Trabajo Fin de Máster

Implementación y evaluación de soluciones de anonimato en redes oportunistas

Autor

Miguel García Giménez

Director

Guillermo Navarro Arrivas

MÁSTER INTERUNIVERSITARIO EN SEGURIDAD DE LAS TIC
2019

Implementación y evaluación de soluciones de anonimato en redes oportunistas

RESUMEN

Actualmente, y desde hace ya algunos años, las redes están evolucionando hacia un paradigma distribuido. La implantación de nuevos servicios que pueden funcionar sin una arquitectura centralizada se ha extendido considerablemente como un medio para la obtención y el procesamiento de nueva información. Gracias a esto, servicios especiales de emergencias o el control de especies en extinción son posibles sin el despliegue de la correspondiente infraestructura física centralizada.

No obstante, este nuevo paradigma también conlleva una serie de riesgos que son necesarios analizar. La capacidad de los dispositivos que normalmente se utilizan en este tipo de redes es baja respecto a las infraestructuras físicas, por lo que aspectos como la privacidad o el anonimato no llegan a implantarse en pro de la funcionalidad. Además, las comunicaciones que suceden en estas redes tienden a ser oportunistas, lo que plantea un mayor reto debido a su carácter aleatorio.

Con este trabajo fin de máster se plantea una implementación de tecnologías *MIXNET* y *Onion Routing* con las que ofrecer privacidad, autenticidad y anonimato en entornos oportunistas. El fin último es analizar estos entornos y sus capacidades, las cuales determinaran la viabilidad de estas tecnologías en redes oportunistas como solución ante los problemas descritos.

Implementation and evaluation of anonymity solutions for Opportunistic Networks

ABSTRACT

In recent years the technology have been evolving to a distributed computing paradigm. It means that there have also been created new services which can work without a centralized architecture, acting as a new channel to obtain and process new information. Thanks to this, some special emergency services or the control of endangered species are possible without the need for a phisical and centralize architecture.

Nevertheless, this new paradigm has also some risks which are necessary to analize. For example, the computational capacity of the devices that are commonly used in this kind of networks is lower than the devices used in physical infrastructures. So, some technology aspects such as privacity or anonymity are not implemented in the cause of functionality. Furthermore, some of the communications given in this kind of networks tend to be opportunistic, which raise biggest challenges due to its random nature.

This Master Thesis proposes the implementation of *MIXNET* and *Onion Routing* technologies to improve the privacy, authenticity and anonimity in opportunistics networks. The main objetive is the analysis of the implemented environments and its capacities to determine if these technologies are a viable solution to the privacity concerns described above.

Índice

1. Introducción **1**

1.1. Motivación 2
1.2. Objetivos 2
1.3. Metodología 2
1.4. Fases y Tiempos de Realización 3
1.5. Estructura del Documento 4

2. Estado del arte **5**

2.1. Introducción 5
2.2. Definición de redes oportunistas 5
2.3. Análisis de los aspectos críticos 6
2.4. Soluciones propuestas 6
2.4.1. Redes Onion 6
2.4.2. Redes MIX 7
2.4.3. Nodos Epidemic Router 8
2.5. Herramienta de simulación 8

3. Desarrollo e implementación de redes oportunistas **11**

3.1. Introducción 11
3.2. Desarrollo de Onion Routing 11
3.2.1. Cifrado de mensajes 12
3.2.2. Reenvío de mensajes 12
3.2.3. Gestión de buffer 13
3.2.4. Gestión de las conexiones 13
3.3. Desarrollo de MIXNET 14
3.3.1. Cifrado de mensajes 14
3.3.2. Reenvío de mensajes 14
3.3.3. Gestión de buffer 15
3.3.4. Gestión de las conexiones 16
3.4. Desarrollos complementarios 16

3.4.1. Mensajes . . . 16
3.4.2. Creación de reportes . . . 17
3.4.3. Scripting para volcado de información . . . 17

4. Configuración y ejecución de las simulaciones 19
4.