa de pérdidas por la interfaz Wi-Fi.

Si nos centramos en el comportamiento de los flujos SCTP/SCTP-AP, observamos que el tiempo necesario para terminar la descarga aumenta conforme crece la tasa de pérdidas por la interfaz Wi-Fi, como era de esperar.

Se puede observar como las prestaciones de SCTP y SCTP-AP son similares, excepto en el caso de la tasa de pérdidas más elevada, donde se puede identificar una disminución de las prestaciones de SCTP-AP frente a SCTP (resaltado en negrita).

Es difícil sacar conclusiones mas concretas sobre este caso particular ya que la tasa de pérdidas es tan elevada que el comportamiento de cada simulación presenta una alta varianza. Nos inclinamos a pensar que la menor agresividad de SCTP-AP empieza a tomar importancia en esas circunstancias, pues las pérdidas se hacen muy habituales y SCTP-AP tiende a enviar los datos en ráfagas más cortas que SCTP por la disminución de la ventana de pérdidas. En todo caso, si se puede concluir que para valores de pérdidas más razonables, las prestaciones de SCTP y SCTP-AP son las mismas.

4.7 Conclusiones

En este capítulo proponemos SCTP-AP, una modificación a SCTP, que permite adaptar la duración de la detección de fallo de ruta a las necesidades de las diferentes aplicaciones.

Este nuevo mecanismo de detección de fallo de ruta se basa en el envío de sondas activas por la ruta sospechosa. Hemos tomado medidas para asegurar la estabilidad del protocolo a cambio de una disminución de la agresividad en el envío de datos por la ruta que está siendo sondeada. Esta menor agresividad no tiene efectos adversos apreciables para tasas de pérdidas hasta del 1 %.

Nuestras simulaciones muestran que SCTP-AP permite adaptar la duración de la detección del fallo de ruta a las aplicaciones de forma configurable, evitando los compromisos y problemas de otras propuestas previas.

La solución propuesta también se recupera antes de fallos de ruta temporales y permite,completar la transferencia de los datos antes que SCTP, independientemente de la duración del fallo de ruta temporal.

Capítulo 5

Estudio de estrategias de observación

5.1 Motivación y contenidos

El principal problema de SCTP como solución de movilidad en dispositivos multiconectados es que tarda demasiado en detectar fallos de ruta.

SCTP necesita recolectar indicios suficientes de que un fallo de ruta es permanente para evitar traspasos innecesarios derivados de fallos de ruta temporales. El tiempo dedicado por SCTP a recopilar estos indicios es demasiado largo para algunas aplicaciones modernas, en especial para aplicaciones interactivas con el usuario.

Por eso se dice que el mecanismo de detección de fallo de ruta de SCTP es demasiado lento ([54]).

Una de las aportaciones de esta tesis es proponer un cambio en el mecanismo de detección de fallo de ruta para agilizarlo, de forma que SCTP reaccione con la rapidez requerida por cada aplicación (§4).

Otro problema que se deriva de cómo SCTP recopila indicios sobre el fallo de ruta se manifiesta cuando nos enfrentamos a un fallo de ruta temporal y aceptable, que no debería desencadenar un traspaso.

En especial, en redes inalámbricas, es habitual tener ráfagas de pérdidas que duran tan sólo unos segundos, debido a que el dispositivo sufre, durante un periodo corto de tiempo, un empeoramiento de la cobertura. Por ejemplo, cuando el usuario del dispositivo gira una esquina, de repente tendremos varias paredes interrumpiendo la línea de visión con el receptor inalámbrico. En otras ocasiones un obstáculo móvil especialmente opaco interrumpe la línea de visión entre el dispositivo y el receptor, por ejemplo cuando un vagón de tren o un camión cargado de mercancías pasan entre el dispositivo y el receptor.

En estas situaciones, SCTP demora innecesariamente la transmisión de datos por la ruta, aún cuando esta ya se ha recuperado del fallo temporal, lo que resulta en un peor aprovechamiento del canal y ralentiza, sin motivo, la transmisión de

los datos.

Esto es debido a la planificación temporal de las sondas que SCTP usa para inspeccionar el estado de la ruta bajo sospecha: SCTP emite las sondas siguiendo una estrategia geométrica en el tiempo, cada sonda se manda pasado el doble de tiempo que se esperó para enviar la sonda anterior.

Como veremos en este capítulo, esta estrategia no es la óptima para detectar un fallo de ruta temporal cuando desconocemos la naturaleza de este fallo y su duración estimada. Además impone un límite insalvable a la rapidez con que SCTP reacciona ante estos fenómenos temporales.

Las modificaciones que proponemos en esta tesis para agilizar la detección de fallo de ruta en SCTP (§4) nos permiten modificar la estrategia de observación a nuestro gusto, por lo que sería bueno descubrir qué estrategias de observación son mejores que la geométrica e incorporarlas a nuestra propuesta general.

En el intento de encontrar la estrategia óptima de observación para SCTP hemos desarrollado contribuciones al estudio de estrategias de observación desde un punto de vista genérico, de aplicación global más allá del ámbito de la telemática, que detallaremos a lo largo de este capítulo y que resumimos a continuación:

- **Propuesta de unos fundamentos matemáticos para el estudio y análisis de estrategias de observación** (§5.2), que incluye las definiciones y la notación necesaria para entender el resto del capítulo. Esto es necesario para avalar con el rigor necesario nuestras propuestas posteriores. No hemos encontrado trabajos previos que nos permitieran ahorrarnos este paso inicial.

- **Definición del *tiempo medio de detección*** (§5.3) como la figura de mérito fundamental a la hora de decidir cúal es la estrategia de observación más apropiada.

- **Análisis de prestaciones de la estrategia periódica** (§5.4). Esto tiene un doble propósito: por un lado, la estrategia periódica de observación es la mas sencilla de analizar, por lo que resulta un buen ejemplo de aplicación de los fundamentos matemáticos propuestos en las secciones anteriores.

  Por otro lado, la estrategia periódica resulta especialmente importante para el problema que nos ocupa, como se demostrará posteriormente, por lo que merece un espacio exclusivo para su análisis en profundidad.

- **Análisis de prestaciones de la estrategia geométrica** (§5.5). SCTP usa una estrategia geométrica, por lo que deducir las expresiones analíticas de sus prestaciones resulta conveniente para luego poder compararla con el resto de estrategias.

- **Comparativa entre la estrategia periódica y geométrica** (§5.6). Donde se demuestra que la estrategia periódica es más apropiada que la geométrica para la detección de fallos de ruta temporales cuando se desconocen la naturaleza y duración de estos fallos. En esta sección también se

identifican las limitaciones intrínsecas derivadas de utilizar una estrategia geométrica.

- **Demostración de que la estrategia periódica es la mejor estrategia posible** (§5.7) cuando se desconocen la naturaleza y duración de los fallos de ruta temporales.
- **Impacto de añadir variaciones aleatorias a la planificación de las sondas** (§5.8). Este análisis resulta especialmente útil en protocolos de comunicaciones.
- **Representación en un plano de estrategias de sondeo** (§5.9), que permite comparar estrategias entre si, de forma visual y sencilla.

La conclusión de este capítulo es que la propuesta de modificación de SCTP que presentamos en el capítulo §4 debería utilizar una estrategia de observación periódica por ser la óptima para un protocolo de propósito general que quiera detectar con rapidez los fallos de ruta temporales.

5.2 Definiciones y nomenclatura

Las siguientes definiciones resultan útiles para manejar matemáticamente estrategias de observación.

Al no haber encontrado trabajo previo relacionado con este tema, decidimos comenzar por establecer una nomenclatura y definiciones generales, aplicables a cualquier problema relacionado con estrategias de observación.

Los primeros intentos de aplicar esta formulación matemática al contexto particular de esta tesis pusieron de manifiesto que algunas representaciones alternativas ayudaban o resultaban más convenientes a la hora de manejar matemáticamente ciertos conceptos, de forma que fuimos añadiendo definiciones adicionales a la estructura básica.

Lo que se presenta a continuación es el resultado de este proceso iterativo, con una ordenación que no tiene que ver con la cronología de su concepción sino con la sencillez de su exposición.

5.2.1 La función mitad triangular derecha

Definición 5.2.1. *La* **función mitad triangular derecha** $\mathbf{\Gamma} : \mathbb{R} \to \mathbb{R}$*, es el producto de la función triangular* $\wedge$ *y la función escalón de Heaviside*[1] *en su versión continua por la derecha,* H*.*

La figura 5.1 muestras las gráficas de $\wedge(x)$, $H(x)$ y $\Gamma(x)$.

La función Γ también puede expresarse como

$$\Gamma(x) \doteq \begin{cases} 1 - x, & \text{si } 0 \leq x < 1 \\ 0, & \text{resto,} \end{cases} \tag{5.1}$$

[1] También conocida como la función escalón unidad.

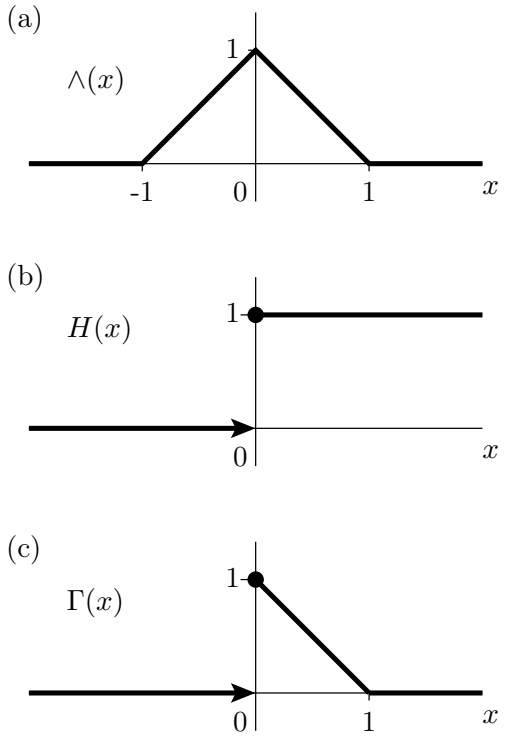

Figura 5.1: (a) La función triangular $\wedge(x)$. (b) La función escalón de Heaviside, en su versión continua por la derecha $H(x)$. (c) La función mitad triangular derecha $\Gamma(x) = \wedge(x)H(x)$.

lo que permite manejarla analíticamente de forma sencilla.

Nótese como la función Γ es continua a tramos y en particular es continua en el intervalo $[0, 1)$.

En las siguientes secciones manejaremos a menudo versiones ensanchadas y desplazadas de la función Γ. Resulta conveniente definir una notación específica para indicar estas variantes tan comunes:

Definición 5.2.2. *Sean dos números reales α y β, con $\beta \neq 0$, la* **función mitad triangular derecha generalizada $\boldsymbol{\Gamma\left(\frac{x-\alpha}{\beta}\right)}$** *es el resultado de ensanchar la función Γ por un factor β y posteriormente desplazarla una cantidad α a la derecha*

La figura 5.2 muestra una gráfica de la función mitad triangular derecha generalizada, $\Gamma\left(\frac{x-\alpha}{\beta}\right)$.

La expresión analítica de esta función es

$$\Gamma\left(\frac{x-\alpha}{\beta}\right) \doteq \begin{cases} 1 - \frac{x-\alpha}{\beta}, & \text{si } \alpha \leq x < (\alpha + \beta) \\ 0, & \text{resto.} \end{cases} \tag{5.2}$$

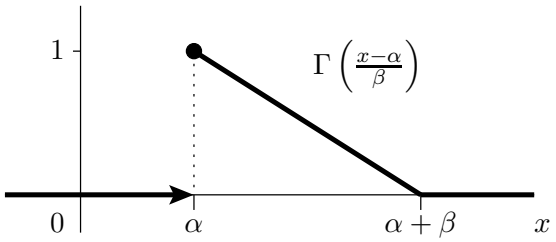

Figura 5.2: La función mitad triangular derecha generalizada $\Gamma\left(\frac{x-\alpha}{\beta}\right)$ es el resultado de ensanchar la función $\Gamma(x)$ por β y desplazar el resultado a la derecha una cantidad α.

Nótese como la función $\Gamma\left(\frac{x-\alpha}{\beta}\right)$ es también continua a tramos y en particular es continua en el intervalo $[\alpha, \alpha+\beta)$.

Mas tarde, en este capítulo necesitaremos calcular el área de la función mitad triangular derecha generalizada, adelantaremos trabajo si la calculamos con antelación. Una interpretación geométrica de la figura 5.2 nos adelanta que el cálculo será sencillo:

Lema 5.2.1. *El área bajo* $\Gamma\left(\frac{x-\alpha}{\beta}\right)$ *es* $\frac{\beta}{2}$.

Demostración. El área bajo $\Gamma\left(\frac{x-\alpha}{\beta}\right)$ es su integral sobre su dominio $\mathbb{R}$. Usando la definición 5.2 para la integral y haciendo el cambio de variable[2] $y = x - \alpha$ demostramos el lema:

$$\int_{\mathbb{R}} \Gamma\left(\frac{x-\alpha}{\beta}\right) dx = \int_{\alpha}^{\alpha+\beta} \left(1 - \frac{x-\alpha}{\beta}\right) dx$$

$$= \int_0^{\beta} \left(1 - \frac{y}{\beta}\right) dy = \beta - \int_0^{\beta} \frac{y}{\beta}\, dy = \beta - \left.\frac{y^2}{2\beta}\right|_o^{\beta} = \frac{\beta}{2}$$

□

5.2.2 Representando eventos mediante variables aleatorias

En el contexto de la detección de fallos de ruta en SCTP, se aprecia un experimento estadístico: la posible recuperación de una ruta al poco de producirse el fallo de ruta.

En este mismo contexto, se puede resumir el comportamiento general de la detección de fallo de ruta de SCTP de la siguiente manera: digamos que el fallo de ruta ocurre en el momento 0. Si la recuperación de la ruta tiene lugar en el

[2]Todos los cambios de variable que aparecen en este documento tienen derivada continua, por lo que son aptos para resolver integrales.

intervalo $[0, T)$, siendo T la duración de la detección de fallo de ruta en SCTP, la recuperación será detectada y la transmisión se retomará normalmente por dicha ruta.

Por contra, si la recuperación de la ruta sucede en el momento T o más tarde, la recuperación no será detectada, ya que para SCTP una ruta que falla durante T o más segundos representa un fallo de ruta permanente.

Definición 5.2.3. *Sea $\mathscr{E}$ un experimento estadístico con un espacio muestral $[0, T)$. Sea* $\mathbf{t}$ *la variable aleatoria continua que describe los resultados de dicho experimento. Sea* $\boldsymbol{f_t(t)}$ *la función de densidad de probabilidad de* t.

Siguiendo con el paralelismo con SCTP, el experimento $\mathscr{E}$ es la ocurrencia de una recuperación de ruta que ha fallado previamente. La variable aleatoria t representará el tiempo en que esta recuperación tiene lugar.

La función de densidad de probabilidad $f_{\mathrm{t}}(t)$ dependerá enteramente de la naturaleza del fallo de ruta original, las tecnologías de transmisión involucradas y las circunstancias particulares a la que la red está expuesta en esos momentos.

Por ejemplo, si el fallo de ruta es debido a una ráfaga de pérdidas en un enlace inalámbrico, probablemente $f_{\mathrm{t}}(t)$ será casi cero para valores de t cercanos al momento del fallo y tendrá un máximo cerca de la duración media de las ráfagas de pérdidas de esa tecnología y en las circunstancias dadas.

En cambio, *si realmente no supiéramos nada sobre el experimento o no hubiese ninguna razón particular para pensar que la recuperación va a tener lugar con mayor probabilidad en un momento u otro, entonces una estrategia minimax sugiere definir $f_{\mathrm{t}}(t)$ como una uniforme.* Para el diseño de protocolos de propósito general, como es el objetivo de esta tesis, está resulta ser la suposición apropiada.

En una aplicación más general de estas técnicas, el experimento puede tener un espacio muestral mayor que $[0, T)$. Para estos casos, recomendamos recortar el espacio muestral a $[0, T)$ y recalcular la correspondiente $f_{\mathrm{t}}(t)$ usando técnicas de probabilidad condicional.

De acuerdo con la definición 5.2.3, el dominio y el rango de t es $[0, T)$ y $f_{\mathrm{t}}(t)$ tiene las siguientes propiedades:

$$f_{\mathrm{t}}(t) = \begin{cases} 0, & t < 0 \\ \geq 0, & 0 \leq t < T \\ 0, & t \geq T. \end{cases} \tag{5.3}$$

Es conveniente utilizar el valor de T más pequeño posible, dentro de lo que permita la naturaleza del experimento, para evitar buscar resultados del experimento más allá del tiempo máximo en que sabemos que este puede ocurrir o que es relevante que ocurra.

Basándonos en esta recomendación podemos añadir una definición adicional sobre el suceso, que nos resultará muy conveniente:

Definición 5.2.4. *El* ***periodo de ocurrencia*** *del experimento $\mathscr{E}$ es la longitud del dominio de* t*, es decir, T.*

De ahora en adelante, utilizaremos la notación $\mathscr{E}^T$ para referirnos a experimentos con periodo de ocurrencia T.

5.2.3 Estrategias de observación

Definición 5.2.5. *Una **observación** permite conocer si el evento ha ocurrido en el pasado.*

Por ejemplo, ante un experimento $\mathscr{E}^T$, una observación en el momento $T/2$ detectará si el evento ha ocurrido en el intervalo $[0, T/2)$, en otras palabras, una observación en $T/2$ descubrirá si la realización de t ha tomado un valor menor a $T/2$.

En el contexto de SCTP, una sonda enviada a la ruta bajo sospecha en el momento $T/2$, por ejemplo, nos permitirá saber si la ruta se ha recuperado ya o todavía no.

En general, realizaremos varias observaciones, en momentos diferentes, para conseguir la precisión deseada en la detección del evento, es decir, para detectarlo lo antes posible.

Definición 5.2.6. *Una **estrategia** de observación $\{t_i\}_N$ de un experimento $\mathscr{E}^T$ es una secuencia positiva, creciente y finita de los tiempos en que se planean realizar N observaciones del experimento. En forma analítica:*

$$\{t_i\}_N = \{t_i \ en \ \mathbb{R} : 0 < t_i \leq T\},$$

con $i = 1, \ldots, N$ y $t_i < t_{i+1}$.

En otras palabras, una estrategia de observación, o simplemente una estrategia, es la secuencia de observaciones que se van a realizar a lo largo del periodo de ocurrencia del experimento.

Por ejemplo, si planeamos enviar 3 sondas a los 1, 5 y 7 segundos, la estrategia será $\{t_i\}_3 = \{1, 5, 7\}$.

Una estrategia de observación puede definirse también mediante el tiempo en que ocurre la primera observación y los tiempos de espera entre observaciones sucesivas:

Definición 5.2.7. *Los **subintervalos** de observación $\{\Delta_i\}_N$ de una estrategia $\{t_i\}$ son la secuencia*

$$\{\Delta_i\}_N \doteq \begin{cases} t_1, & i = 1 \\ t_i - t_{i-1}, & resto. \end{cases} \tag{5.4}$$

Una estrategia de observación se puede definir por completo tanto mediante su secuencia de observaciones $\{t_i\}_N$ como por sus subintervalos $\{\Delta_i\}_N$, dado que

Lema 5.2.2. *El momento en que una observación tiene lugar es la suma de sus subintervalos previos,*

$$t_i = \sum_{j=1}^{i} \Delta_j. \tag{5.5}$$

Demostración. Sustituyendo directamente la definición (5.4),

$$\sum_{j=1}^{i}\Delta_j = \Delta_1 + \Delta_2 + \cdots + \Delta_i = t_1 + t_2 - t_1 + \cdots + t_i - t_{(i-1)} = t_i.$$

□

En las siguientes secciones, utilizaremos tanto la secuencia $\{t_i\}_N$ como los subintervalos de observación $\{\Delta_i\}_N$, para referirnos a una estrategia de observación concreta, dependiendo de en qué forma los cálculos resulten más sencillos.

Definición 5.2.8. *Una **estrategia de observación fiable** es una estrategia que permite detectar la ocurrencia del evento para todos los posibles resultados del experimento.*

Por ejemplo, si tenemos un experimento con un periodo de ocurrencia $T = 5$, pero nuestra estrategia de observación es $\{1, 2, 3\}$, cualquier realización del experimento en el rango $[3, 5)$ pasará inadvertida, y diremos que nuestra estrategia de observación no es fiable.

Lema 5.2.3. *Una estrategia es fiable si y sólo si $t_N = T$.*

Demostración. De acuerdo con la definición (5.2.6), $t_N \leq T$. Por otro lado, la restricción (5.3) establece que $f_{\mathrm{t}}(t) \geq 0$ para $0 \leq t < T$.

Si t_N fuera menor que T, entonces la probabilidad de $\mathrm{t} \geq t_N$ sería mayor que 0, lo que querría decir que algunas de las realizaciones del experimento no se detectarían por ocurrir después de la última observación.

La única manera de garantizar que todas las posibles realizaciones del experimento sean detectadas es programando la última observación al final del periodo de ocurrencia, es decir, $t_N = T$. □

La subfigura 5.3a muestra un ejemplo de estrategia fiable de 3 observaciones.

Lema 5.2.4. *La suma de los subintervalos de una estrategia fiable es el periodo de ocurrencia.*

Demostración. Usando el lema 5.2.3 y el lema 5.2.2, la suma de los subintervalos es

$$\sum_{i=1}^{N}\Delta_i = t_N = T.$$

□

Esto significa que los subintervalos de una estrategia fiable cubren la totalidad del periodo de ocurrencia del experimento.

En la práctica sólo consideraremos estrategias fiables, ya que en general, el caso en que un evento ocurra después de la última observación, será indistinguible del caso en que el experimento no se haya realizado jamás, puesto que no se podrá detectar mediante la estrategia utilizada.

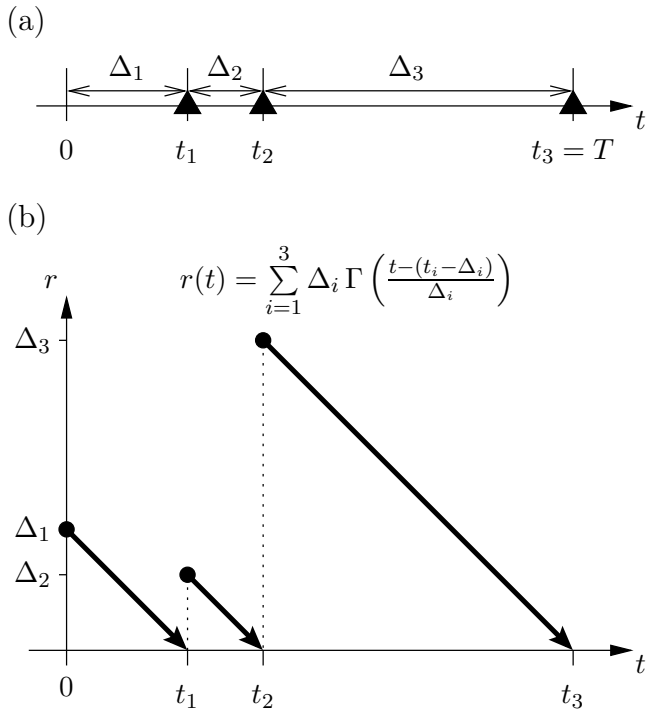

Figura 5.3: (a) Ejemplo de estrategia fiable de 3 observaciones. Cada observación se representa mediante un triángulo negro, en los momentos t_1, t_2 y t_3. (b) Retardo de detección de la estrategia mostrada en la subfigura a.

5.2.4 Retardo de detección

Dependiendo de la estrategia de observación que se use y el momento concreto del tiempo en que ocurre el evento, se incurrirá en un cierto retardo de detección. *Por ejemplo, si el evento ocurre en el momento $t' = 3$ segundos y la estrategia de observación es $\{t_i\}_3 = \{2, 5, 10\}$ segundos, entonces el evento será detectado por la segunda observación con un retraso de 2 segundos ($t_2 - t' = 5 - 3 = 2$ segundos).*

Usaremos este retardo de detección como la figura de mérito principal para comparar estrategias entre sí, ya que nuestro objetivo es encontrar la estrategia que menos tarde en detectar un suceso. *Retomando el ejemplo anterior, una estrategia diferente, con observaciones en $\{t_i\}_3 = \{2, 4, 10\}$ segundos, también detectará el evento con la segunda observación, pero con sólo un segundo de retraso.*

Como los eventos ocurren aleatoriamente, el retardo de detección es también una variable aleatoria, y tendremos que usar sus propiedades estadísticas para poder comparar estrategias entre si:

Definición 5.2.9. *El **retardo de detección** **r** de una estrategia de observación*

es la variable aleatoria que representa el tiempo transcurrido entre la ocurrencia del evento y la primera observación posterior a él.

El retardo de detección $r(t)$ puede expresarse analíticamente como:

$$r(t) = t_j - t,$$

donde j es el índice de la primera observación posterior a la ocurrencia del evento t:

$$j(t) = \underset{i}{\text{argmín}}(t_i),\ \forall t_i > t.$$

Por lo tanto, el dominio del retardo de detección r es igual al rango de t, es decir, $[0, T)$.

La figura 5.3b muestra el retardo de detección $r(t)$ que le correspondería a la estrategia de ejemplo representada en la figura 5.3a. Tal y como sugiere esta figura, $r(t)$ puede definirse como la suma de N funciones mitad triangular derecha, cada una de las cuales está

1) amplificada por Δ_i,

3) ensanchada por Δ_i,

2) y desplazada a la derecha la cantidad $t_i - \Delta_i$.

Por tanto, una expresión concisa del retardo de detección sería:

$$r(t) \doteq \sum_{i=1}^{N} \Delta_i \, \Gamma\left(\frac{t - (t_i - \Delta_i)}{\Delta_i}\right). \tag{5.6}$$

En las siguientes secciones necesitaremos a menudo conocer el área del retardo de detección $r(t)$, por lo que resulta apropiado calcularla por adelantado:

Lema 5.2.5. *El área R bajo el retardo de detección $r(t)$ de una estrategia de observación $\{\Delta_i\}_N$ es*

$$R = \frac{1}{2} \sum_{i=1}^{N} \Delta_i^2. \tag{5.7}$$

Esto significa que R es la suma de las áreas del tren de pulsos triangulares que forman el retardo de detección $r(t)$, como era de esperar.

Demostración. Usando la definición (5.6), R es

$$R = \int_{\mathbb{R}} r(t)\, dt = \int_0^T r(t)\, dt = \int_0^T \sum_{i=1}^{N} \Delta_i \, \Gamma\left(\frac{t - (t_i - \Delta_i)}{\Delta_i}\right) dt.$$

Como $r(t)$ es continua a tramos en $[t_i-\Delta_i, t_i)$, es integrable en esos intervalos. Por tanto, la integral y la suma pueden intercambiarse[3]:

$$R = \sum_{i=1}^{N} \Delta_i \int_{t_i-\Delta_i}^{t_i} \Gamma\left(\frac{t-(t_i-\Delta_i)}{\Delta_i}\right) dt,$$

que puede resolverse sustituyendo el resultado del lema 5.2.1,

$$R = \sum_{i=1}^{N} \Delta_i \frac{\Delta_i}{2} = \frac{1}{2}\sum_{i=1}^{N} \Delta_i^2.$$

□

5.3 Retardo medio de detección

Teorema 5.3.1. *Sea $f_{\mathrm{t}}(t)$ la función de densidad de probabilidad del experimento $\mathscr{E}^T$. El* ***retardo medio de detección*** $\bar{r}$ *de una estrategia de observación $\{t_i\}_N$ es*

$$\bar{r} = \sum_{i=1}^{N} \Delta_i \int_{t_i-\Delta_i}^{t_i} \Gamma\left(\frac{t-(t_i-\Delta_i)}{\Delta_i}\right) f_{\mathrm{t}}(t)\, dt, \tag{5.8}$$

donde $\{\Delta_i\}$ son los subintervalos de observación de la estrategia.

Demostración. Según la ley del estadístico inconsciente ([85]), el valor medio de la variable aleatoria r es

$$\bar{r} = \int_{-\infty}^{\infty} r(t) f_{\mathrm{t}}(t)\, dt.$$

Usando la definición (5.6), tenemos que

$$\bar{r} = \int_0^T \sum_{i=1}^{N} \Delta_i\, \Gamma\left(\frac{t-(t_i-\Delta_i)}{\Delta_i}\right) f_{\mathrm{t}}(t)\, dt.$$

Cada elemento de la suma es integrable, por tanto la suma y la integral pueden ser intercambiadas, es decir

$$\bar{r} = \sum_{i=1}^{N} \Delta_i \int_0^T \Gamma\left(\frac{t-(t_i-\Delta_i)}{\Delta_i}\right) f_{\mathrm{t}}(t)\, dt.$$

Como la función mitad triangular derecha generalizada $\Gamma\left(\frac{t-\alpha}{\beta}\right)$ es cero fuera de $[\alpha, \alpha+\beta)$ (recordemos la figura 5.2), entonces

$$\bar{r} = \sum_{i=1}^{N} \Delta_i \int_{t_i-\Delta_i}^{t_i} \Gamma\left(\frac{t-(t_i-\Delta_i)}{\Delta_i}\right) f_{\mathrm{t}}(t)\, dt.$$

□

[3]Cada elemento de la suma está acotado y es continuo excepto en un conjunto de puntos de medida cero, por tanto, cumplen el criterio de Lebesgue para la integrabilidad de Riemann.

Como ya hemos mencionado anteriormente, los experimentos con funciones de densidad de probabilidad uniforme son especialmente interesantes para el diseño de protocolos de propósito general. A continuación explicamos cómo particularizar el anterior resultado para estos casos.

Lema 5.3.2. *El retardo medio de detección de una estrategia, cuando la función de densidad de probabilidad del experimento es uniforme en* $[0, T)$ *es*

$$\bar{r}_{uni} = \frac{R}{T} \tag{5.9}$$

Demostración. Un experimento con una función de densidad de probabilidad uniforme en $[0, T)$ está definida por $f_{\mathrm{t}}(t) = 1/T$ en ese intervalo y $f_{\mathrm{t}}(t) = 0$ en el resto. A partir del lema 5.3.1, el retardo medio de detección sería

$$\begin{aligned}\bar{r}_{\mathrm{uni}} &= \sum_{i=1}^{N} \Delta_i \int_{t_i-\Delta_i}^{t_i} \Gamma\left(\frac{t-(t_i-\Delta_i)}{\Delta_i}\right)\frac{1}{T}\,dt \\ &= \frac{1}{T}\sum_{i=1}^{N} \Delta_i \int_{t_i-\Delta_i}^{t_i} \Gamma\left(\frac{t-(t_i-\Delta_i)}{\Delta_i}\right)\,dt.\end{aligned}$$

Podemos utilizar el lema 5.2.5 para convertir la suma de integrales en el área total del retardo, por lo que

$$\bar{r}_{\mathrm{uni}} = \frac{1}{2T}\sum_{i=1}^{N} \Delta_i^2 = \frac{R}{T}.$$

□

De ahora en adelante, nos referiremos a los experimentos con función de densidad de probabilidad uniforme en $[0, T)$ como experimentos uniformes o más concisamente $\mathscr{E}_{\mathrm{uni}}^{T}$.

5.4 Análisis de las estrategias periódicas

Una estrategia de observación periódica es una estrategia cuyas observaciones se realizan a intervalos regulares: el retardo entre una observación y la siguiente es siempre el mismo. La figura 5.4a muestra un ejemplo de estrategia de observación periódica.

Por definición, las estrategias periódicas cumplen las siguientes propiedades:

1.
$$t_i^{(\mathrm{per})} \doteq i\frac{T}{N}, \tag{5.10}$$

2.
$$\Delta_i^{(\mathrm{per})} \doteq \frac{T}{N}. \tag{5.11}$$

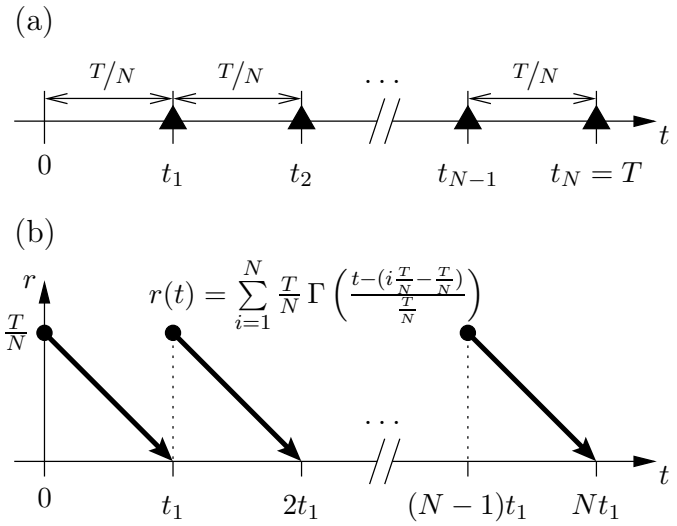

Figura 5.4: (a) Un ejemplo de una estrategia de observación periódica de N observaciones: el retraso entre una observación y la siguiente es $\frac{T}{N}$. (b) Retardo de detección de la estratégia periódica del ejemplo mostrado en la subfigura a.

Lema 5.4.1. *El área del retardo de detección de una estrategia periódica es*

$$R^{(per)} = \frac{T^2}{2N}. \tag{5.12}$$

Demostración. Utilizando la propiedad (5.11) y la expresión (5.7):

$$R^{(\mathrm{per})} = \frac{1}{2}\sum_{i=1}^{N}\left(\frac{T}{N}\right)^2 = \sum_{i=1}^{N}\frac{T^2}{2N^2} = \frac{T^2}{2N}.$$

□

La figura 5.4b muestra el retardo de detección derivado de utilizar una estrategia de observación periódica.

Como veremos en §5.7, las estrategias de observación periódicas resultan especialmente interesantes para la detección de experimentos uniformes, por lo que resulta apropiado adelantar ahora algunos resultados:

Teorema 5.4.2. *El retardo medio de detección de un experimento uniforme $\mathscr{E}^T_{uni}$ usando una estrategia de observación periódica de N observaciones es*

$$\bar{r}^{(per)}_{uni} = \frac{T}{2N}. \tag{5.13}$$

Este teorema se demuestra fácilmente combinando el lema 5.4.1 y el lema 5.3.2.

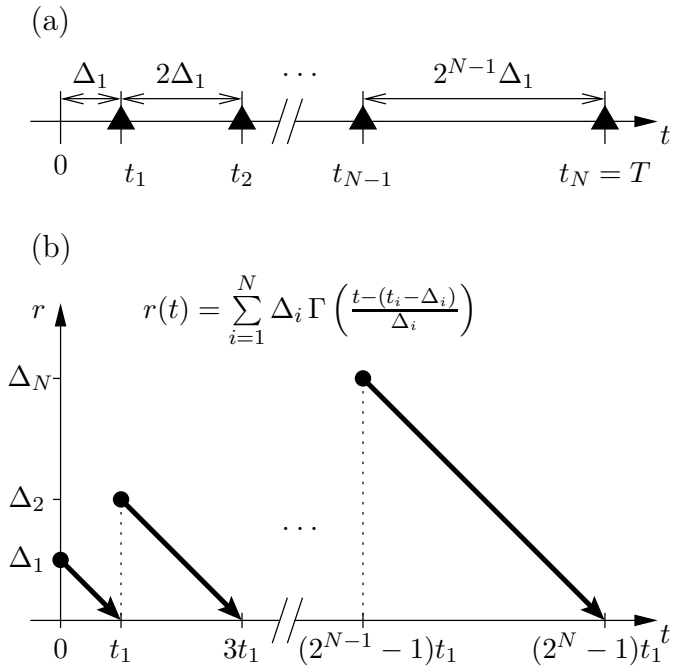

Figura 5.5: (a) Una estrategia de observación geométrica. El retraso entre observaciones se dobla con cada observación. (b) Retardo de detección de la estratégia geométrica de la subfigura a.

Nótese que el teorema 5.4.2 indica que se puede reducir el retardo medio de detección de una estrategia periódica tanto como se desee, simplemente incrementando el número de observaciones. No todas las estrategias de observación tienen esta interesante propiedad, como veremos en la siguiente sección.

5.5 Análisis de las estrategias geométricas

Como ya hemos mencionado anteriormente, el mecanismo de detección de fallo de ruta de SCTP utiliza una estrategia de observación geométrica.

Las estrategias geométricas se caracterizan en que el tiempo entre una observación y la siguiente se va duplicando con cada observación. La figura 5.5a muestra un ejemplo de estrategia geométrica.

Por definición, las estrategias geométricas cumplen la siguiente propiedad:

$$\Delta_i^{(geo)} \doteq 2^{i-1}\Delta_1^{(geo)}. \tag{5.14}$$

Por ejemplo, una estrategia geométrica cuya primera observación se realiza en el momento $t_1 = 1$ segundos ($\Delta_1 = 1$ segundos), tendrá la segunda observación 2 segundos más tarde, es decir en $t_2 = 3$ segundos. La tercera observación tendrá lugar 4 segundos después, en $t_3 = 7$, y así sucesivamente.

Lema 5.5.1. *La primera observación de una estrategia geométrica tiene lugar en*

$$t_1^{(geo)} = \Delta_1^{(geo)} = \frac{T}{2^N - 1}$$

Demostración. Por sencillez, omitiré el superíndice (geo) en las demostraciones.

En una estrategia fiable, el lema 5.2.4 asegura que

$$T = \sum_{i=1}^{N} \Delta_i.$$

Sustituyendo la definición (5.14) y resolviendo la suma geométrica resultante, tenemos que

$$T = \sum_{i=1}^{N} 2^{i-1}\Delta_1 = (2^N - 1)\Delta_1,$$

por tanto,

$$\Delta_1 = \frac{T}{2^N - 1}.$$

Además, según la definición (5.4), $t_1 = \Delta_1$. □

Lema 5.5.2. *En una estrategia de observación geométrica, las observaciones se realizan en los momentos*

$$t_i^{(geo)} = (2^i - 1)\frac{T}{2^N - 1}.$$

Por ejemplo, una estrategia de observación geométrica de $N = 6$ observaciones, para un experimento con un periodo de ocurrencia $T = 63$ segundos, tendrá sus observaciones en $\{t_i\}_6 = \{1, 3, 7, 15, 31, 63\}$ segundos.

Demostración. Combinando el lema 5.2.2 y la definición (5.14), tenemos que

$$t_i = \sum_{j=1}^{i} \Delta_j = \sum_{j=1}^{i} 2^{j-1}\Delta_1.$$

Utilizando el lema 5.5.1 y resolviendo la suma geométrica, tenemos que

$$t_i = \frac{T}{2^N - 1}\sum_{j=1}^{i} 2^{j-1} = (2^i - 1)\frac{T}{2^N - 1}.$$

□

Lema 5.5.3. *Los subintervalos de una estrategia geométrica son*

$$\Delta_i^{(geo)} = 2^{i-1}\frac{T}{2^N - 1}. \tag{5.15}$$

Demostración. La ecuación (5.15) es el resultado de sustituir el lema 5.5.1 en la definición (5.14), pero puede demostrarse con sencillez también calculando $\Delta_i = t_i - t_{i-1}$ con la ayuda del lema 5.5.2:

$$\begin{aligned}\Delta_i = t_i - t_{i-1} &= (2^i - 1)\frac{T}{2^N - 1} - (2^{i-1} - 1)\frac{T}{2^N - 1} \\ &= (2^i - 1 - 2^{i-1} + 1)\frac{T}{2^N - 1} \\ &= (2 \cdot 2^{i-1} - 2^{i-1})\frac{T}{2^N - 1} \\ &= 2^{i-1}\frac{T}{2^N - 1}.\end{aligned}$$

La demostración para el caso $i = 1$ está implícita en el lema 5.5.1. □

Siguiendo con el ejemplo anterior ($T = 63$ segundos y $N = 6$ observaciones), los subintervalos de $\{t_i\}_6 = \{1, 3, 7, 15, 31, 63\}$ serán $\{\Delta_i\}_6 = \{1, 2, 4, 8, 16, 32\}$ segundos.

Lema 5.5.4. *El área del retardo de detección de una estrategia geométrica es*

$$R^{(geo)} = \frac{T^2(2^{2N} - 1)}{6\,(2^N - 1)^2}. \tag{5.16}$$

Demostración. Utilizando la propiedad (5.15) en (5.7),

$$\begin{aligned}R &= \frac{1}{2}\sum_{i=1}^{N}\left(2^{i-1}\frac{T}{2^N - 1}\right)^2 \\ &= \frac{1}{2}\left(\frac{T}{2^N - 1}\right)^2 \sum_{i=1}^{N}\left(2^{i-1}\right)^2 \\ &= \frac{1}{2}\left(\frac{T}{2^N - 1}\right)^2 \sum_{i=1}^{N} 2^{2i-2}.\end{aligned}$$

Resolviendo la suma geométrica,

$$\begin{aligned}R &= \frac{1}{2}\left(\frac{T}{2^N - 1}\right)^2 \frac{2^{2N} - 1}{3} \\ &= \frac{T^2(2^{2N} - 1)}{6\,(2^N - 1)^2}.\end{aligned}$$

□

Teorema 5.5.5. *El retardo medio de detección de un experimento uniforme $\mathscr{E}^T_{uni}$ utilizando una estrategia geométrica de N observaciones es*

$$\bar{r}^{(geo)}_{uni} = \frac{T(2^{2N} - 1)}{6\,(2^N - 1)^2}. \tag{5.17}$$

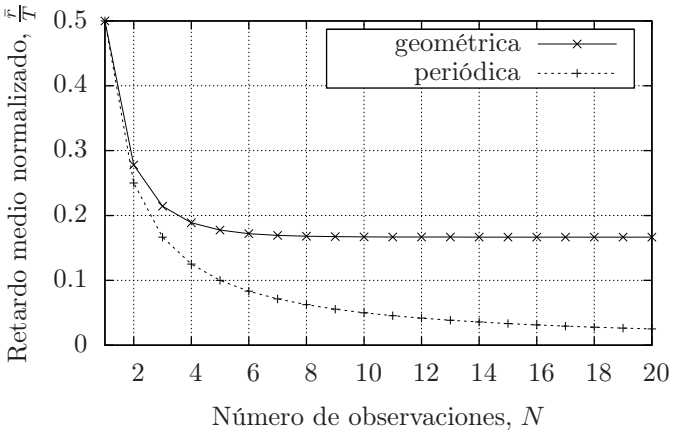

Figura 5.6: Retardo medio de detección de la estrategia periódica y geométrica ante experimentos uniformes. El retardo medio de detección se muestra normalizado por el periodo de ocurrencia T del experimento, para mayor generalización.

Demostración. Por sustitución directa de (5.16) en (5.9). □

Nótese que $\bar{r}_{\text{uni}}^{(geo)}$ tiene una asíntota horizontal en $T/6$, que es también su umbral inferior. Esto significa que es imposible reducir el retardo medio de detección por debajo de $T/6$ al usar una estrategia geométrica, no importa cuantas observaciones se añadan al experimento.

5.6 Comparativa de estrategias periódicas y geométricas

La figura 5.6 compara el retardo medio de detección de las estrategias periódicas y geométricas ante experimentos uniformes (expresiones (5.13) y (5.17)). Como la figura sugiere, la estrategia periódica ofrece un retardo de detección menor que la geométrica, de hecho menor que la mitad a partir de $N > 5$.

La figura también evidencia la asíntota horizontal de la estrategia geométrica, que establece un umbral mínimo al retardo medio de detección que se puede conseguir con esta estrategia.

5.7 Estrategia óptima ante experimentos uniformes

Acabamos de demostrar como una estrategia periódica ofrece un retardo medio de detección inferior al de la estrategia geométrica. ¿Existe alguna estrategia mejor que la periódica para esta tarea? En esta sección demostramos que no,

de hecho, la estrategia uniforme es la estrategia óptima ante experimentos uniformes cuando lo que se busca es el menor retardo medio de detección.

Para simplificar los cálculos necesarios para esta demostración, resulta práctico introducir ahora una representación nueva de las estrategias de observación:

5.7.1 Representación alternativa de estrategias

Cualquier estrategia de N observaciones puede expresarse intuitivamente como la "distancia" que separa cada una de sus observaciones de las de la estrategia periódica de N observaciones correspondiente:

$$t_i = t_i^{(\mathrm{per})} + b_i, \quad \forall i \in 1, \ldots, N,$$

donde $\{b_i\}_N$ es una secuencia finita de números reales.

Por ejemplo, sea $\left\{t^{(per)}\right\}_5 = \{3, 6, 9, 12, 15\}$ *segundos la estrategia periódica de 5 observaciones para un periodo de ocurrencia* $T = 15$ *segundos, una estrategia cualquiera, como por ejemplo* $\{t\}_5 = \{4, 9, 10, 11, 15\}$*, puede expresarse como aquella que está a* $\{1, 3, 1, -1, 0\}$ *de "distancia" de la periódica.*

Nótese que b_i es cero para todo i si y sólo si $\{t_i\}_N$ es una estrategia periódica.

Se puede hacer una consideración similar acerca de los subintervalos de observación:

Teorema 5.7.1. *Los subintervalos* $\{\Delta_i\}_N$ *de cualquier estrategia de observación pueden ser expresados como la suma de los subintervalos de la estrategia periódica de* N *observaciones y la secuencia* $\{a_i\}_N$*, con* $i = 1, \ldots, N$ *y* $a_i \in \mathbb{R}$*, por tanto, por definición*

$$\Delta_i \doteq \Delta_i^{(per)} + a_i, \quad \forall i \in 1, \ldots, N. \tag{5.18}$$

La secuencia $\{a_i\}_N$ *tiene las siguiente propiedades:*

1.

$$\sum_{i=1}^{N} a_i = 0 \tag{5.19}$$

2.

$$\sum_{i=1}^{N} a_i^2 \geq 0, \tag{5.20}$$

donde la igualdad se cumple si y sólo si $\{t_i\}_N$ *es periódica.*

Demostración. Para la demostración de (5.19) usaremos el lema 5.2.4,

$$\sum_{i=1}^{N} \Delta_i = T,$$

y sustituiremos la definición (5.18):

$$\sum_{i=1}^{N} \Delta_i^{(\mathrm{per})} + \sum_{i=1}^{N} a_i = T.$$

Por tanto,

$$T + \sum_{i=1}^{N} a_i = T,$$

y entonces

$$\sum_{i=1}^{N} a_i = 0.$$

La propiedad (5.20) es trivial, ya que la suma de los cuadrados de números reales es siempre positiva o cero.

La propiedad (5.20) se convierte en una igualdad si y sólo si $a_i = 0$ para todo i, por tanto, si y sólo si Δ_i fueran los subintervalos de una estrategia periódica. □

Teorema 5.7.2. *Una estrategia de detección periódica ofrece el retardo mínimo de detección ante experimentos uniformes.*

Demostración. Sustituyendo la definición (5.18) en (5.9), tenemos que

$$\begin{aligned}
\bar{r}_{\mathrm{uni}} &= \frac{1}{2T} \sum_{i=1}^{N} \left(\Delta_i^{(\mathrm{per})} + a_i \right)^2 = \frac{1}{2T} \sum_{i=1}^{N} \left(\frac{T}{N} + a_i \right)^2 \\
&= \frac{1}{2T} \sum_{i=1}^{N} \left(\frac{T^2}{N^2} + 2a_i \frac{T}{N} + a_i^2 \right) \\
&= \frac{T}{2N} + \frac{1}{N} \sum_{i=1}^{N} a_i + \frac{1}{2T} \sum_{i=1}^{N} a_i^2.
\end{aligned}$$

De acuerdo con (5.13), el primer término de la suma es $\bar{r}_{\mathrm{uni}}^{(\mathrm{per})}$ y de acuerdo a la propiedad (5.19) el segundo término es cero. Por tanto,

$$\bar{r}_{\mathrm{uni}} = \bar{r}_{\mathrm{uni}}^{(\mathrm{per})} + \frac{1}{2T} \sum_{i=1}^{N} a_i^2.$$

Como la propiedad (5.20) es una igualdad si y sólo si la estrategia es periódica, el último término de la suma es mayor que cero para estrategias no periódicas, por lo que,

$$\bar{r}_{\mathrm{uni}}^{(\text{non-per})} > \bar{r}_{\mathrm{uni}}^{(\mathrm{per})}.$$

□

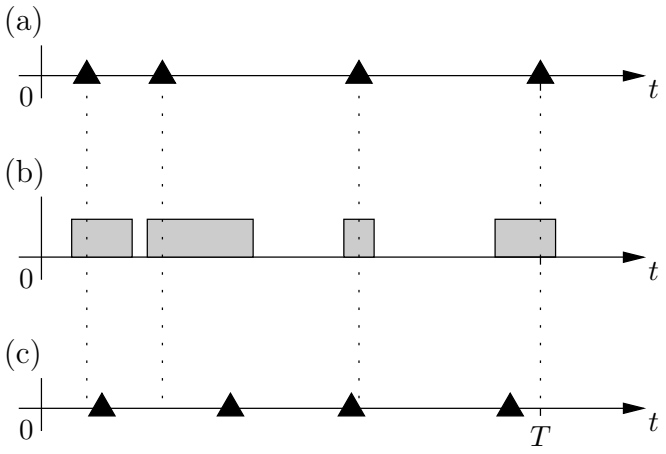

Figura 5.7: (a) Ejemplo de estrategia de 4 observaciones. (b) En gris, los periodos de tiempo en que podrían acontecer las observaciones si le aplicamos unas variaciones aleatorias determinadas a la estrategia de la subfigura a. (c) Una realización concreta de la estrategia con variaciones de la subfigura b.

5.8 Variación aleatoria de observaciones

Como ya explicamos en §4.5.4, una manera de evitar los problemas de sincronismo de SCTP-AP es introducir una variación aleatoria al momento en que se envían las sondas activas.

La figura 5.7 ilustra el resultado de aplicar variaciones a una estrategia.

Desde el punto de vista del análisis matemático, estas variaciones aleatorias convierten las estrategias de observación, de sucesiones de números constantes, en sucesiones de variables aleatorias. Lo mismo pasa con los subintervalos de observación.

El retardo de detección, que en el caso de una estrategia sin variaciones, es una función de una variable aleatoria (el momento de ocurrencia del evento), pasa a convertirse en una función de varias variables aleatorias.

Por ejemplo, una estrategia periódica de 4 observaciones con variaciones, tendrá un retardo de detección que es función de 5 variables aleatorias: el momento de ocurrencia del evento t y el momento en que se realizan cada una de las 4 observaciones $\{v_1, v_2, v_3, v_4\}$.

Intuitivamente, cuanto mayor sea la varianza de las variaciones aplicadas, menos probable es que nos encontremos con un caso de sincronismo. Sin embargo, es importante dejar claro desde el principio que cuanto mayor sea esta varianza más nos estaremos alejando del caso ideal de la estrategia periódica pura, es decir, el retardo medio de detección será mayor.

Esto quiere decir que nos encontramos ante un compromiso, es necesario aplicar variaciones sobre la estrategia periódica para evitar problemas de sincronismo pero cuanto mayores sean estas variaciones mayor será el retardo medio

de detección.

El resto de esta sección está dedicada a estudiar los efectos de aplicar variaciones a una estrategia periódica, en el contexto de protocolos de comunicaciones que usan sondas activas para la detección de eventos con probabilidad uniforme.

Nuestro propósito es contribuir a la resolución racional del compromiso entre reducción del riesgo de sincronismo y aumento del retardo medio de detección.

5.8.1 Particularidades en protocolos de comunicaciones

Aunque en principio las variaciones pueden tomar cualquier forma, si nos centramos en utilizarlas para evitar el sincronismo en protocolos de comunicaciones podemos concretar algunas de sus características:

- **Variaciones uniformes**: dado que el objetivo de las variaciones es evitar el sincronismo entre asociaciones, resulta óptimo utilizar variaciones con densidad de probabilidad uniforme.

  Utilizar variaciones con densidad de probabilidad no uniforme estará dando preferencia a que se envíen sondas en unos momentos determinados por lo que la probabilidad de coincidencia entre asociaciones diferentes será más alta que si utilizamos variaciones de densidad de probabilidad uniforme.

- **Las variaciones de cada observación son independientes entre si**: Se podría pensar que basta con alterar un mismo tiempo aleatorio *todas* las observaciones de una estrategia para evitar el problema del sincronismo.

  Sin embargo, la probabilidad de coincidencia entre asociaciones es mayor si se hace así: si dos asociaciones coinciden en el momento de enviar la primera de sus sondas, el riesgo de que también coincidan en el resto de sus sondas es mayor que si aplicamos variaciones independientes al envío de cada sonda.

  Por esta razón preferimos utilizar variaciones independientes entre si para cada sonda. La probabilidad de coincidencia de cada sonda en particular entre asociaciones sigue siendo la misma, pero la probabilidad de coincidencia de *varias* sondas es sensiblemente menor.

  Matemáticamente, esto se traduce en que las variables aleatorias que gobiernan la variación a aplicar en cada sonda son independientes entre si. Es decir $\{\mathrm{v}_1, \cdots, \mathrm{v}_N\}$ son independientes entre si y desde luego también lo son del momento de ocurrencia del evento t.

- **¿Variaciones con la misma distribución para cada observación?** Aunque en el contexto que nos ocupa preferimos utilizar variaciones con distribución uniforme para todas las observaciones, cabe la posibilidad de utilizar distribuciones con diferente media y varianza para cada observación o aplicar variaciones con la misma distribución para todas.

  Hemos decidido estudiar variaciones con la misma distribución en todas las observaciones, por sencillez.

- **Magnitud de la variación**. Aunque a lo largo de nuestro análisis de variaciones la magnitud de la variación β va a ser uno de los parámetros a estudiar, vamos a limitarla, por sencillez, a la separación entre las observaciones, de forma que ninguna observación pueda programarse después de su siguiente por causa de la variación introducida.

  De acuerdo con esto, la variación máxima que vamos a utilizar es

  $$\beta_{\max} = \frac{T}{N},$$

  que es el tamaño de los subintervalos de la estrategia periódica.

- **La estrategia resultante tras la variación debe ser fiable**: El resultado de aplicar variaciones a los momentos en que se realizan las observaciones *debe* dar lugar a una estrategia fiable, es decir con su última observación en un tiempo T o posterior.

  De otro modo, si la última observación ocurre antes de T por efecto de la variación, las ocurrencias del evento posteriores a ella pasarían inadvertidas. La figura 5.7c muestra un ejemplo de estrategia que acaba siendo no fiable por culpa del tipo de variación aplicada a la última sonda. Es recomendable evitar que esto suceda.

  Por tanto debemos asegurarnos de aplicar variaciones a la última observación que sólo puedan retrasarla, evitando el tipo de variaciones que puedan adelantarla.

- **Detección de eventos ocurridos posteriormente a** T: En un contexto de protocolos de comunicaciones, la probabilidad de que el evento de recuperación del fallo de ruta ocurra posteriormente a T no es nula.

  Según lo explicado en el punto anterior, enviaremos la última observación pasado T, por lo que existe una probabilidad de detectar ocurrencias del evento posteriores a T. Desgraciadamente, si la última observación detecta el evento, es imposible saber si este ocurrió antes o después de T, por lo que tendremos que asumir este riesgo.

  La alternativa desemboca en estrategias no fiables y no nos interesa ya que preferimos sufrir falsos positivos que falsos negativos. En contextos diferentes al de nuestra tesis podría ser conveniente invertir está preferencia.

  Por tanto, la probabilidad de detección de falsos positivos PFP de eventos uniformes, en estas circunstancias, es de

  $$PFP = \frac{\beta}{T + \beta},$$

  que para estrategias periódicas está acotada por $\beta_{\max}$ al valor de

  $$PFP_{\max} = \frac{1}{1 + N}.$$

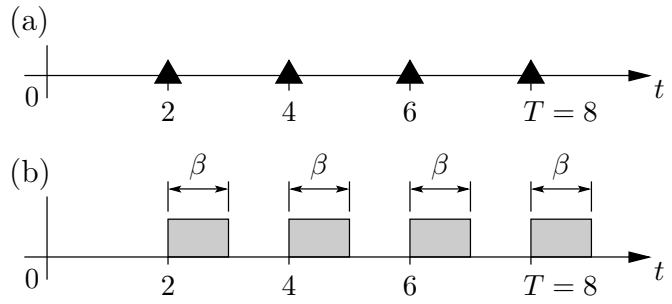

Figura 5.8: Ejemplo del tipo de variaciones que vamos a estudiar en el contexto de SCTP-AP. (a) Estrategia periódica original, de 4 observaciones para un periodo de ocurrencia $T = 8$ segundos. (b) En gris, periodos en que puede acontecer cada observación en la estrategia con variaciones correspondiente. El retraso máximo sufrido por cada observación es de β.

Resumen

La figura 5.8 resume nuestra particularización del estudio de variaciones para la detección de la recuperación del fallo de ruta en SCTP-AP.

Utilizaremos variaciones con densidad de probabilidad uniforme, independientes entre si y que dan lugar a estrategias fiables, aunque eso implique detectar algunos eventos ocurridos después de T.

Partimos de una estrategia periódica pura y aplicamos las variaciones, retrasando cada una de las observaciones un valor aleatorio de distribución uniforme entre 0 y β.

5.8.2 Estrategias con variaciones

Definición 5.8.1. *La* **función rectangular Π** $: \mathbb{R} \to \mathbb{R}$*, definida como*

$$\Pi(x) \doteq \begin{cases} 0, & si\ x < 0 \\ 1, & si\ 0 < x < 1 \\ 0, & resto, \end{cases} \tag{5.21}$$

tiene la forma que mostramos en la figura 5.9a.

Por lo general, no es necesario especificar los valores que toma $\Pi(x)$ en sus discontinuidades (esto es, en $x = 0$ y $x = 1$), ya que no afecta a los cálculos ([86]). En la figura hemos optado por representarlos igualmente, por completitud, utilizando el valor 1/2, por ser el más habitual en la bibliografía.

Es conveniente definir también una versión generalizada de la Π, resultado de ensancharla por un valor β:

Definición 5.8.2. *Sea el número real $\beta \neq 0$, la* **función rectangular gene-**

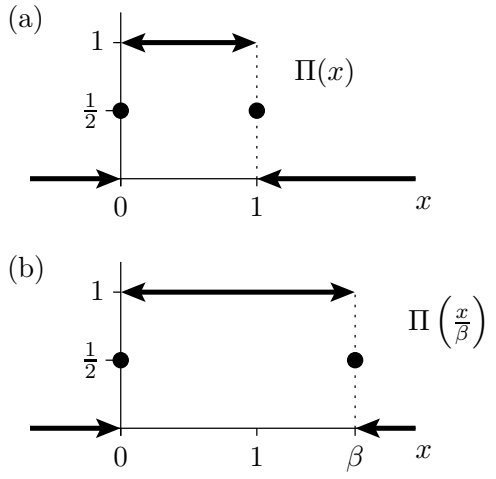

Figura 5.9: (a) La función rectangular $\Pi(x)$. (b) La función rectangular generalizada $\Pi(\frac{x}{\beta})$.

ralizada $\mathbf{\Pi\left(\frac{x}{\beta}\right)} : \mathbb{R} \to \mathbb{R}$, *se puede expresar como*

$$\Pi\left(\frac{x}{\beta}\right) \doteq \begin{cases} 0, & si\ x < 0 \\ 1, & si\ 0 < x < \beta \\ 0, & resto, \end{cases} \tag{5.22}$$

y tiene la forma que mostramos en la figura 5.9b.

Estrategias como vectores aleatorios

Una estrategia de N observaciones $\{t_i\}_N$, a la que se le aplica una variación β, es la secuencia aleatoria $\{\mathrm{t_i}\}_N$ formada por las N variables aleatorias correspondientes. En forma analítica,

$$\{\mathrm{t_i}\}_N \doteq \{t_i\}_N + \left\{ \mathrm{v}_1, \ldots, \mathrm{v_N} \mid f_{\mathrm{v_i}}(v_i) = \frac{1}{\beta}\,\Pi\!\left(\frac{v_i}{\beta}\right) \right\}, \tag{5.23}$$

donde $f_{\mathrm{v_i}}(v_i)$ es la función de densidad de probabilidad de $\mathrm{v_i}$.

Por ejemplo, sea la estrategia periódica $\{t^{\mathrm{per}}\}_4 = \{2, 4, 6, 8\}$ (figura 5.8a). Si aplicamos una variabilidad β de 1 segundo sobre la estrategia periódica original, tendremos que la primera observación se realizará en algún momento al azar entre el segundo 2 y el 3, en lugar de realizarse en el segundo 2. La segunda observación, en vez de realizarse en el segundo 4, se realizará en algún momento entre los segundos 4 y 5, y así sucesivamente. La figura 5.8b ilustra este efecto.

5.8.3 Retardo medio de detección

Según la definición 5.2.9, el retardo detección r de una estrategia sin variaciones $\{t_i\}_N$, es una función de la variable aleatoria que representa el momento de ocurrencia del evento t, es decir,

$$\mathrm{r} = r(\mathrm{t}), \tag{5.24}$$

donde $r(t)$ corresponde a la ecuación (5.6), que reproducimos a continuación por conveniencia:

$$r(t) \doteq \sum_{i=1}^{N} \Delta_i \, \Gamma\left(\frac{t-(t_i-\Delta_i)}{\Delta_i}\right).$$

Análogamente, el retardo de detección correspondiente a una estrategia con variaciones $\{\mathrm{t_i}\}_N$, es función de las variables aleatorias t, $\mathrm{v}_1, \ldots \mathrm{v_N}$:

$$\mathrm{r}_\beta = r\left(\mathrm{t}, \mathrm{v}_1, \ldots, \mathrm{v_N}\right). \tag{5.25}$$

De igual manera que en §5.3, podemos calcular el retardo medio de detección de estrategias sin variación mediante

$$\bar{r} = \int_{-\infty}^{\infty} r(t) f_{\mathrm{t}}(t)\, dt,$$

que para estrategias con variación se convierte en

$$\bar{r}_\beta = \int \cdots \int_{-\infty}^{\infty} r(t, v_1, \ldots, v_N) f_{\mathrm{t},\mathrm{v}_1 \ldots \mathrm{v_n}}(t, v_1, \ldots, v_N)\, dt\, dv_1 \cdots dv_N.$$

Teniendo en cuenta que las variables aleatorias t, $\mathrm{v}_1, \ldots \mathrm{v_N}$ son independientes entre si,

$$\bar{r}_\beta = \int \cdots \int_{-\infty}^{\infty} r(t, v_1, \ldots, v_N)\, f_{\mathrm{t}}(t) \prod_{j=1}^{N} f_{\mathrm{v_i}}(v_i)\, dt\, dv_1 \cdots dv_N. \tag{5.26}$$

En el caso que nos ocupa, tanto t, como todas las $\mathrm{v_i}$, son variables aleatorias de distribución uniforme, por lo tanto

$$\bar{r}_\beta = \int \cdots \int_{-\infty}^{\infty} r(t, v_1, \ldots, v_N)\, \frac{1}{T}\Pi\left(\frac{t}{T}\right) \prod_{j=1}^{N} \frac{1}{\beta}\Pi\left(\frac{v_i}{\beta}\right)\, dt\, dv_1 \cdots dv_N,$$

$$\bar{r}_\beta = \frac{1}{T\beta^N} \int_0^T \int_0^\beta \cdots \int r(t, v_1, \ldots, v_N)\, dt\, dv_1 \cdots dv_N. \tag{5.27}$$

Aplicando el teorema de Fubini[4] podemos expresar la integral en $N+1$ dimensiones como $N+1$ integrales iteradas, de forma que

$$\bar{r}_\beta = \frac{1}{\beta} \int_0^\beta \left(\frac{1}{\beta} \int_0^\beta \left(\cdots \frac{1}{T} \int_0^T r(t, v_1, \ldots, v_N)\, dt \cdots \right) dv_{N-1} \right) dv_N. \tag{5.28}$$

[4]La función $r(t, v_1, \ldots, v_N)$ es medible en el espacio $T \times V_1 \times \cdots \times V_N$, que es, además un espacio medible.

Vamos a denominar a estar integrales iteradas mediante la notación $I_{i,N}$, para $i = 1, \ldots, N+1$, donde N indica el número de observaciones de la estrategia e i la integral correspondiente en la igualdad anterior de dentro a fuera, de tal forma que

$$I_{i,N} \doteq \begin{cases} \frac{1}{T}\int_0^T r(t, v_1, \ldots, v_N)\, dt, & \text{si } i = 1 \\ \\ \frac{1}{\beta}\int_0^\beta I_{i-1}\, dv_{i-1}, & \text{resto.} \end{cases}$$

Combinando esta notación con (5.28) tenemos que

$$\bar{r}_\beta = I_{N+1,N}. \tag{5.29}$$

Por ejemplo, para una estrategia periódica de 3 observaciones con una variación de magnitud β, tenemos que

$$\begin{aligned} \bar{r}_\beta = I_{4,3} &= \frac{1}{\beta}\int_0^\beta I_{3,3}\, dv_3, \\ I_{3,3} &= \frac{1}{\beta}\int_0^\beta I_{2,3}\, dv_2, \\ I_{2,3} &= \frac{1}{\beta}\int_0^\beta I_{1,3}\, dv_1, \\ I_{1,3} &= \frac{1}{T}\int_0^T r(t, v_1, v_2, v_3)\, dt. \end{aligned}$$

Caso particular: estrategias de una única observación

Particularizando (5.29) para el caso de $N = 1$, tenemos que

$$\bar{r}_\beta = I_{2,1} \tag{5.30}$$

$$I_{2,1} = \frac{1}{\beta}\int_0^\beta I_{1,1}\, dv_1, \tag{5.31}$$

$$I_{1,1} = \frac{1}{T}\int_0^T r(t, v_1)\, dt. \tag{5.32}$$

La figura 5.10 muestra el retardo de detección de una estrategia con variaciones de una sola observación $r(t, v_1)$.

Según (5.32) el cálculo de $I_{1,1}$ es proporcional al área bajo $r(t, v_1)$ entre 0 y T. Ahora bien, la figura 5.10 ilustra que este área puede expresarse como la resta del área de las regiones $A \cup Z$ menos la de Z, que son ambas regiones con forma de triángulos isósceles rectángulos, por tanto

$$I_{1,1} = \frac{1}{T}\int_0^T r(t, v_1)\, dt = \frac{1}{T}\left(\frac{(T + v_1)^2}{2} - \frac{v_1^2}{2}\right),$$

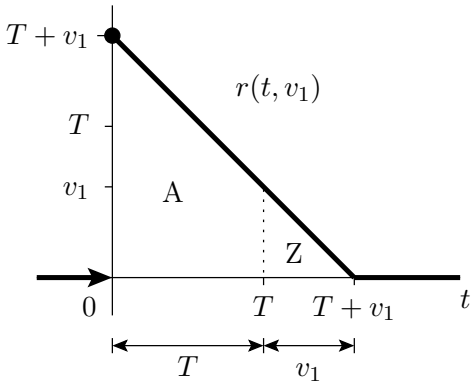

Figura 5.10: Retardo de detección $r(t, v_1)$ de una estrategia de una única observación con una variación v_1. Hemos dividido el área bajo $r(t, v_1)$ en dos regiones mediante una línea de puntos, la región A (de $t = 0$ hasta T) y la región Z (de $t = T$ hasta $T + v_1$).

$$I_{1,1} = \frac{1}{2}\left(T + 2v_1\right). \tag{5.33}$$

El retardo medio de detección $\bar{r}_\beta$ (ecuaciones (5.30) y (5.31)) resulta ahora sencillo de calcular:

$$\begin{aligned}\bar{r}_\beta = I_{2,1} &= \frac{1}{\beta}\int_0^\beta I_1\, dv_1 = \frac{1}{\beta}\int_0^\beta \frac{1}{2}\left(T + 2v_1\right)\, dv_1 \\ &= \frac{1}{2\beta}\left(T\beta + \beta^2\right) = \frac{1}{2}\left(T + \beta\right).\end{aligned}$$

Resumiendo, el retardo medio de detección de un evento uniforme de periodo T usando una estrategia de una sola observación con una variación de magnitud β es

$$\bar{r}_\beta = \frac{1}{2}\left(T + \beta\right). \tag{5.34}$$

Caso particular: estrategias de dos observaciones

Particularizando (5.29) para el caso de $N = 2$, tenemos que

$$\bar{r}_\beta = I_{3,2} \tag{5.35}$$

$$I_{3,2} = \frac{1}{\beta}\int_0^\beta I_{2,2}\, dv_2, \tag{5.36}$$

$$I_{2,2} = \frac{1}{\beta}\int_0^\beta I_{1,2}\, dv_1, \tag{5.37}$$

$$I_{1,2} = \frac{1}{T}\int_0^T r(t, v_1, v_2)\, dt. \tag{5.38}$$

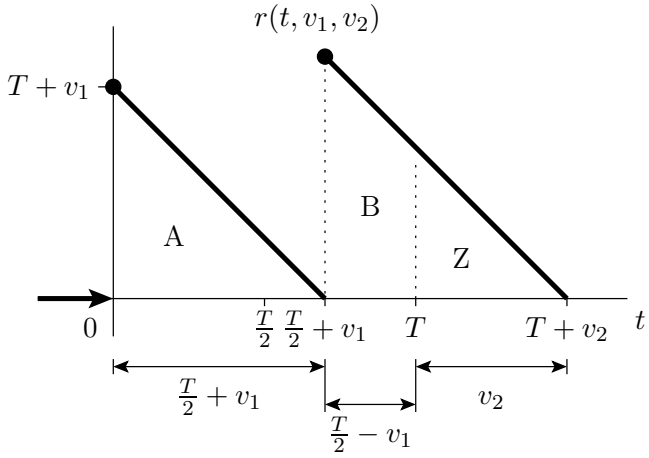

Figura 5.11: Retardo de detección $r(t, v_1, v_2)$ de una estrategia de dos observaciones con variaciones $\{v_1, v_2\}$. Hemos dividido el área bajo $r(t, v_1, v_2)$ en tres regiones mediante líneas de puntos: la región A (de $t = 0$ hasta $T/2 + v_1$), la región B (de $t = \frac{T}{2} + v_1$ hasta T) y la región Z (de $t = T$ hasta $T + v_2$).

La figura 5.11 muestra el retardo de detección de una estrategia con variaciones de dos observaciones $r(t, v_1, v_2)$.

Según (5.38) el cálculo de $I_{1,2}$ es proporcional al área bajo $r(t, v_1, v_2)$ entre 0 y T. Siguiendo el mismo razonamiento que en el caso de estrategias de una sola observación, el área bajo $r(t, v_1, v_2)$ entre $t = 0$ y T puede expresarse como el área de la región A mas la de la región $B \cup Z$ menos la de la región Z, que son todas regiones con forma de triángulos isósceles rectángulos, por tanto

$$I_{1,2} = \frac{1}{T}\int_0^T r(t, v_1, v_2)\, dt = \frac{1}{T}\left(\frac{(\frac{T}{2} + v_1)^2}{2} + \frac{(\frac{T}{2} + v_2 - v_1)^2}{2} - \frac{v_2^2}{2}\right),$$

$$I_{1,2} = \frac{1}{2}\left(\frac{T}{2} + v_2 + \frac{2v_1^2}{T} - \frac{2v_1 v_2}{T}\right). \tag{5.39}$$

El cálculo de $I_{2,2}$, según (5.37), es el siguiente:

$$\begin{aligned} I_{2,2} &= \frac{1}{\beta}\int_0^\beta I_{1,2}\, dv_1 = \frac{1}{\beta}\int_0^\beta \frac{1}{2}\left(\frac{T}{2} + v_2 + \frac{2v_1^2}{T} - \frac{2v_1 v_2}{T}\right) dv_1 \\ &= \frac{1}{2}\left(\frac{T}{2} + v_2 + \frac{2}{3}\frac{\beta^2}{T} - \frac{\beta v_2}{T}\right). \end{aligned}$$

El retardo medio de detección $\bar{r}_\beta$, según (5.35) y (5.36), resulta ahora sencillo

de calcular:

$$\bar{r}_\beta = I_{3,2} = \frac{1}{\beta}\int_0^\beta I_{2,2}\,dv_2 = \frac{1}{\beta}\int_0^\beta \frac{1}{2}\left(\frac{T}{2} + v_2 + \frac{2}{3}\frac{\beta^2}{T} - \frac{\beta v_2}{T}\right) dv_2$$
$$= \frac{1}{2}\left(\frac{T}{2} + \frac{\beta}{2} + \frac{2}{3}\frac{\beta^2}{T} - \frac{\beta^2}{2T}\right) = \frac{1}{2}\left(\frac{T}{2} + \frac{\beta}{2} + \frac{\beta^2}{6T}\right).$$

Resumiendo, el retardo medio de detección de un evento uniforme de periodo T usando una estrategia periódica de dos observaciones con una variación de magnitud β es

$$\bar{r}_\beta = \frac{1}{2}\left(\frac{T}{2} + \frac{\beta}{2} + \frac{\beta^2}{6T}\right). \tag{5.40}$$

Caso general: estrategias de N observaciones

Si se aborda el cálculo del retardo medio de detección para estrategias de más de dos observaciones, se aprecia un patrón interesante: El cálculo de $I_{1,N}$ implica la suma de las áreas de:

- El triángulo correspondiente a la primera observación, de base $T/N + v_1$ (el ancho del triángulo de la estrategia periódica pura ensanchado por la primera variación.).
- El resto de triángulos correspondientes al resto de observaciones, de bases $T/N + v_i - v_{i-1}$ (el ancho del triángulo de la estrategia periódica pura, ensanchado por la variación correspondiente y reducido por la anterior).
- El triángulo correspondiente a la última variación (identificado con la letra Z en las figuras 5.10 y 5.11) que debe ser restado y que siempre tiene una base de longitud v_n.

Utilizando estas características para ordenar adecuadamente los términos del desarrollo de $I_{1,N}$ llegamos a la siguiente expresión:

$$I_{1,N} = \frac{1}{2}\left(\frac{T}{N} + 2\frac{v_N}{N} + \frac{2}{T}\sum_{i=1}^{N-1} v_i^2 - \frac{2}{T}\sum_{i=1}^{N-1} v_i v_{i+1}\right). \tag{5.41}$$

El cálculo del $I_{2,N}, I_{3,N}, \ldots, I_{N+1,N}$ también presenta unos patrones claros. Si integramos N veces la expresión anterior, cada una de ellas en $v_1, \ldots, v_n$, obtenemos el siguiente resultado:

$$\bar{r}_\beta = I_{N+1,N} = \frac{1}{2}\left(\frac{T}{N} + \frac{\beta}{N} + (N-1)\frac{2\beta^2}{3T} - (N-1)\frac{\beta^2}{2T}\right),$$

que simplificando queda:

$$\bar{r}_\beta = \frac{1}{2}\left(\frac{T}{N} + \frac{\beta}{N} + (N-1)\frac{\beta^2}{6T}\right). \tag{5.42}$$

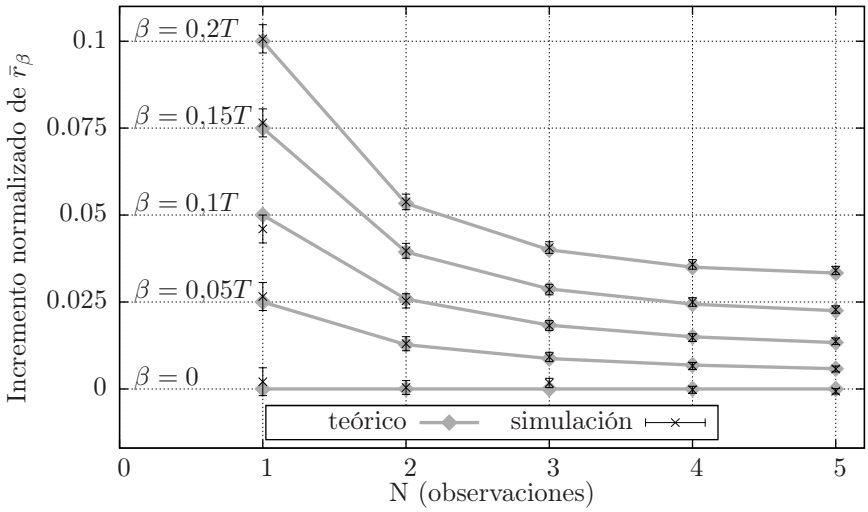

Figura 5.12: Incremento del retardo medio de detección de estrategias periódicas con variaciones. La figura muestra el incremento para diferentes magnitudes de la variación (β) y número de observaciones (N). Los intervalos de confianza de los resultados de las simulaciones se han calculado para un nivel de confianza del 95 %.

La figura 5.12 muestra el incremento en el retardo medio de detección al aplicar variaciones de diferentes magnitudes a una estrategia periódica.

La figura incluye también resultados obtenidos mediante simulación con la intención de validar el análisis matemático de estas últimas secciones. Las simulaciones se han realizado de la siguiente manera:

Se generan las variaciones de las observaciones de la estrategia periódica mediante N valores extraídos de un generador de números aleatorios entre 0 y β. Se genera también el momento de la ocurrencia del evento mediante otro valor aleatorio entre 0 y T. Se averigua cual es la primera observación que detecta el evento y se resta su tiempo de ocurrencia al del evento. Se repite este mismo experimento un número elevado de veces y se estima el retardo medio de detección mediante la media muestral de todos los experimentos.

La ecuación (5.42) depende linealmente de la relación entre T y β, por lo que los datos de la figura pueden presentarse normalizados a T sin perder exactitud.

5.9 Representación de estrategias en un plano

La figura 5.13 representa el plano $(\bar{r}, N)$ (retardo medio de detección - número de observaciones) de todas las posibles estrategias de observación de experimentos uniformes.

Las mejores estrategias se sitúan abajo a la izquierda, donde con un número reducido de observaciones se obtiene un bajo retardo medio de detección.

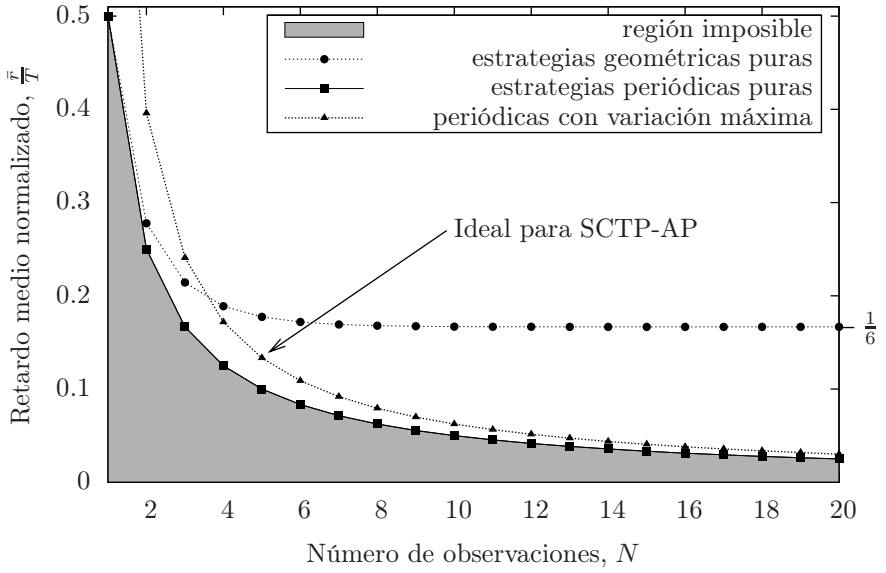

Figura 5.13: Representación en un plano de las estrategias de observación para experimentos uniformes, según su retardo medio de detección y su número de observaciones.

La zona sombreada representa la región imposible de alcanzar según el teorema 5.7.2: no existe ninguna estrategia que se sitúe en esta zona del plano.

En la figura se representan dos familias particularmente importantes de estrategias de observación:

- la estrategia periódica, que además es la estrategia óptima (está en el límite de la zona imposible)
- y la estrategia geométrica, que además de ser la utilizada en SCTP, ilustra el tipo de estrategias que tienen un umbral mínimo de retardo medio de detección, es decir una asíntota horizontal, que en el caso concreto de la geométrica es de $T/6$.

Cualquier otra estrategia será un punto situado en la zona no sombreada de la figura. Por ejemplo, en la figura también se representan las estrategias periódicas con la variación máxima posible ($\beta_{\max} = T/N$).

Las estrategias periódicas con variaciones menores a la máxima ($\beta < \beta_{\max}$) están localizadas en la región comprendida entre la periódica pura y la periódica con la máxima variación.

Un examen visual de la figura sugiere que estrategias periódicas de 5 observaciones con variación, presentan una mejora razonable con respecto a las estrategias geométricas utilizadas en SCTP. Incluso si utilizamos el máximo valor posible para la variación, obtendremos un retardo de detección $0{,}25T$ menor que en el caso de SCTP, por lo que resultan ideales para SCTP-AP.

5.10 Conclusiones

Los fundamentos matemáticos y la notación introducidas en §5.2 y §5.3 nos han permitido estudiar las propiedades de las estrategias de observación más relevantes para el problema de la detección de fallo de ruta en SCTP. Pensamos que también pueden ser útiles en una aproximación más general al problema del la detección de sucesos mediante sondas.

En §5.4 y §5.5, proporcionamos expresiones analíticas para el retardo de detección de las estrategias periódicas y geométricas ante experimentos uniformes.

En §5.7 demostramos como la estrategia periódica es la estrategia óptima en cuanto al retardo medio de detección de experimentos uniformes.

En §5.8 analizamos el incremento en el retardo medio de detección que se produce al aplicar variaciones a las estrategias periódicas, dentro de un contexto razonable para la detección de recuperación de fallos de ruta en protocolos de comunicaciones.

En §5.9 ilustramos de manera gráfica las propiedades principales de las estrategias de observación, lo que facilita comparar estrategias entre si a partir de su representación en un espacio de dos dimensiones.

El objetivo original de este capítulo era justificar la elección de una estrategia periódica con variaciones para la planificación del sondeo activo en SCTP-AP, pero pensamos que las contribuciones que contiene superan esta finalidad y proporcionan una base matemática para el estudio de estrategias de sondeo en general.

Parte III

Contribuciones a nivel de sistema operativo

Capítulo 6

Criterios de selección de interfaces de red

6.1 Introducción

En este capítulo presentamos una lista, pensamos que completa, de los criterios a tener en cuenta a la hora de elegir por qué interfaz encaminar las comunicaciones.

El objetivo de esta lista es servir como referencia a la hora de implementar servicios en el sistema operativo que permitan gestionar eficientemente la diversidad de interfaces disponibles.

A pesar de la abundancia de dispositivos de usuario multiconectados, hoy en día sigue siendo difícil realizar operaciones simples con ellos que aprovechen sus capacidades. Por ejemplo, transferir los flujos de datos a la interfaz más barata o a la que ofrezca una calidad de servicio suficiente. Los sistemas operativos no sólo no ayudan sino que, a menudo, presentan trabas al usuario para llevar a cabo este tipo de operaciones.

Para ilustrar algunos de los problemas asociados a la elección de la interfaz más apropiada podemos utilizar un ejemplo de la vida cotidiana:

LLegamos a casa del trabajo y nuestro dispositivo móvil personal intenta encender las luces de la entrada de la casa. El dispositivo está todavía utilizando la interfaz GPRS por la que ha estado conectado a Internet durante nuestro viaje hasta casa. Sin embargo, sería preferible utilizar la interfaz Wi-Fi para conectarse a nuestra red Wi-Fi doméstica: de esta forma evitaríamos el cortafuegos de la puerta de enlace residencial de nuestra casa, y probablemente sería también energéticamente más eficiente y más barato.

Desde luego, se puede pensar en soluciones sencillas y directas a este problema; en el escenario propuesto en el ejemplo, el dispositivo móvil de usuario podría conectarse a la red Wi-Fi de nuestras casa siempre que estuviera disponible. De hecho, algunos dispositivos modernos utilizan esta estratégia. Sin embargo, este tipo de soluciones tan concretas pueden no ser suficientes en escenarios de movilidad más interesantes:

Según nos acercamos a la entrada de nuestra casa, el dispositivo móvil detecta nuestra red Wi-Fi casera y se conecta a ella. El dispositivo enciende las luces de la entrada sin problemas, pero la actualización del firmware que el dispositivo móvil estaba llevando a cabo en ese momento se verá retrasada, ya que nuestra conexión ADSL casera está siendo saturada por una aplicación P2P de intercambio de ficheros que hemos dejado ejecutando esta mañana en uno de nuestros ordenadores caseros. Además de eso, la descarga del próximo plan de viaje que nuestro dispositivo móvil estaba realizando a nuestro coche por Bluetooth se ve cancelada, ya que nuestro coche no dispone de conexión a Internet y sólo es alcanzable desde una interfaz Bluetooth.

Los escenarios de movilidad genéricos en entornos ubicuos requieren de estrategias de selección de interfaces basadas en criterios diversos, teniendo en cuenta además que el usuario no debe ser importunado a menudo a la hora de tomar estas decisiones ([87]).

Los sistemas operativos han soportado desde hace décadas los dispositivos multiconectados, y disponen de abstracciones cómodas y convenientes para aislar a las aplicaciones del número y naturaleza de las interfaces disonibles. Sin embargo, a estas infraestructuras tradicionales les falta dinamismo, conocimiento del contexto y las capacidades de autogestión propias de un entorno ubícuo: la responsabilidad de seleccionar qué interfaz es la más apropiada para nuestras comunicaciones recae en el usuario en el mejor de los casos, y para ello es necesario cierto conocimiento avanzado del sistema operativo.

Incluso para los usuarios más avanzados, una asignación más inteligente de la plétora de interfaces disponibles hoy en día es prácticamente imposible. ¿Qué interfaz elegir para cada uno de nuestros flujos de datos, dados una localización, entorno y estado de la batería del dispositivo? La respuesta habitual de los sistemas operativos dominantes es demasiado estática y restrictiva y fuerza al middleware de movilidad y las aplicaciones a pelear contra el sistema para intentar sacarle partido a la multiconectividad.

La decisión final tampoco debe recaer en el usuario. En entornos ubícuos no es viable cargar al usuario con la responsabilidad y la atención requerida para estas decisiones: es necesaria una automatización de todo el proceso de selección de interfaces. Esto significa disponer de una lista completa y descriptiva de los posibles criterios de selección de interfaces y algún API del sistema operativo que los exponga para que puedan ser tratados programaticamente.

A continuación presentamos la lista de los criterios de selección de interfaces que se deberían tener en cuenta para una solución completa de movilidad en entornos ubícuos, así como una discusión de las posibles dificultades que pueden presentar cada uno a la hora de incorporarlos a un nuevo API del sistema operativo.

6.2 Lista de criterios

En un dispositivo multiconectado, más de una interfaz puede ser capaz de enviar los datos a un determinado destino IP. La elección de la interfaz más apropiada

en estos casos es un problema conocido desde hace tiempo ([88, 89]).

Actualmente, el sistema operativo escoge una de estas interfaces atendiendo a criterios sencillos (§2.3.3), ignorando algunos factores que son importantes en un escenario ubicuo genérico y que presentamos a continuación[1].

Algunos de estos criterios aparecen reiterativamente en la bibliografía sobre dispositivos multiconectados y site multihoming ([91, 5, 92, 8, 93, 94, 95, 96, 97, 98]), otros son menos frecuentes y hemos citado sus fuentes en su propia descripción.

- **Información del nivel de enlace y físico**: como la relación señal a ruido, el máximo ancho de banda disponible, el nivel de interferencias o el consumo de batería.

  Por ejemplo, sería interesante descartar aquellas interfaces incapaces de satisfacer los requerimientos mínimos de calidad de servicio de la aplicación, aunque eso no asegure su cumplimiento por restricciones en enlaces posteriores. También sería interesante descartar las interfaces con mayor consumo de potencia cuando el dispositivo opera en modo de ahorro de energía.

  Recomendamos un cierto grado de aislamiento de los detalles particulares de cada implementación del nivel de enlace y físicos; procesar estos datos no debe requerir un conocimiento profundo del funcionamiento interno a bajo nivel de estos protocolos, y debe permitir comparar valores entre tecnologías bien diferentes. La tecnología MIH está haciendo progresos precisamente en este sentido.

  Algunos de estos datos dependerán de cómo la interfaz está siendo utilizada, por ejemplo, el consumo de potencia en tarjetas inalámbricas depende de la relación entre envíos y recepciones o de la presencia de dispositivos cercanos utilizando ese mismo enlace.

  Muchos de estos detalles son difíciles de recopilar, por lo que recomendamos el uso de figuras de mérito y escenarios ficticios estándar a la hora de expresarlos. Por ejemplo: consumo de la interfaz, principalmente a la escucha, en un entorno de bajo tráfico en el enlace (que podría representar la descarga de un streaming de vídeo en un enlace casero de baja ocupación).

- **Infromación adicional del nivel IP**. Otros elementos del protocolo IP pueden incorporarse a la hora de clasificar y decidir qué interfaz utilizar; Por ejemplo, el protocolo de transporte utilizado, ¿es fiable como TCP o estamos usando UDP?, o si está el tráfico clasificado mediante servicios diferenciados o clases de tráfico de IPv6.

  Teniendo en cuenta estos factores se podría encaminar todo el tráfico similar al de las aplicaciones VoIP por cierta interfaz y el resto por otra, por ejemplo.

[1] Hemos publicado una versión de esta misma lista en [90].

- **Información sobre el protocolo de transporte**: Algunos protocolos de transporte disponen de varios modos de funcionamiento o permiten enviar varios flujos en la misma conexión. Por ejemplo en SCTP se podría encaminar el tráfico de sus streams por interfaces diferentes.

  Esto también podría utilizarse para aquellos protocolos de transporte para dispositivos multiconectados que permiten el envío simultáneo de los datos por varias interfaces al mismo tiempo ([10]).

  Algunas métricas extremo a extremo pueden ser tenidas en cuenta eficientemente a este nivel de la torre de protocolos, por ejemplo, el retraso extremo a extremo suele ser un factor importante para algunas aplicaciones ([99, 100]), así como las pérdidas extremo a extremo ([101, 102]).

- **Información sobre el nivel de sesión y aplicación**. Por ejemplo, encaminar todo el tráfico HTTP por en enlace a una red de uso público y todo el tráfico SSH por la red corporativa de nuestra empresa.

  Obtener este tipo de información sobre el nivel de sesión y aplicación no siempre es posible, debido al cifrado y el uso de puertos no estándar. Las aplicaciones podrían ayudar a resolver estos problemas mediante notificaciones específicas sobre sus flujos de datos.

- **Información sobre el proveedor de servicio**: coste, políticas de AAA, filtrado de puertos, cortafuegos, políticas de uso o calidad de servicio contractual. Por ejemplo, sería interesante encaminar las descargas de un podcast por una interfaz barata aunque más lenta, ya que no se va a visualizar el contenido en un tiempo próximo.

  En [8] se describen algunas consideraciones de seguridad sobre como los proveedores de servicio pueden publicar este tipo de información y restricciones de uso, aunque hoy el día, este problema es principalmente semántico, ¿cómo expresar, de forma estándar, todos estos detalles?

- **Seguridad**. Algunos protocolos de nivel físico y de enlace son inherentemente más inseguros que otros, o revelan demasiada información sobre el emisor.

  El uso de técnicas criptográficas, ofuscación de protocolos o esteanografía en los niveles superiores puede no ser suficiente. Por ejemplo la radiación electromagnética que genera una interfaz inalámbrica puede ser suficiente para revelar la presencia del emisor o su localización aproximada.

  Incluso el patrón estadístico de los datos enviados puede ser un riesgo para la seguridad en determinados escenarios. También se pueden hacer consideraciones políticas sobre por que países u organizaciones se van a encaminar los datos.

  En [103] se incluye una breve descripción de este tipo de problemas en escenarios militares.

- La **fiabilidad del enlace** es una consideración importante en muchas situaciones. Por ejemplo: sistemas de aviso de emergencias civiles o para las comunicaciones de las fuerzas del orden.

  Incluso si disponemos de un dispositivo multiconectado, si los traspasos no son imperceptibles, el coste de tener que realizar traspasos frecuentes por estar en redes poco fiables puede ser una buena razón para no usarlas.

- **Redes de distribución de contenidos**. Determinados contenidos sólo pueden accederse desde ciertas redes de acceso y proveedores de servicio. Las aplicaciones que accedan a estos servicios deberían ser conscientes de este hecho y seleccionar las interfaces apropiadas para sus comunicaciones.

- **Calidad de servicio**. Los criterios de calidad de servicio tienen un alto componente extremo a extremo, por lo que puede ser difícil recopilar datos sobre estos aspectos desde un ámbito exclusivamente local. Si los cuellos de botella no se encuentran en la red de acceso, puede no ser evidente relacionar unos criterios de calidad de servicio extremo a extremo con qué interfaz utilizar para disfrutarlos.

- **Localización espacial y temporal** del dispositivo. Todos los sistemas operativos disponen de una interfaz estándar para saber la hora y la fecha, pero pocos disponen de un API para saber donde está el dispositivo, aunque la mayor parte de las veces éste pudiera responder con "localización desconocida".

  Con la reciente incorporación de tecnologías de localización como el GPS, los sistemas operativos están empezando a incorporar un API de localización.

  La localización espacial y temporal puede ayudar a descartar interfaces inalámbricas por ejemplo, en caso de que estemos saliendo de su zona de cobertura. También es útil para predecir los próximos desplazamientos del usuario y realizar conexiones a priori con la intención de realizar traspasos make-before-break siempre que sea posible.

- **Estado del dispositivo**. El estado del dispositivo puede darnos pistas sobre qué interfaces utilizar. Por ejemplo, puede ser interesante saber cuando un dispositivo tiene la batería baja o si el usuario ha activado un modo de ahorro de energía.

- **Preferencias de usuario y aplicaciones**. Las preferencias de usuario y de las aplicaciones se expresan generalmente como políticas de selección, ya que suelen combinar varios de los criterios mencionados hasta el momento.

  En dispositivos multiusuario, flujos de datos con características similares pueden ser encaminados por interfaces diferentes debido a estas políticas de usuario.

  Los administradores del sistema pueden imponer restricciones de uso a las interfaces del dispositivo mediante estas mismas políticas. Recomendamos

que las políticas más abstractas, de más alto nivel, sean extremadamente sencillas de expresar y utilizar, por ejemplo, "Usar la interfaz más barata", debería ser trivial de solicitar al sistema operativo.

La descripción de políticas de usuario y aplicación es un tema complejo en si mismo que está fuera de los objetivos de esta tesis. En [104] y [105] se propusieron dos lenguajes de definición de políticas de encaminamiento que ilustran la complejidad de estos asuntos en dispositivos multiconectados.

- **Preferencias del proveedor de servicio**. Para un proveedor de servicio puede ser interesante solicitar a sus usuarios que encaminen sus comunicaciones por unas interfaces u otras, con el propósito de optimizar el tráfico por sus diferentes redes.

  La consideraciones sobre privacidad y libertad de los usuarios deben ser tenidas en cuenta y pueden plantear problemas de difícil solución.

Resulta bastante complejo extender la funcionalidad de las tablas de rutas actuales para hacerlas conscientes de todos estos criterios, la mayoría de los cuales no son ni siquiera conceptos de nivel 3. Las tablas de rutas simplemente no tienen la capacidad expresiva suficiente como para poder incorporar estos criterios.

6.3 Conclusiones

En este capítulo hemos enumerado los posibles criterios de selección de interfaces que, en nuestra opinión, pueden ser relevantes a la hora de elegir qué interfaz de red utilizar para cada comunicación.

Es una lista útil para aquellos interesados en el diseño e implementación de servicios del sistema operativo para facilitar traspasos.

Algunos de estos criterios resultan especialmente complicados de tener en cuenta y hemos incluido nuestras dudas y reservas en su descripción.

En el siguiente capítulo presentamos nuestra propia implementación de un nuevo servicio del sistema operativo que permite almacenar y exportar datos y figuras de mérito sobre algunos de estos criterios para beneficio de las aplicaciones interesadas.

Capítulo 7

Netqos - Servicio de publicación de criterios de selección de interfaces

El objetivo de *netqos* es proporcionar una API estándar para las aplicaciones y middleware de movilidad, de forma que puedan acceder a criterios de selección de interfaces, y así, mejorar sus decisiones acerca de qué interfaz de red utilizar en cada momento.

Hemos implementado una versión de este nuevo servicio como un sistema de ficheros sintético para el núcleo de Linux. En este capítulo describimos su diseño, características, las experiencias que hemos recopilado al usarlo en escenarios reales y sus limitaciones prácticas[1].

7.1 Breve estado del arte

¿Cómo se las apañan las aplicaciones, hoy en día, para seleccionar la interfaz adecuada, sin el servicio que proponemos? La respuesta es que la mayoría de las aplicaciones no lo hacen, vamos a ver por que.

Para el propósito de esta discusión, podemos distinguir dos maneras de que una aplicación seleccione la interfaz más apropiada:

- **Recopilando ella misma los criterios de selección de interfaz que le interesen**: Ya sea consultando al sistema operativo (estado de la batería, pérdidas por cada interfaz, su ancho de banda máximo...) o mediante herramientas diseñadas específicamente para la medida de otros criterios (ancho de banda disponible por sus flujos, pérdidas sufridas hasta el momento, retraso medio...).

[1] Hemos publicado una descripción de este nuevo servicio del sistema operativo en [90].

Si se añade funcionalidad a la aplicación para consultar al sistema operativo sobre alguno de los criterios de selección de interfaz que éste ofrece, tendremos problemas de portabilidad ya que cada sistema operativo exporta diferentes APIs para consultar estos datos.

Incluso aunque esto no fuera un problema, no todas las interfaces exportan sus datos de la misma manera, por ejemplo, la API para conocer el ancho de banda máximo de una red Ethernet (¿es 100baseTX o 1000baseT?) es diferente a la API para conocer el esquema de modulación en una Wi-Fi (¿estamos conectados al punto de acceso a 54 Mbps, 48, 32, 22...).

Si se añade funcionalidad a la aplicación para interactuar con herramientas específicamente diseñadas para la medida de otros criterios de selección de interfaz, estamos añadiendo una dependencia de herramientas externas que puede presentar los mismos problemas de portabilidad y otros nuevos de integración. La implementación de la funcionalidad de estas herramientas como parte de la propia aplicación es también posible.

Por ejemplo, el reproductor de vídeo *Apple Quicktime*, es capaz de ajustar el pre-buffering de los datos de un stream recibido por la red para que el vídeo no comience a reproducirse hasta que haya una confianza razonable en que, si se mantienen las condiciones actuales de calidad de servicio en la red, el vídeo podrá reproducirse entero sin interrupciones.

Existen herramientas que realizan mediciones de criterios de selección de interfaz utilizando dispositivos distribuidos en diferentes puntos de la red y que definen sus propios protocolos para el intercambio de las medidas obtenidas, como por ejemplo QOSMET ([98]). Estás herramientas suelen presentar una API específica para que otras aplicaciones puedan aprovecharse de los datos que recopilan.

Resultaría práctico que todo este código condicional a la interfaz de red o a las herramientas de medida residiese en el sistema operativo o al menos en una biblioteca de espacio de usuario con un interfaz estándar, en vez de replicarlo en cada aplicación que lo necesite.

- **Delegando en el usuario**: Algunas aplicaciones pueden delegar la elección de la interfaz de red a utilizar en el usuario, evitando tener que implementar ellas mismas la recopilación de criterios de selección de interfaz.

  Algunas aplicaciones bien conocidas utilizan este método, a pesar de que no resulte apropiado en entornos ubicuos, por ejemplo `tcpdump`, `traceroute` o `ping`.

  Sin embargo, las aplicaciones citadas no hacen un uso *corriente* de la red, son aplicaciones de red *especiales*, por así decirlo. A lo que nos referimos es que no encontrará en los menús de configuración de su navegador Web favorito un diálogo para que usted seleccione por que interfaz mandar las peticiones HTTP, por las razones que explicamos a continuación.

Independientemente de como una aplicación seleccione la interfaz a utilizar, le va a resultar prácticamente imposible llevar ese deseo a la práctica: la mayoría

de aplicaciones de red utilizan protocolos como TCP y UDP y normalmente están programadas utilizando la API de sockets, que a su vez recurre a la tabla de rutas para la selección del interfaz de salida, es decir, la aplicación no tiene ningún control sobre que interfaz de red utilizar.

Tanto la API de sockets, como la tabla de rutas *están diseñadas para que la aplicación no pueda elegir la interfaz de red* por la que encaminar sus comunicaciones, esto no es un bug, es una funcionalidad (§2.3.2 y §2.3.3).

Existen varias maneras de esquivar esta "funcionalidad" de la tabla de rutas, pero ninguna resulta satisfactoria como solución completa y general para dispositivos multiconectados:

- Los usuarios y administradores de sistemas Windows que se enfrentan a este problema, pueden recurrir a modificar el orden en que aparecen los interfaces en la tabla de rutas. Existe incluso un diálogo en la configuración del sistema operativo que permite elegir en qué orden se van a cargar los interfaces de red en la tabla de rutas con el siguiente reinicio del sistema[2]. Ni que decir tiene que reiniciar el sistema con cada traspaso probablemente no resulte *imperceptible* para muchas aplicaciones.

- Algunos sistemas permiten seleccionar la interfaz de salida mediante encaminamiento basado en políticas (policy based routing). Por ejemplo en sistemas GNU/Linux se puede elegir la interfaz de salida basándose en criterios como el puerto y protocolo de aplicación de cada comunicación (ver "Múltiples espacios de nombres de red", en la página 63).

  Aunque esta solución puede resultar conveniente en algunos casos, no resuelve completamente el problema ya que las aplicaciones en espacio de usuario no pueden configurar estas políticas dinámicamente, es una solución pensada para ser utilizada por administradores del sistema en entornos estáticos.

- Una solución que resulta bastante sencilla en terminales de usuario es desactivar los interfaces que no se quieren utilizar, dejando en funcionamiento únicamente el que si queremos utilizar. Esto elimina de la tabla de rutas los interfaces no deseados y deja como única alternativa el preferido por el usuario.

  Cualquiera que tenga un servidor de ficheros en casa y un portátil con interfaces Wi-Fi y Ethernet habrá utilizado esta solución cuando los datos a descargar al portátil, por su gran tamaño, resulten en una transmisión demasiado lenta para realizarla por Wi-Fi (se acerca uno con el portátil al concentrador Ethernet, se conecta a él, desactiva la Wi-Fi y comienza la descarga de los datos).

  Esto tampoco resulta una solución general aceptable, ya que las comunicaciones que ya estaban en curso por otros interfaces se verán interrumpidas.

[2] `http://support.microsoft.com/kb/894564`.

- Una variante un poco más técnica de lo anterior es modificar la tabla de rutas del núcleo para alterar el encaminamiento de las comunicaciones. Desgraciadamente esto alterará el encaminamiento de todas las comunicaciones del dispositivo, no solo de la aplicación en cuestión, por lo que no resulta una solución interesante en el caso general.

- Las aplicaciones UNIX diseñadas para seleccionar el interfaz de red, se saltan la API de sockets tradicional, y utilizan la API de raw sockets. Esto permite saltarse la tabla de rutas y elegir el interfaz de salida de sus comunicaciones, al precio de tener que implementar en la propia aplicación las pilas de protocolos de niveles superiores (IP y TCP por ejemplo).

  Es por eso que este método se utiliza sobre todo en aplicaciones que no utilizan protocolos complejos, como es el caso de `ping` o aplicaciones que no son las destinatarias finales de los datos, como `tcpdump`.

En el caso especial de aplicaciones sobre SCTP, la solución a este problema podría ser mucho más sencilla: se modifica la tabla de rutas para que encamine los paquetes preferentemente por la interfaz indicada por la dirección IP de origen de los mismos.

Esta nos parece una solución ideal para aplicaciones SCTP, ya que independientemente de si la aplicación a elegido la interfaz a utilizar recopilando ella misma los criterios que le interesan, delegando en el usuario o incluso utilizando el servicio que proponemos en este capítulo, bastaría con que la aplicación iniciase un petición de cambio de dirección primaria a SCTP para hacer efectivo el cambio. Esta afirmación es puramente especulativa ya que en el desarrollo de esta tesis no hemos abordado directamente la problemática de como hacer recomendaciones a la tabla de rutas desde SCTP.

En resumen, la mayoría de aplicaciones no eligen por que interfaz encaminar sus datos, principalmente por que hacerlo por si mismas implica complicar su portabilidad y por que una vez lo hubiesen averiguado tampoco podrían forzar fácilmente el uso de una interfaz u otra.

El resto de este capítulo está dedicado a describir un nuevo servicio del sistema operativo que facilita la recopilación, por parte de las aplicaciones, de datos sobre criterios de selección de interfaz, su diseño, implementación y nuestras experiencias utilizándolo. El problema de como forzar el uso de una interfaz u otra sin recurrir a soluciones ad-hoc no lo hemos abordado en esta tesis.

7.2 Diseño

Estamos convencidos que la publicación de criterios de seleción de interfaces es una tarea que debe ser proporcionada por el sistema operativo. Desde el punto de vista de las aplicaciones, hay dos maneras inmediatas de sacarle partido a este tipo de servicio:

- Las aplicaciones pueden seleccionar la interfaz que les resulte más interesante utilizar y pedirle al sistema operativo que encamine sus paquetes

```
; tree /sys/kernel/netqos
.
|-- figures/
|   |-- bw/
|   |   |-- eth0
|   |   |-- eth1
|   |   |-- ...
|   |   `-- units
|   |-- loss/
|   |   |-- eth0
|   |   |-- eth1
|   |   |-- ...
|   |   `-- units
|   `-- ...
|-- policies/
|   |-- on_battery/
|   |   `-- preferred
|   `-- ...
`-- version
```

Figura 7.1: Un listado incompleto pero representativo de los ficheros proporcionados por netqos.

por ella.

- Las aplicaciones pueden adaptar sus flujos de datos a las capacidades específicas de las interfaces disponibles para poder sacarle el mayor partido posible a sus comunicaciones.

Proponemos utilizar un sistema de ficheros sintético, que describimos más adelante en §7.3, para almacenar y publicar los criterios de selección de interfaces.

El contenido de los ficheros que ofrece este nuevo sistema de ficheros serán datos y figuras de mérito de las diferentes interfaces, que las aplicaciones y middleware de movilidad podrán consultar y modificar.

Organizamos estos ficheros en una estructura jerárquica de directorios. Los nombres de los ficheros y directorios recuerdan al tipo de datos que almacenan y a la interfaz a la que aplican.

El nombre que hemos elegido para este nuevo servicio, *netqos*, resulta engañoso, ya que su función es proporcionar información sobre varios tipos de criterios de selección de interfaces, no sólo de calidad de servicio. De hecho, la calidad de servicio es uno de los criterios que peor se representan en nuestra implementación como veremos más adelante en §7.4. Aún así, hemos mantenido el nombre por razones históricas.

Una vez montado, el sistema de ficheros netqos tiene el aspecto que se muestra en la figura 7.1: en ella se presenta un listado parcial de los ficheros ofrecidos

por netqos en nuestro dispositivo de pruebas, que dispone de varias interfaces (`eth0`, `eth1`...). De ahora en adelante, nos referiremos a los ficheros y directorios de netqos mediante su path relativo al punto de montaje de netqos.

El fichero `version` almacena la versión del sistema de ficheros que se está ejecutando; los procesos que usen netqos pueden consultar este fichero para evitar problemas de incompatibilidades entre versiones.

Los directorios como `figures/bw/` o `figures/loss/`, contienen los ficheros correspondientes a la información que se espera de sus nombres, ancho de banda y pérdidas, por ejemplo. Los ficheros que contienen estos directorios tienen el nombre de la interfaz a la que se refieren.

Los ficheros `figures/*/units` informan de las unidades de medida en que se representan los datos del resto de ficheros de su mismo directorio.

La información que proporciona netqos es bastante *cruda*: una representación textual del último valor que se ha medido en cada uno de los criterios de selección de interfaz soportados.

Para tomar decisiones teniendo en cuenta combinaciones complejas de estos valores se necesita generar figuras derivadas de ellos, como por ejemplo sistemas con historia o umbrales de histéresis. Para implementar estas figuras más complejas se utilizan procesos externos que las calculan a partir de los valores en crudo proporcionados por netqos y que las publican en sus correspondientes ficheros `policy/*/preferred`.

7.2.1 Procesos que usan netqos

La figura 7.2 muestra un diagrama del tipo de procesos que usan netqos.

- **Demonios de toma de medidas** que recopilan datos sobe los diferentes criterios de selección de interfaces. Escribirán sus medidas directamente a los ficheros correspondientes en netqos. Por ejemplo, un demonio que mida las pérdidas de la interfaz `eth0` una vez por segundo, escribirá este valor en el fichero `figures/loss/eth0` cada segundo.

- **Aplicaciones** interesadas en los criterios de selección de interfaz. Leerán los ficheros servidos por netqos. Por ejemplo, un servidor de streaming de vídeo puede leer los valores de `figures/bw/*` y adaptar la codificación del vídeo que está enviando a las limitaciones de ancho de banda de la interfaz que está utilizando o pedir al núcleo que le asigne otra interfaz con mejor ancho de banda.

- **Demonios de políticas**, que vigilan las diferentes figuras y datos ofrecidos por netqos y calculan figuras de mérito complejas a partir de ellas. Publican sus resultados en `policy/*/preferred` y las aplicaciones pueden consultar estos ficheros si prefieren delegar su selección de interfaz a alguno de estos implementadores de políticas. Por ejemplo, un demonio puede recomendar la interfaz más barata, simplemente vigilando `figures/price/*`, procesando estos valores y escribiendo el nombre de la interfaz más barata en `policy/cheapest/preferred`.

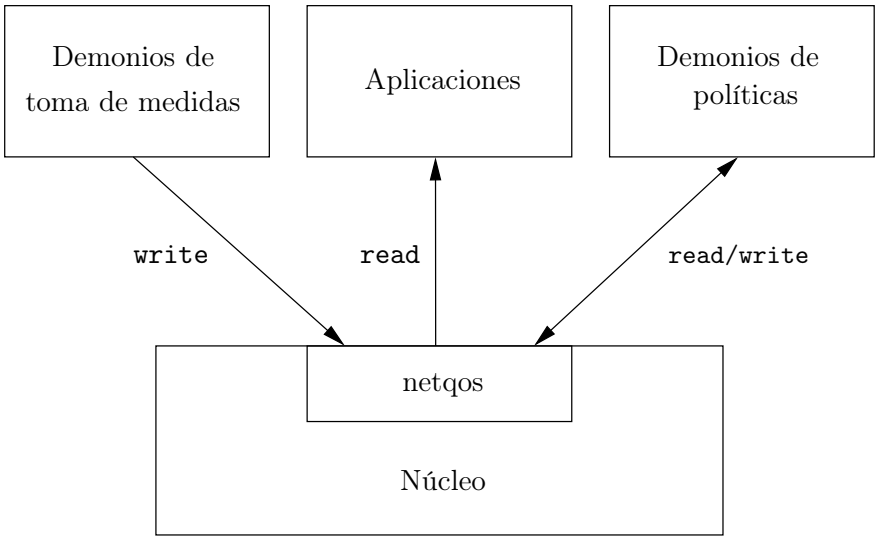

Figura 7.2: Tipos de procesos que usan netqos y las operaciones que típicamente realizan.

7.2.2 Un ejemplo de uso

Los datos leídos de los ficheros ofrecidos por netqos serán una representación textual de los valores internos almacenados por el módulo e incluirán un terminador `\n` para una mejor experiencia de usuario.

Los datos escritos en los ficheros ofrecidos por netqos no necesitan terminar en `\n`, pero si lo hacen no será tampoco un problema.

Un usuario puede leer o escribir estos ficheros usando cualquier tipo de comandos disponibles para acceder al sistema de ficheros:

```
1 ; cd /sys/kernel/netqos/
2 ; cat figures/loss/eth0
3 0.003
4 ; cat figures/loss/units
5 % of packets loss
6 ; echo 2.34 > figures/loss/eth0
7 ; cat figures/loss/eth0
8 2.34
```

Listado 7.1: Mostrar las pérdidas de la interfaz eth0 e introducir una nueva medida

Por supuesto el sistema de ficheros puede ser accedido también desde programas utilizando la API estándar del sistema operativo (`open(2)`, `read(2)`, `write(2)`, `close(2)`...).

7.3 Implementación

7.3.1 ¿Por qué un sistema de ficheros?

Existen varias interfaces de comunicación entre el núcleo y el espacio de usuario que pueden utilizarse para implementar un servicio como el que proporciona netqos: nuevas llamadas al sistema, nuevas variaciones de ioctls, notificaciones asíncronas usando señales, sockets corrientes o de tipo netlink y sistemas sintéticos de ficheros.

Hemos decidido utilizar un sistema de ficheros porque la API que exporta al desarrollador es extremadamente simple y universalmente conocida. Simplemente se necesita poder leer y escribir ficheros, cosa que puede realizar cualquier programado nobel en cualquier lenguaje de programación. Esto facilita la integración del sistema en aplicaciones mientras se mantiene el servicio portable a cualquier sistema operativo con soporte de ficheros.

Desde el punto de vista del usuario, existe una larga tradición de aplicaciones bien conocidas que pueden utilizarse para manejar esta API, como por ejemplo `ls(1)`, `cat(1)`, o las redirecciones de entrada/salida de la shell. Esto permite crear scritps fácilmente y agiliza la creación de prototipos para aplicaciones nuevas.

7.3.2 Detalles de la implementación

Hemos programado netqos como un módulo para el núcleo de Linux, versión 2.6.29, usando el soporte de `sysfs` ([106]) para sistemas de ficheros sintéticos. El código fuente al completo de netqos está disponible desde `https://github.com/alcortes-uc3m/netqos`.

Sysfs es un sistema de ficheros virtual que exporta información del núcleo de Linux sobre dispositivos y controladores (drivers) a espacio de usuario. Los ficheros que exporta existen en la memoria del núcleo, por lo que no ocupan espacio de disco y pueden ser accedidos por las mismas llamadas al sistema y aplicaciones que los ficheros tradicionales. Es el equivalente de `procfs` ([107]) para dispositivos y controladores, en lugar de para procesos.

Un sistema sintético de ficheros es la denominación genérica para una interfaz jerárquica a objetos que, no siendo ficheros, aparecen como si fueran ficheros corrientes. `Sysfs` resulta especialmente apropiado para construir sistemas de ficheros sintéticos en Linux.

El núcleo de la implementación de netqos está terminado y funcionando, sin embargo, los datos y figuras de mérito que exporta y la organización jerárquica es, intencionadamente, abierta y flexible. A lo largo de la realización de esta tesis hemos utilizado este servicio en varios proyectos de investigación, donde hemos ido recopilando las preferencias y necesidades de los diferentes miembros colaboradores en estos proyectos. El estado actual de netqos refleja estas circunstancias.

7.4 Limitaciones

Interfaces dinámicas

Netqos *no* soporta la carga dinámica de interfaces, ya que construye el árbol de ficheros y directorios en el momento en que se carga el módulo. Netqos no es consciente de cuando una nueva interfaz se activa o una interfaz ya existente se desactiva.

Resolveremos esta limitación fundamental en futuras versiones, haciendo que el módulo esté al tanto de las notificaciones del núcleo sobre activación dinámica de interfaces. Se puede utilizar un fichero `events` en lo más alto del árbol de directorios para notificar a los procesos que usan netqos las modificaciones sufridas por el árbol de directorios como consecuencia de la carga dinámica de interfaces.

Módulos de políticas en espacio de usuario

Las políticas de selección de interfaces deben mantenerse fuera del núcleo del sistema operativo. La implementación actual de netqos sólo permite el uso de una única política en espacio de usuario, a través del fichero `preferred`. Es necesario extender la funcionalidad de netqos para que sean los propios demonios de políticas los que puedan solicitar al módulo la creación de sus propios directorios bajo `policies/`.

Espacios de nombres de red múltiples

Netqos no soporta múltiples espacios de nombres de red ([108]). El módulo sólo soporta el espacio de nombres de red por defecto. La funcionalidad del núcleo de ofrecer múltiples espacios de nombres de red es de reciente incorporación y no se usa habitualmente. Una posible ampliación de netqos podría incluir el soporte de múltiples espacios de nombres de red utilizando el nuevo soporte que sysfs incorpora a tal efecto.

Criterios de selección extremo a extremo

Netqos no resulta útil para la publicación de criterios de selección de interfaces que tienen en cuenta figuras extremo a extremo de una comunicación: por ejemplo, el ancho de banda disponible en la ruta concreta que conecta nuestro dispositivo con un determinado servidor de streaming.

La recopilación de criterios de selección de interfaces extremo a extremo es más compleja que obtener datos sobre la interfaz en sí y en ocasiones requiere de la colaboración del otro extremo.

Además de esto, un servicio en el sistema operativo capaz de representar datos extremo a extremo debería tener las siguientes características:

1. **Altamente dinámico**: debido a que los interlocutores con los que se mantienen conexiones pueden aparecer y desaparecer rápidamente, según las aplicaciones que ejecuten en el dispositivo abran y cierren sucesivas conexiones de red.

2. **Extremadamente escalable**: ya que el número de interlocutores con los que se mantienen conexiones de red puede ser muy elevado.

Por nuestra experiencia en el uso de netqos, un sistema de ficheros sintético, al menos en la forma en que lo hemos implementado, puede resultar complejo de utilizar en escenarios con estas características.

En general, los criterios extremo a extremo son poco aprovechables por otras aplicaciones que no sean la directamente involucrada en la comunicación entre esos extremos.

La necesidad de incluir un nuevo servicio en el sistema operativo para compartir este tipo de datos es discutible. El diseño de netqos se ha centrado en aquellos criterios que si pueden resultar comunes a todas las aplicaciones que ejecutan en un dispositivo.

Aún así, netqos puede seguir siendo útil para criterios extremo a extremo en otro sentido: se puede aprovechar este servicio para que las aplicaciones suministren datos sobre criterios extremo a extremo a los demonios implementadores de políticas, enriqueciendo sus decisiones y liberando a las propias aplicaciones de implementar sus propios decisores o implementadores de políticas.

7.5 Experiencias de uso

En esta sección resumimos las experiencias que nosotros y terceras personas han tenido utilizando las contribuciones de esta tesis, no sólo lo concerniente a este capítulo (netqos), sino también de los anteriores ([109, 110]).

A lo largo de los últimos 4 años hemos tenido ocasión de utilizar nuestras contribuciones a esta tesis en los proyectos de investigación EW y EW2.

En el proyecto EW utilizamos netqos para integrar el trabajo de los diversos miembros del proyecto:

- Algunos de los miembros del proyecto contribuían con diversos mecanismos de medida de prestaciones y calidad de servicio, tanto activos como pasivos, locales y distribuidos.

- Otros miembros contribuyeron con demonios implementadores de políticas que leían diversas medidas de los criterios de netqos, las procesaban y publicaban en el propio netqos sus recomendaciones acerca de qué interfaz de red utilizar preferentemente.

- Otros miembros debían utilizar estas recomendaciones para realizar traspasos en los momentos adecuados, de forma que se mantuviese una cierta calidad de servicio en la recepción. Los mecanismos de movilidad utilizados fueron SCTP y MIP.

- Los criterios de selección de interfaz publicados en netqos se utilizaban también para modificar las características del tráfico intercambiado por las aplicaciones del demostrador (principalmente servidores y reproductores de streams de vídeo) para adaptar los datos a las capacidades de las

interfaces utilizadas (por ejemplo, reproduciendo un vídeo de baja calidad cuando las limitaciones de la interfaz en uso no permitía la transmisión de vídeo en alta calidad).

Los desarrollos de los diferentes miembros del proyecto utilizaban diversos lenguajes de programación, como Java, C y C++ y estaban programados en plataformas diferentes (Windows XP y Linux 2.6 en ordenadores de sobremesa, ordenadores portátiles, tabletas y móviles de altas prestaciones). Las redes utilizadas eran redes locales convencionales como Ethernet y Wi-Fi y redes de telefonía móvil como GPRS y UMTS.

La opinión general de los miembros del proyecto fue que la integración se produjo de forma sencilla gracias a netqos y su interfaz de sistema de ficheros.

Los diferentes miembros del proyecto pudieron simular los desarrollos del resto de miembros mediante sencillos scripts que utilizaban netqos. La preparación y desarrollo de la demostración final del proyecto se benefició especialmente de netqos, ya que facilitó la elaboración de prototipos y la concepción y puesta en marcha de diferentes escenarios de pruebas de forma rápida y sencilla.

Para el proyecto EW2 planteamos ampliar la activación de traspasos tradicional (mediante medidas de calidad de servicio) con el sondeo activo de SCTP-AP ([111]).

Aunque el proyecto se canceló a mitad, tuvimos ocasión de comprobar como la detección de fallo de ruta de SCTP-AP resulta un complemento ideal al resto de aplicaciones de medida de calidad de servicio, ya que las desconexiones por fallos de ruta son especialmente rápidas de detectar mediante SCTP-AP.

Posteriormente a la cancelación del proyecto EW2 hemos tenido ocasión de seguir trabajando, informalmente ([112]), con algunos de los miembros del proyecto, que han mostrado especial interés en aprovechar los resultados del estudio de estrategias de observación (§5) en sus propios sistemas de monitorización de calidad de servicio en el núcleo de redes tipo GPRS.

7.6 Conclusiones

Hemos desarrollado un nuevo servicio del sistema operativo, llamado netqos, que almacena y publica los datos y figuras de mérito de algunos criterios de selección de interfaces. Este servicio está implementado como un sistema de ficheros sintético para el núcleo de Linux.

Netqos ha demostrado ser útil y ha facilitado la tarea del desarrollo e integración de aplicaciones y middleware de movilidad entre los participantes de dos proyectos de investigación.

Sin embargo, netqos muestra importantes limitaciones, la más importante de todas es su ineficiencia para publicar y almacenar criterios de selección de interfaces extremo a extremo, debido a su dinamismo y escalabilidad.

Parte IV

Cierre

Capítulo 8

Conclusiones y trabajos futuros

8.1 Resumen de nuestras contribuciones

- **SCTP-AP** (§4) es una modificación al protocolo SCTP que permite a las aplicaciones ajustar la duración de la detección de fallo de ruta al valor que más les convenga.

  Cuando se dispone de varias interfaces de red, detectar con prontitud un fallo de ruta permite continuar la comunicación por una interfaz alternativa a tiempo de mantener la calidad de servicio percibida por la aplicación. Por tanto, el objetivo de esta modificación es facilitar traspasos imperceptibles en SCTP a aplicaciones con diferentes requerimientos en cuanto al tiempo de detección de fallo de ruta.

  El sistema de detección de fallo de ruta que proponemos se basa en un sondeo activo de las rutas reutilizando antiguos mensajes de control del propio SCTP. Hemos tomado precauciones para asegurar la estabilidad de nuestra propuesta.

- **Estudio de estrategias de observación** (§5). Nuestra propuesta de modificación del mecanismo de detección de fallo de ruta en SCTP da libertad en cuanto a cómo planificar temporalmente las sondas que se envían.

  Con el objeto de decidir qué planificación temporal de las sondas resulta más eficiente, hemos realizado un estudio matemático genérico de las estrategias de sondeo. Nuestras conclusiones son generales y aplicables a cualquier otro ámbito científico en el que se realicen sondeos para detectar la ocurrencia de un evento.

  Mas concretamente, en §5.2 y en §5.3, proponemos una notación y unos fundamentos matemáticos que permiten estudiar las propiedades de dife-

rentes estrategias de observación para la detección de un evento.

En §5.4 y en §5.5 aplicamos estos fundamentos para analizar dos estrategias particularmente importantes en el caso de SCTP, la estrategia periódica y la geométrica, y en §5.6 las comparamos.

En §5.7 demostramos por qué la estrategia periódica es la estrategia óptima para reducir el tiempo medio de detección en protocolos de propósito general.

En §5.8 estudiamos el impacto sobre el retardo medio de detección de someter la programación de las sondas de una estrategia periódica a una variación aleatoria, ya que resulta útil en entornos de protocolos de comunicaciones para reducir el riesgo de sincronismos entre diferentes comunicaciones que compartan rutas.

En §5.9 resumimos los resultados de nuestro estudio de estrategias mediante un mapa en dos dimensiones (retardo medio de detección frente a número de sondas) que permite comparar estrategias entre si de forma visual y sencilla.

- **Lista de los criterios de selección de interfaces** (§6). Hay diversas razones para utilizar una interfaz de red u otra a la hora de encaminar nuestras comunicaciones en un dispositivo de usuario multiconectado. En este capítulo enumeramos los posibles criterios de selección a tener en cuenta.

  El objetivo de elaborar esta lista es que sirva de referencia a la hora de implementar nuevos servicios del sistema operativo para la selección de la interfaz más apropiada.

- **Netqos** (§7) es nuevo servicio del sistema operativo para la publicación y consulta de criterios de selección de interfaces. El objetivo de este servicio es facilitar a las aplicaciones la elección de la interfaz más apropiada para sus comunicaciones.

  En §7.2 presentamos cómo hemos diseñado este servicio y en §7.3 explicamos los detalles de nuestra implementación de referencia: un sistema de ficheros sintético para Linux.

  En §7.4 revisamos las limitaciones de nuestra implementación y concluimos el capítulo con §7.5, donde presentamos nuestras experiencias de uso del servicio, en conjunto con el resto de contribuciones de esta tesis.

8.2 Conclusiones

El objetivo de esta tesis es mejorar el aprovechamiento inteligente de las múltiples interfaces de red con las que vienen equipados los dispositivos de usuario modernos.

Idealmente, deberíamos ser capaces de transferir los diferentes flujos de nuestras aplicaciones de unas interfaces a otras según las condiciones en las que nos

encontremos, sacándole el máximo partido a nuestras interfaces de red, a ser posible, sin demasiada intervención por parte del usuario.

Al comienzo de nuestro trabajo identificamos los principales problemas que impiden aprovechar convenientemente la multiconexión de los dispositivos modernos:

- Los sistemas operativos no ofrecen un servicio que permita a las aplicaciones averiguar cual es la interfaz más apropiada en cada momento.

- Los protocolos de transporte presentan problemas para sobrevivir a un cambio de interfaz o resultan ineficientes en estas circunstancias.

El primero de los problemas, acerca de cómo proporcionar recomendaciones a las aplicaciones sobre qué interfaz de red utilizar, tiene dos vertientes. Primera, ¿cuáles son los criterios a tener en cuenta para elegir una interfaz u otra?, y segunda, ¿cómo recopilar estos criterios, evaluarlos y hacerle llegar a las aplicaciones las recomendaciones?

En §6 hemos recopilado una lista completa de los criterios a tener en cuenta para la selección de interfaces. Los criterios son de lo más variado, desde preferencias de usuario, a recomendaciones del proveedor de servicio, pasando por la calidad de servicio disponible en cada interfaz.

Algunos de estos criterios son sencillos de recopilar, por ejemplo, el sistema operativo tiene datos precisos sobre las características de transmisión de las interfaces del dispositivo o el estado de la batería, aunque en ocasiones no existe una API estándar para acceder a estos datos.

Otros criterios resultan más complicados de obtener o de expresar, como por ejemplo las preferencias de usuario, que involucran una comunicación directa entre dispositivo y usuario, y el diseño de una semántica apropiada para ello.

Algunos de los criterios involucran la colaboración de terceras personas o entidades, por ejemplo, los criterios de calidad de servicio extremo a extremo suelen requerir de la colaboración de los interlocutores en una comunicación y averiguar las preferencias del proveedor de servicio implica su colaboración directa.

En general, no es necesario tener en cuenta todos los criterios que hemos enumerado, la mayor parte de las veces tendremos datos suficientes para una buena elección de la interfaz considerando sólo unos pocos.

De hecho, algunos de estos criterios no pueden ser tomados en cuenta hoy en día por falta de mecanismos, tecnologías y aplicaciones capaces de medirlos con precisión.

Aún así, creemos que resulta valioso contar con una lista completa de criterios de selección de interfaces, de forma que si se ignoran algunos criterios, al menos se haga conscientemente.

Independientemente del número de criterios a considerar, pensamos que no debe ser la aplicación la encargada de recopilarlos y evaluarlos. Debería ser un servicio del sistema operativo el que recopilase los datos sobre los diferentes

criterios. Estos datos pueden venir tanto de información interna al propio sistema operativo como de demonios que, ejecutando en segundo plano, son los encargados de medirlos.

El sistema operativo tampoco debería encargarse de implementar políticas concretas de selección de interfaz a partir de los datos recopilados. Demonios implementadores de políticas, corriendo en espacio de usuario, pueden hacer uso de los datos recopilados por el sistema operativo y evaluarlos para ofrecer recomendaciones específicas.

Esto abre la puerta al desarrollo de diferentes políticas, que coexisten en el sistema, siendo las aplicaciones y el usuario, los que deciden de que política fiarse. La competencia para el desarrollo de estas políticas es libre y puede dar lugar a un amplia variedad de desarrollos y oportunidades de negocio. Unas buenas políticas de selección pueden ser el elemento principal de la campaña de mercadotecnia de un nuevo dispositivo, por ejemplo.

En §7 presentamos un nuevo servicio para el sistema operativo diseñado con este propósito. Permite publicar datos y figuras de mérito sobre diferentes criterios de selección de interfaz. También permite que sean demonios implementadores de políticas los que se encarguen de calcular figuras de mérito avanzadas sobre las diferentes medidas y generen recomendaciones específicas. Las aplicaciones pueden suscribirse a las recomendaciones de sus implementadores de políticas favoritos para dejarse aconsejar sobre qué interfaz deben utilizar.

El diseño de un servicio de este tipo, así como su interfaz a las aplicaciones y su comodidad de uso depende del tipo de criterios a tener en cuenta. Razón de mas para disponer de una lista completa de criterios y poder decidir a priori cuales van a ser considerados, ya que esto tiene un impacto directo sobre el diseño del servicio y su utilidad.

En cuanto al segundo de los problemas, acerca de las dificultades que tienen los protocolos de transporte para realizar traspasos de una interfaz a otra, creemos que SCTP lo resuelve exitosamente.

Sin embargo SCTP tiene sus propias limitaciones, que impiden un aprovechamiento adecuado de los dispositivos multiconectados. En concreto, la detección de fallo de ruta en SCTP tiene una duración excesiva para algunas aplicaciones.

En §4 hemos propuesto una modificación a SCTP que permite adaptar la duración de la detección de fallo de ruta a las necesidades de la aplicación. Nuestro objetivo no era tanto reducir el tiempo de detección de fallo de ruta, como poder adaptarlo a las necesidades de la aplicación, aunque, en ocasiones, ambas cosas son equivalentes.

En los últimos años se han propuesto algunas modificaciones al funcionamiento de SCTP para reducir el tiempo de detección de fallo de ruta, ninguna resulta tan configurable por la aplicación como la nuestra, que además está diseñada para eliminar los compromisos típicos de esas otras alternativas: en nuestra propuesta, los límites de la duración de la detección de fallo de ruta no los imponen el protocolo de transporte, sino las características de transmisión de la ruta, que es como creemos que debe ser.

Otro de los problemas importantes de SCTP es que retrasa innecesariamente el envío de los datos cuando la ruta sufre fallos temporales de corta duración.

Este tipo de fallos es habitual en redes inalámbricas, donde las pérdidas temporales de la cobertura son habituales. Dado el tipo de dispositivos de usuario a los que tiende el mercado últimamente y las interfaces que incorporan, pensamos que debemos abordar este problema o arriesgarnos a que una solución basada en SCTP no resulte atractiva.

En §5 hemos estudiado las propiedades de las estrategias de sondeo y hemos identificado qué estrategia de sondeo proporciona el menor retardo de detección de un fallo de ruta temporal.

Nuestra modificación a SCTP incorpora un sondeo de este tipo, de forma que la recuperación de los fallos de ruta temporales es mejor que en el caso de usar SCTP.

La estrategia de sondeo que proponemos no es la óptima, ya que en ese caso se presentarían algunos problemas de estabilidad por sincronismo. Para evitar este problema, incorporamos una variación aleatoria a la planificación de las sondas que reduce el riesgo de sincronismo a costa de retrasar la detección de fallos de ruta temporales. Aún así, la detección de fallos de ruta de nuestra propuesta es mejor que en SCTP.

Creemos que las contribuciones de esta tesis, en conjunto, resuelven los principales problemas que existen hoy en día para aprovechar inteligentemente los dispositivos de usuario modernos y abren la puerta al desarrollo de tecnologías que enriquecerán la experiencia de los usuarios finales.

8.3 Vías de investigación futuras

- La mayoría de resultados que hemos obtenido como consecuencia del estudio de estrategias de observación sólo son aplicables a sucesos uniformes.

  Sin embargo, en determinados entornos sí se puede conocer la distribución estadística de las recuperaciones de ruta o al menos estimarla.

  En estos casos sería muy provechoso poder deducir la estrategia de sondeo óptima que permite el menor tiempo medio de detección para experimentos con distribuciones de probabilidad diferentes a la uniformes, en especial para entornos inalámbricos donde las pérdidas de conectividad pueden seguir patrones bien definidos.

  Para ello es necesario generalizar nuestros resultados a distribuciones genéricas, algo a lo que nos hemos acercado ya mediante observaciones empíricas y simulaciones pero que todavía no hemos sabido formular y demostrar matemáticamente.

- La reducción de la agresividad de SCTP-AP mediante la reserva en la ventana de congestión de pérdidas de los octetos a consumir por las sondas activas tiene el efecto adverso de reducir la velocidad de reacción en la competencia con otros flujos.

  Nos gustaría evaluar con detalle el impacto de esta medida en el goodput en función del número de sondas a enviar, de forma que encontremos

un compromiso entre la probabilidad de falsos positivos en la detección, consecuencia de reducir el número de sondas, frente al goodput ganado con la medida.

- Es necesario introducir una variabilidad en la planificación temporal de las sondas de SCTP-AP para evitar sincronismos entre asociaciones que compartan ruta y valores de configuración de los parámetros del protocolo.

  Sin embargo, como se ha demostrado, cualquier variación sobre la planificación periódica pura provoca un incremento del tiempo medio de detección de la recuperación de la ruta.

  Nos gustaría estudiar en qué proporción alivia los problemas de sincronismo el hecho de introducir variaciones de diferente magnitud, de forma que se pueda llegar a un compromiso aceptable en la disminución del tiempo medio de detección de la recuperación de la ruta y el riesgo de sincronismo.

- Nuestra propuesta para reducir el tiempo de detección de fallo de ruta es aplicable también a escenarios de transferencia simultánea por varias interfaces.

  Sin embargo, cuando se utiliza transferencia simultánea las consideraciones de estabilidad se complican y los ajustes a realizar en SCTP-AP son más complejos.

  En general, el envío de datos mediante transferencia simultánea es nuevo para nosotros y creemos que podemos aprovechar nuestra experiencia con dispositivos multiconectados de usuario para contribuir a este nuevo campo.

- Durante la realización de esta tesis hemos intentado incluir nuestra propuesta de modificación de SCTP en el trabajo que está realizando el TSVWG para reducir la duración de la detección de fallos de ruta en SCTP.

  Aunque hasta ahora nuestras contribuciones no han gozado del respaldo del TSVWG, pensamos seguir trabajando en colaborar con ellos en este objetivo común.

- Uno de los problemas fundamentales de SCTP es que la mayoría de aplicaciones que se usan hoy en día tendrían que modificarse para utilizar este protocolo de transporte.

  Una manera de solucionar esto sería llevar a cabo una campaña de infiltración en diversos proyectos de software famosos para dotarles de soporte SCTP.

  También sería interesante perfeccionar los modelos SCTP de los diferentes simuladores de red utilizados en investigación con el fin de impulsar el uso de esta tecnología.

- Publicar algunos criterios de selección de interfaces mediante una abstracción de sistema de ficheros ha sacado a la luz algunos problemas prácticos. Es necesario seguir trabajando en resolverlos, o pensar en nuevas abstracciones para la implementación de este servicio, que nos parece fundamental en los sistemas operativos modernos.

  En especial nos gustaría profundizar en la representación de criterios extremo a extremo, ya que algunos de ellos, como la calidad de servicio, son fundamentales para una elección apropiada de la interfaz a utilizar, ya que representan una fuente de información fundamental para los demonios implementadores de políticas.

- Creemos que la selección de interfaz basada en las recomendaciones de los proveedores de servicio es un tema novedoso y con mucho futuro.

  Sin embargo, durante la realización de esta tesis sólo hemos llegado a arañar la superficie. Los problemas que plantea, de seguridad y semántica, y las oportunidades que abre en cuanto a dimensionamiento dinámico de redes nos parecen interesantes.

  Recientemente hemos empezado un proyecto de investigación ([113]) que trata sobre estos temas y nos gustaría profundizar en ellos, pues las oportunidades de contribuir en un tema tan novedoso nos resultan atractivas.

- De poco sirve un servicio como netqos si el sistema no dispone de aplicaciones de medida de los diferentes criterios de selección de interfaz.

  Es necesario concretar aún más los criterios, identificar que aplicaciones proporcionan medidas sobre ellos e incorporarlas al sistema operativo base. En los casos en que no existan aplicaciones apropiadas para la medida de determinados criterios, se abre una excelente oportunidad para nuevos desarrollos e investigaciones.

- Los detalles de implementación de servicios del sistema operativo para la selección de interfaces dependen en gran medida de que tipo de criterios se van a considerar, principalmente por su dinamismo y escalabilidad.

  Sería conveniente agrupar los diferentes criterios según estas propiedades, intentando prever que características debería cumplir un servicio de selección de interfaz que quisiera considerarlos.

- La tabla de rutas del núcleo es hoy en día un obstáculo para la correcta elección de la interfaz en dispositivos multiconectados.

  Aunque para el caso de SCTP (y de SCTP-AP) se pueden proponer soluciones sencillas, creemos que sería mucho más interesante replantearse la funcionalidad de la tabla de rutas, de forma que se solucionase este problema *en general*, para cualquier protocolo de transporte o de red.

Apéndice A

Descripción formal de SCTP-AP

1. SCTP-AP

1.1. The AP state

The following paragraphs describe a new state for SCTP destination addresses, called Active Probing or AP, from now on.

A destination address enters the AP state if:

- The address is not already in the AP state.
- And it is active.
- And the association is in the ESTABLISHED state.
- And the T3-rtx (retransmission timer) for that address expires.
- And there is enough room in the initial congestion window for the reservation of the active probing heartbeats (see bellow).

A destination address leaves the AP state if:

- A new RTT measurement for that address is obtained (from now on path recovery). This happens only when new data has been sent and acknowledged by that address, as per Karn's algorithm, or when a heartbeat to that address is acknowledged.

- Or the address becomes inactive (from now on quick retransmit).
- Or the primary address is changed to an active address other than this one.
- Or the association enters the SHUTDOWN-ACK-SENT state.
- Or the association enters the CLOSED state.

While a destination address is in the AP state, a number of heartbeats, ActiveProbing.Burst, must be sent to it. If a destination address is in the AP state for more than ActiveProbing.GiveUp seconds, the address is marked as inactive.

The first heartbeat should be sent as soon as the destination address enters the AP state.

When a destination address leaves the AP state, the pending transmissions and retransmissions are scheduled immediately.

The heartbeats sent due to Active Probing should drain bytes from the initial congestion window. The needed portion of the cwnd must be reserved in the corresponding path when entering the AP state. If there are not enough bytes in the cwnd to send these heartbeats, the address SHOULD NOT enter the AP state.

1.2. Configuration parameters

The parameters ActiveProbing.Burst and ActiveProbing.GiveUp are new protocol parameters that can be configured by the upper layer.

The default value of ActiveProbing.Burst is PMR and the default value of ActiveProbing.GiveUp is 63*RTOmin.

The delay between each AP heartbeat (Thb) should be:

Thb = ActiveProbing.GiveUp / ActiveProbing.Burst + V

Where V is a random variable between 0 and Thb.

Apéndice B

Siglas

3GPP 3rd Generation Partnership Project. Colaboración entre varios organismos de telecomunicaciones para diseñar la tercera generación de telefonía móvil.

AAA Authentication, Authorization and Accounting. Autenticación, autorización y contabilidad, o los tipos de protocolos que llevan a cabo estas tareas.

ACK Acknowledgement. Asentimiento o tipo de segmento de control que se utiliza en algunos protocolos para confirmar la recepción de otro segmento.

API Application programming interface. Interfaz para el programador de aplicaciones.

AP Access point. Punto de acceso de una red inalámbrica.

dDNS Dynamic DNS, [36].

DNS Domain Name System, [21].

EW2 proyecto "Easy Wireless 2", [2].

EW proyecto "Easy Wireless", [1].

GPRS General packet radio service. Servicio de datos para móviles de segunda y tercera generación, basado en conmutación de paquetes.

GPS Global Positioning System. Sistema de navegación vía satélite.

HIP Host identfication protocol, [27].

HTTP Hypertext Transfer Protocol, [34].

IEEE Institute of Electrical and Electronics Engineers, asociación profesional, sin ánimo de lucro, para el avance en la innovación tecnológica y su excelencia.

IETF Internet Engineering Task Force, organismo para el desarrollo y promoción de estándares para Internet.

IP Internet Protocol, [114, 115].

MH mobile host, dispositivo móvil.

MIHF Media Independent Handover Function, protocolo para la distribución de información entre los diferentes elementos involucrados en MIH.

MIH Media Independent Handover, IEEE 802.21, [22].

MIP Mobile IP, [24, 25].

MIPv6 Mobile IP version 6, [26].

mSCTP Mobile SCTP, [32].

NIC Network Interface Card, tarjeta o interfaz de red.

OSI Open Systems Interconnection, modelo de interconexión entre sistemas abiertos.

OS Operating System, sistema operativo.

P2P Point to Point, aplicación punto a punto, por ejemplo Emule.

PFD Path Failure Detection, mecanismo de detección de fallo de ruta en SCTP.

PF Potential Failure, estado de fallo potencial, propuesto por [54] para mejorar el PFD en SCTP.

PMR Path Maximum Retransmissions, número de retransmisiones máximas admisibles en SCTP antes de marcar la ruta como inactiva.

RFC Request for Coments, documentos, publicados por el IETF, describiendo los métodos, comportamientos, investigaciones e innovaciones aplicables a tecnologías de Internet.

RPDB Routing Policy Database, base de datos del núcleo de Linux donde se establece el orden en que se utilizan las diferentes tablas de rutas.

SCTP-AP SCTP with Active Probing, nuestra propuesta de mejora del PFD de SCTP, descrita en §4.

SCTP Stream Control Transmission Protocol, [29].

SIP Session Initiation Protocol, [33].

SLM Session Layer Mobility, [46].

SMTP Simple Mail Transfer Protocol, [35].

SM Session Management, entidad de gestión de sesión en SLM.

TCP Transport Control Protocol, [116].

TOS Type of Service, campo en la cabecera de los datagramas IPv4, que ha sido utilizado para varios propósitos a lo largo de los últimos años, últimamente se utiliza para indicar el tipo de servicio diferenciado y para la notificación explicita de congestión.

TP Tasa de pérdidas, generalmente en tanto por ciento de paquetes perdidos.

TSN Transport Sequence Number, número de secuencia de transporte, el índice que lleva cada segmento de datos en SCTP para poder ordenarlos en recepción.

TSVWG IETF Transport Area Working Group, grupo de trabajo del IETF para temas del nivel de transporte que no están siendo ya tratados en otros grupos de trabajo específicos.

ULS User Location Server, base de datos de las localizaciones de los dispositivos en SLM.

UMTS Universal Mobile Telecommunications System, tecnología de telefonía móvil de tercera generación.

VoIP Voice over IP, aplicación de voz sobre IP, por ejemplo Skype.

Wi-Fi Nombre comercial para las interfaces del estándar IEEE 802.11.

Bibliografía

[1] *Easy wireless.* http://www.itea2.org/project/index/view/?project=86.

[2] *Easy wireless 2.* http://www.celtic-initiative.org/Projects/Celtic-projects/Call5/EW-2/ew2-default.asp.

[3] Knuth, Donald E., Tracy L. Larrabee y Paul M. Roberts: *Mathematical Writing.* Mathematical Association of America, 1989, ISBN 0-88385-063-X. http://tex.loria.fr/typographie/mathwriting.pdf.

[4] Spivak, Michael: *Calculus.* Cambridge University Press, 3ª edición, Agosto 2006, ISBN 0-521-86744-4. http://books.google.es/books?id=7JKVu_9InRUC.

[5] Abley, J., B. Black y V. Gill: *Goals for IPv6 Site-Multihoming Architectures.* Rfc 3582, Internet Engineering Task Force, Agosto 2003. http://www.rfc-editor.org/rfc/rfc3582.txt.

[6] Clayton, R.: *Internet multi-homing problems: Explanations from economics.* En *Eighth Annual Workshop on Economics and Information Security (WEIS09)*, London, UK, Junio 2009.

[7] de Launois, C. y M. Bagnulo: *The paths towards IPv6 multihoming.* IEEE Communications Surveys and Tutorials, 8(2), 2006.

[8] Ylitalo, Jukka, Tony Jokikyyny, Antti J. Tuominen y Jaakko Laine: *Dynamic network interface selection in multihomed mobile hosts.* En *Hawaii Intl. Conf. on System Sciences (HICSS'03)*, volumen 9, página 315. IEEE Computer Society, 2003, ISBN 0-7695-1874-5.

[9] Ernst, T., N. Montavont, R. Wakikawa, C. Ng y K. Kuladinithi: *Motivations and Scenarios for Using Multiple Interfaces and Global Addresses.* Internet-draft draft-ietf-monami6-multihoming-motivation-scenario-03, Internet Engineering Task Force, Mayo 2008. http://tools.ietf.org/html/draft-ietf-monami6-multihoming-motivation-scenario-03, Work in progress.

[10] Iyengar, Janardhan R., Paul D. Amer y Randall Stewart: *Concurrent multipath transfer using SCTP multihoming over independent end-to-end paths*. IEEE/ACM Trans. Netw., 14:951–964, Octubre 2006, ISSN 1063-6692. http://dx.doi.org/10.1109/TNET.2006.882843.

[11] Fu, Qiang, Jadwiga Indulska, Sylvie Perreau y Liren Zhang: *Exploring TCP parallelisation for performance improvement in heterogeneous networks*. Computer Communications, 30(17):3321–3334, 2007.

[12] Hasegawa, Yohei, Ichiro Yamaguchi, Takayuki Hama, Hideyuki Shimonishi y Tutomu Murase: *Deployable multipath communication scheme with sufficient performance data distribution method*. Computer Communications, 30(17):3285–3292, 2007.

[13] Yabandeh, Maysam, Sajjad Zarifzadeh y Nasser Yazdani: *Improving performance of transport protocols in multipath transferring schemes*. Computer Communications, 30(17):3270–3284, 2007.

[14] Manner, J. y M. Kojo: *Mobility Related Terminology*. Rfc 3753, Internet Engineering Task Force, Junio 2004. http://www.rfc-editor.org/rfc/rfc3753.txt.

[15] Droms, R.: *Dynamic Host Configuration Protocol*. Rfc 2131, Internet Engineering Task Force, Marzo 1997. http://www.rfc-editor.org/rfc/rfc2131.txt.

[16] Aboba, B., D. Thaler y L. Esibov: *Link-local Multicast Name Resolution (LLMNR)*. Rfc 4795, Internet Engineering Task Force, Enero 2007. http://www.rfc-editor.org/rfc/rfc4795.txt.

[17] *Zero configuration networking (Zeroconf)*. http://www.zeroconf.org/.

[18] *802.11r-2008 IEEE Standard for Information Technology-Telecommunications and Information Exchange Between Systems-Local and Metropolitan Area Networks-Specific Requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 2: Fast Basic Service Set (BSS)*, 2008. http://ieeexplore.ieee.org/servlet/opac?punumber=4573290.

[19] *802.11e-2005 IEEE Standard for Wireless LAN Medium Access Control (MAC) and Physical Layer Specifications (Amendment: Quality of Service (QoS) Enhancement)*, 2005. http://standards.ieee.org/findstds/standard/802.11e-2005.html.

[20] Rekhter, Y., T. Li y S. Hares: *A Border Gateway Protocol 4 (BGP-4)*. Rfc 4271, Internet Engineering Task Force, Enero 2006. http://www.rfc-editor.org/rfc/rfc4271.txt.

[21] Mockapetris, P.V.: *Domain names - concepts and facilities*. Rfc 1034, Internet Engineering Task Force, Noviembre 1987. `http://www.rfc-editor.org/rfc/rfc1034.txt`.

[22] IEEE Computer Society: *IEEE Standrad for local and metropolitan area networks - part 21: Media Independent Handover Services*, Enero 2009. `http://standards.ieee.org/getieee802/download/802.21-2008.pdf`.

[23] Gupta, Vivek: *IEEE p802.21 tutorial*, Julio 2006. `http://www.ieee802.org/21/Tutorials/802%2021-IEEE-Tutorial.ppt`.

[24] Perkins, C.: *IP Mobility Support for IPv4*. Rfc 3344, Internet Engineering Task Force, Agosto 2002. `http://www.rfc-editor.org/rfc/rfc3344.txt`.

[25] Perkins, C.: *IP Mobility Support for IPv4, Revised*. Rfc 5944, Internet Engineering Task Force, Noviembre 2010. `http://www.rfc-editor.org/rfc/rfc5944.txt`.

[26] Perkins, C., D. Johnson y J. Arkko: *Mobility Support in IPv6*. Rfc 6275, Internet Engineering Task Force, Julio 2011. `http://www.rfc-editor.org/rfc/rfc6275.txt`.

[27] Moskowitz, R. y P. Nikander: *Host Identity Protocol (HIP) Architecture*. Rfc 4423, Internet Engineering Task Force, Mayo 2006. `http://www.rfc-editor.org/rfc/rfc4423.txt`.

[28] Moskowitz, R., P. Nikander, P. Jokela y T. Henderson: *Host Identity Protocol*. Rfc 5201, Internet Engineering Task Force, Abril 2008. `http://www.rfc-editor.org/rfc/rfc5201.txt`.

[29] Stewart, R.: *Stream Control Transmission Protocol*. Rfc 4960, Internet Engineering Task Force, Septiembre 2007. `http://www.rfc-editor.org/rfc/rfc4960.txt`.

[30] Stewart, Randall R. y Qiaobing Xie: *Stream Control Transmission Protocol (SCTP), a reference guide*. Addison-Wesley Professional, 1ª edición, 2001. ISBN 0-201-72186-4.

[31] Stewart, R., M. Tuexen, K. Poon, P. Lei y V. Yasevich: *Sockets API Extensions for the Stream Control Transmission Protocol (SCTP)*. Rfc 6458, Internet Engineering Task Force, Diciembre 2011. `http://www.rfc-editor.org/rfc/rfc6458.txt`.

[32] Stewart, R., Q. Xie, M. Tuexen, S. Maruyama y M. Kozuka: *Stream Control Transmission Protocol (SCTP) Dynamic Address Reconfiguration*. Rfc 5061, Internet Engineering Task Force, Septiembre 2007. `http://www.rfc-editor.org/rfc/rfc5061.txt`.

[33] Rosenberg, J., H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley y E. Schooler: *SIP: Session Initiation Protocol*. Rfc 3261, Internet Engineering Task Force, Junio 2002. `http://www.rfc-editor.org/rfc/rfc3261.txt`.

[34] Fielding, R., J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach y T. Berners-Lee: *Hypertext Transfer Protocol – HTTP/1.1*. Rfc 2616, Internet Engineering Task Force, Junio 1999. `http://www.rfc-editor.org/rfc/rfc2616.txt`.

[35] Klensin, J.: *Simple Mail Transfer Protocol*. Rfc 5321, Internet Engineering Task Force, Octubre 2008. `http://www.rfc-editor.org/rfc/rfc5321.txt`.

[36] Vixie, P., S. Thomson, Y. Rekhter y J. Bound: *Dynamic Updates in the Domain Name System (DNS UPDATE)*. Rfc 2136, Internet Engineering Task Force, Abril 1997. `http://www.rfc-editor.org/rfc/rfc2136.txt`.

[37] Kempf, J., R. Austein y IAB: *The Rise of the Middle and the Future of End-to-End: Reflections on the Evolution of the Internet Architecture*. Rfc 3724, Internet Engineering Task Force, Marzo 2004. `http://www.rfc-editor.org/rfc/rfc3724.txt`.

[38] Tulloch, Mitch: *Windows 2000 Administration in a Nutshell*. O'Reilly & Associates, Inc., 1005 Gravenstein Highway North, Sebastopol, California, USA, CA 95472, 1ª edición, 2001.

[39] Hunt, Craig: *TCP/IP Network Administration*. O'Reilly Media, Inc., 3ª edición, Abril 2002, ISBN 0596002971.

[40] Brown, Martin A.: *Guide to IP layer network administration with Linux*. `http://linux-ip.net/`.

[41] Microsoft Inc. (editor): *Microsoft Windows NT Resource Kit*, volumen 2. Microsoft Press, Redmond, Washington, USA, 98052-6399, 1995.

[42] Peddemors, Arjan, Henk Eertink y Ignas Niemegeers: *Communication context for adaptive mobile applications*. En *PERCOMW '05: Proceedings of the Third IEEE International Conference on Pervasive Computing and Communications Workshops*, páginas 173–177, Washington, DC, USA, 2005. IEEE Computer Society, ISBN 0-7695-2300-5.

[43] Pawar, Pravin, Bert-Jan van Beijnum, Arjan Peddemors y Aart van Halteren: *Context-aware middleware support for the nomadic mobile services on multi-homed handheld mobile devices*. En *Proceedings of the 12th IEEE Symposium on Computers and Communications, ISCC 2007*, páginas 341–348. IEEE Computer Society, Julio 2007. `http://doc.utwente.nl/64494/`.

[44] Wac, Katarzyna: *Towards QoS-awareness of context-aware mobile applications and services*. En Meersman, Robert, Zahir Tari y Pilar Herrero (editores): *On the Move to Meaningful Internet Systems 2005: OTM Workshops*, volumen 3762 de *Lecture Notes in Computer Science*, páginas 751–760. Springer Berlin / Heidelberg, 2005.

[45] Sun, Jun Zhao, Jaakko Sauvola y Jukka Riekki: *Application of connectivity information for context interpretation and derivation*. En *Proceedings of the 8th International Conf. on Telecom, ConTEL'05*, volumen 1, páginas 303 – 310, Junio 2005, ISBN 953-184-081-4. `http://dx.doi.org/10.1109/CONTEL.2005.185880`.

[46] Landfeldt, B., T. Larsson, Y. Ismailov y A. Seneviratne: *SLM, a framework for session layer mobility management*. En *Proceedings of the Eight International Conference on Computer Communications and Networks*, páginas 452 –456, 1999.

[47] Hsieh, R. y A. Seneviratne: *Performance analysis of mobile IP and SLM*. En *Proceedings of the Ninth IEEE International Conference on Networks*, páginas 492 – 497, Octubre 2001.

[48] Salz, Jon, Alex C. Snoeren y Hari Balakrishnan: *TESLA: a transparent, extensible session-layer architecture for end-to-end network services*. En *Proceedings of the Fourth USENIX Symposium on Internet Technologies and Systems (USITS)*, páginas 211–224, 2003.

[49] *Project migrate*. `http://nms.csail.mit.edu/projects/migrate/`.

[50] Snoeren, Alex C. y Hari Balakrishnan: *An end-to-end approach to host mobility*. En *Proceedings of the 6th annual International Conference on Mobile Computing and Networking*, MobiCom'00, páginas 155–166. ACM, 2000, ISBN 1-58113-197-6. `http://doi.acm.org/10.1145/345910.345938`.

[51] Funato, D., K. Yasuda y H. Tokuda: *TCP-R: TCP mobility support for continuous operation*. En *Proceedings of the 1997 International Conference on Network Protocols (ICNP '97)*, ICNP '97, páginas 229–236, Washington, DC, USA, Octubre 1997. IEEE Computer Society, ISBN 0-8186-8061-X. `http://dl.acm.org/citation.cfm?id=850935.852432`.

[52] Allman, M., V. Paxson y E. Blanton: *TCP Congestion Control*. Rfc 5681, Internet Engineering Task Force, Septiembre 2009. `http://www.rfc-editor.org/rfc/rfc5681.txt`.

[53] *IETF Transport Area Working Group*. `http://datatracker.ietf.org/wg/tsvwg/charter/`.

[54] Nishida, Y., Preethi Natarajan y A. Caro: *Quick Failover Algorithm in SCTP*. Internet-draft draft-nishida-tsvwg-sctp-failover-04, Internet Engineering Task Force, Septiembre 2011. `http://www.ietf.org/internet-drafts/draft-nishida-tsvwg-sctp-failover-04.txt`, Work in progress.

[55] Mathis, M. y J. Heffner: *Packetization Layer Path MTU Discovery*. Rfc 4821, Internet Engineering Task Force, Marzo 2007. `http://www.rfc-editor.org/rfc/rfc4821.txt`.

[56] McCann, J., S. Deering y J. Mogul: *Path MTU Discovery for IP version 6*. Rfc 1981, Internet Engineering Task Force, Agosto 1996. `http://www.rfc-editor.org/rfc/rfc1981.txt`.

[57] Mogul, J.C. y S.E. Deering: *Path MTU discovery*. Rfc 1191, Internet Engineering Task Force, Noviembre 1990. `http://www.rfc-editor.org/rfc/rfc1191.txt`.

[58] Allman, M., V. Paxson y W. Stevens: *TCP Congestion Control*. Rfc 2581, Internet Engineering Task Force, Abril 1999. `http://www.rfc-editor.org/rfc/rfc2581.txt`.

[59] Jacobson, V.: *Congestion avoidance and control*. SIGCOMM Comput. Commun. Rev., 18:314–329, Agosto 1988, ISSN 0146-4833. `http://doi.acm.org/10.1145/52325.52356`.

[60] Paxson, V. y M. Allman: *Computing TCP's Retransmission Timer*. Rfc 2988, Internet Engineering Task Force, Noviembre 2000. `http://www.rfc-editor.org/rfc/rfc2988.txt`.

[61] Karn, P. y C. Partridge: *Improving round-trip time estimates in reliable transport protocols*. SIGCOMM Comput. Commun. Rev., 17:2–7, Agosto 1987, ISSN 0146-4833. `http://doi.acm.org/10.1145/55483.55484`.

[62] Paxson, V., M. Allman, J. Chu y M. Sargent: *Computing TCP's Retransmission Timer*. Rfc 6298, Internet Engineering Task Force, Junio 2011. `http://www.rfc-editor.org/rfc/rfc6298.txt`.

[63] Budzisz, Ł., R. Ferrus, K. J. Grinnemo, A. Brunstrom y F. Casadevall: *An analytical estimation of the failover time in SCTP multihoming scenarios*. En *Wireless Communications and Networking Conference, 2007.WCNC 2007. IEEE*, páginas 3929 –3934, Marzo 2007.

[64] Caro, A.L., Jr., P.D. Amer y R.R. Stewart: *End-to-end failover thresholds for transport layer multihoming*. En *Military Communications Conference, 2004. MILCOM 2004. IEEE*, volumen 1, páginas 99–105, 2004.

[65] Eklund, Johan y Anna Brunstrom: *Impact of SACK delay and link delay on failover performance in SCTP*. En *3rd International Conference on Communications and Computer Networks (IC3N06)*, páginas 69–74, 2006.

[66] Eklund, Johan, Anna. Brunstrom y K. J. Grinnemo: *On the relation between SACK delay and SCTP failover performance for different traffic distributions.* En *Broadband Communications, Networks and Systems, 2008. BROADNETS 2008. 5th International Conference on*, páginas 577–584, Septiembre 2008.

[67] Psaras, Ioannis y Vassilis Tsaoussidis: *The TCP minimum RTO revisited.* En *Proceedings of the 6th international IFIP-TC6 conference on Ad Hoc and sensor networks, wireless networks, next generation internet*, NETWORKING'07, páginas 981–991. Springer-Verlag, 2007, ISBN 978-3-540-72605-0. `http://portal.acm.org/citation.cfm?id=1772322.1772428`.

[68] Kelly, A., Gabriel Miro Muntean, P. Perry y J. Murphy: *Delay-centric handover in SCTP over WLAN.* En *Trans. on Automatic Control & Control Science*, 2004.

[69] Becke, M., T. Dreibholz, J. Iyengar, P. Natarajan y M. Tuexen: *Load Sharing for the Stream Control Transmission Protocol (SCTP).* Internet-draft draft-tuexen-tsvwg-sctp-multipath-02, Internet Engineering Task Force, Julio 2011. `http://www.ietf.org/internet-drafts/draft-tuexen-tsvwg-sctp-multipath-02.txt`, Work in progress.

[70] Natarajan, Preethi, Nasif Ekiz, Paul D. Amer, Janardhan R. Iyengar y Randall R. Stewart: *Concurrent multipath transfer using SCTP multihoming: introducing the potentially-failed destination state.* En *Proceedings of the 7th International IFIP-TC6 Networking Conference on AdHoc and Sensor Networks, Wireless Networks, Next Generation Internet*, NETWORKING'08, páginas 727–734. Springer-Verlag, 2008, ISBN 3-540-79548-0, 978-3-540-79548-3. `http://portal.acm.org/citation.cfm?id=1792514.1792594`.

[71] Cortes-Martin, Alberto, Carlos Garcia-Rubio, Celeste Campo, Andres Marin, Florina Almenarez y Daniel Diaz: *Decoupling path failure detection from congestion control to improve SCTP failovers.* IEEE Communications Letters, 12(11):858–860, Noviembre 2008, ISSN 1089-7798.

[72] Rüngeler, Irene, Michael Tüxen y Erwin P. Rathgeb: *Congestion and flow control in the context of the message-oriented protocol SCTP.* En *Proceedings of the 8th International IFIP-TC 6 Networking Conference*, NETWORKING '09, páginas 468–481. Springer-Verlag, 2009, ISBN 978-3-642-01398-0. `http://dx.doi.org/10.1007/978-3-642-01399-7_37`.

[73] *OMNeT++ Network Simulation Framework.* `http://www.omnetpp.org/`.

[74] *INET Framework.* `http://inet.omnetpp.org/`.

[75] Braden, R.: *Requirements for Internet Hosts - Communication Layers*. Rfc 1122, Internet Engineering Task Force, Octubre 1989. `http://www.rfc-editor.org/rfc/rfc1122.txt`.

[76] Kashihara, Shigeru, Katsuyoshi Iida, Hiroyuki Koga, Youki Kadobayashi y Suguru Yamaguchi: *Multi-path transmission algorithm for end-to-end seamless handover across heterogeneous wireless access networks.* En *IWDC'03*, páginas 174–183, 2003.

[77] Cano, Maria Dolores: *Improving path failure detection in SCTP using adaptive heartbeat time intervals*. SIGMETRICS Perform. Eval. Rev., 39:50–52, Septiembre 2011, ISSN 0163-5999. `http://doi.acm.org/10.1145/2034832.2034845`.

[78] Nagle, J.: *Congestion Control in IP/TCP Internetworks*. Rfc 0896, Internet Engineering Task Force, Enero 1984. `http://www.rfc-editor.org/rfc/rfc896.txt`.

[79] Floyd, S.: *Congestion Control Principles*. Rfc 2914, Internet Engineering Task Force, Septiembre 2000. `http://www.rfc-editor.org/rfc/rfc2914.txt`.

[80] Tsao, Shih Ching, Yuan Cheng Lai y Ying Dar Lin: *Taxonomy and evaluation of TCP-friendly congestion-control schemes on fairness, aggressiveness, and responsiveness*. IEEE Network, 21(6):6–15, 2007.

[81] Raiciu, C., M. Handley y D. Wischik: *Coupled Congestion Control for Multipath Transport Protocols*. Rfc 6356, Internet Engineering Task Force, Octubre 2011. `http://www.rfc-editor.org/rfc/rfc6356.txt`.

[82] Caro, A. L., Keyur Shah, Janardhan R. Iyengar, Paul D. Amer y Randall R. Stewart: *SCTP and TCP variants: Congestion control under multiple losses*. Informe técnico TR2003-04, CIS Dept., Uni. of Delaware, Febrero 2003.

[83] Allman, M., S. Floyd y C. Partridge: *Increasing TCP's Initial Window*. Rfc 3390, Internet Engineering Task Force, Octubre 2002. `http://www.rfc-editor.org/rfc/rfc3390.txt`.

[84] Chu, J., N. Dukkipati, Y. YuchungCheng y M. Mathis: *Increasing TCP's Initial Window*. Internet-draft draft-ietf-tcpm-initcwnd-02, Internet Engineering Task Force, Octubre 2011. `http://www.ietf.org/internet-drafts/draft-ietf-tcpm-initcwnd-02.txt`, Work in progress.

[85] Ross, Sheldon M.: *Random variables: Proposition 4.1.* En *Introduction to Probability Models*, capítulo 2. Academic Press, New York, 2ª edición, 1980.

[86] Bracewell, R.: *Rectangle function of unit height and base.* En *The Fourier Transform and Its Applications*, páginas 52–53. McGraw-Hill, New York, 1965.

[87] Weiser, Mark: *The computer for the 21st century.* SIGMOBILE Mob. Comput. Commun. Rev., 3:3–11, Julio 1999, ISSN 1559-1662. `http://doi.acm.org/10.1145/329124.329126`.

[88] Berkowitz, H.: *To Be Multihomed: Requirements & Definitions.* Internet-draft draft-berkowitz-multirqmt-01, Internet Engineering Task Force, Marzo 1998. `http://tools.ietf.org/html/draft-berkowitz-multirqmt-01`, Work in progress.

[89] Abley, J., K. Lindqvist, E. Davies, B. Black y V. Gill: *IPv4 Multihoming Practices and Limitations.* Rfc 4116, Internet Engineering Task Force, Julio 2005. `http://www.rfc-editor.org/rfc/rfc4116.txt`.

[90] Cortes-Martin, Alberto, Carlos Garcia-Rubio, Celeste Campo, Estrella M. Garcia-Lozano y Alicia Rodriguez-Carrion: *Selection and publication of network interface cards in multihomed pervasive computing devices.* En *2011 IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*, páginas 239–244, Seattle, WA, USA, Marzo 2011. IEEE Computer Society, ISBN 978-1-61284-936-9.

[91] Lear, E.: *Things Multihoming in IPv6 (MULTI6) Developers Should Think About.* Rfc 4219, Internet Engineering Task Force, Octubre 2005. `http://www.rfc-editor.org/rfc/rfc4219.txt`.

[92] Krishnan, S., N. Montavont, E. Njedjou, S. Veerepalli y A. Yegin: *Link-Layer Event Notifications for Detecting Network Attachments.* Rfc 4957, Internet Engineering Task Force, Agosto 2007. `http://www.rfc-editor.org/rfc/rfc4957.txt`.

[93] Puttonen, Jani, Gabor Fèkete, Tapio Väärämäki y Timo Hämäläinen: *Multiple interface management of multihomed mobile hosts in heterogeneous wireless environments.* En *Proceedings of the 2009 Eighth International Conference on Networks*, páginas 324–331, Washington, DC, USA, 2009. IEEE Computer Society, ISBN 978-0-7695-3552-4. `http://dl.acm.org/citation.cfm?id=1547552.1547817`.

[94] Sutinen, T. y H. Rivas: *Cross-layer assisted network interface selection for multi-interface video streaming.* En *Computer Communications and Networks (ICCCN), 2011 Proceedings of 20th International Conference on*, páginas 1 –6, Agosto 2011.

[95] Makela, J. y K. Pentikousis: *Trigger management mechanisms.* En *Wireless Pervasive Computing, 2007. ISWPC '07. 2nd International Symposium on*, Febrero 2007.

[96] Makela, J., R. Aguero, J. Tenhunen, V. Kyllonen, J. Choque y L. Munoz: *Paving the way for future mobility mechanisms: a testbed for triggering and moving network support*. En *2nd International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities, 2006. TRIDENTCOM 2006.*, página 424, 2006.

[97] Puente, M., J. Sainz, M. Garcia, J. Prokkola, P.Ruuska y A. Fernandez: *Qos methods*. Informe técnico, Easy wireless 2 proyect, 2009.

[98] Sainz, J., F. Abril, K. Oinonen, E. Bunn, J. Prokkola, P. Ruuska y J. Ikäheimo: *Measurement, monitoring and analysis*. Informe técnico, Easy Wireless 2 proyect, 2009.

[99] Almes, G., S. Kalidindi y M. Zekauskas: *A One-way Delay Metric for IPPM*. Rfc 2679, Internet Engineering Task Force, Septiembre 1999. `http://www.rfc-editor.org/rfc/rfc2679.txt`.

[100] Demichelis, C. y P. Chimento: *IP Packet Delay Variation Metric for IP Performance Metrics (IPPM)*. Rfc 3393, Internet Engineering Task Force, Noviembre 2002. `http://www.rfc-editor.org/rfc/rfc3393.txt`.

[101] Almes, G., S. Kalidindi y M. Zekauskas: *A One-way Packet Loss Metric for IPPM*. Rfc 2680, Internet Engineering Task Force, Septiembre 1999. `http://www.rfc-editor.org/rfc/rfc2680.txt`.

[102] Koodli, R. y R. Ravikanth: *One-way Loss Pattern Sample Metrics*. Rfc 3357, Internet Engineering Task Force, Agosto 2002. `http://www.rfc-editor.org/rfc/rfc3357.txt`.

[103] Palet, J., M. Diaz, C. Olvera, A. Vives, E. Fleischman y D. Lanciani: *Analysis of IPv6 Multihoming Scenarios*. Internet-draft draft-palet-multi6-scenarios-00, Internet Engineering Task Force, Julio 2004. `http://www.ietf.org/internet-drafts/draft-palet-multi6-scenarios-00.txt`, Work in progress.

[104] Mitsuya, K., K. Tasaka, R. Wakikawa y R. Kuntz: *A Policy Data Set for Flow Distribution*. Internet-draft draft-mitsuya-monami6-flow-distribution-policy-04, Internet Engineering Task Force, Agosto 2007. `http://www.ietf.org/internet-drafts/draft-mitsuya-monami6-flow-distribution-policy-04.txt`, Work in progress.

[105] Larsson, C., M. Eriksson, K. Mitsuya, K. Tasaka y R. Kuntz: *Flow Distribution Rule Language for Multi-Access Nodes*. Internet-draft draft-larsson-mext-flow-distribution-rules-02, Internet Engineering Task Force, Febrero 2009. `http://www.ietf.org/internet-drafts/draft-larsson-mext-flow-distribution-rules-02.txt`, Work in progress.

[106] Mochel, Patrick: *The sysfs filesystem.* En *Proceedings of the Annual Linux Sumposium*, volumen 1, páginas 203 – 207, Julio 2005. `http://www.linuxsymposium.org/archives/OLS/Reprints-2005/mochel-Reprint.pdf`.

[107] Killian, T. J.: *Processes as files.* En *Proceedings of the USENIX Summer 84 Conference.* USENIX Association, 1984.

[108] Corbet, Jonathan: *Network namespaces.* `http://lwn.net/Articles/219794/`.

[109] Vermeulen, Frederik *y cols.*: *Demostrator specification.* Informe técnico, Easy wireless proyect, 2008.

[110] Sandvik, Erling *y cols.*: *Final demostration.* Informe técnico, Easy wireless proyect, 2008.

[111] Cortés-Martín, Alberto *y cols.*: *Enhancing performance and control in heterogeneous networks.* Informe técnico, Easy wireless 2 proyect, 2010.

[112] *MARCH, Multilink architecture for multiple services, Workshop, 21 Sep. 2011, Oslo, Noruega.* `http://projects.celtic-initiative.org/march/march/news.php?id=69`.

[113] *MONOLOC, localización y gestión móvil.* `http://mobilenet.ic.uma.es/research-projects/monoloc-2011`.

[114] Postel, J.: *Internet Protocol.* Rfc 0791, Internet Engineering Task Force, Septiembre 1981. `http://www.rfc-editor.org/rfc/rfc791.txt`.

[115] Deering, S. y R. Hinden: *Internet Protocol, Version 6 (IPv6) Specification.* Rfc 2460, Internet Engineering Task Force, Diciembre 1998. `http://www.rfc-editor.org/rfc/rfc2460.txt`.

[116] Postel, J.: *Transmission Control Protocol.* Rfc 0793, Internet Engineering Task Force, Septiembre 1981. `http://www.rfc-editor.org/rfc/rfc793.txt`.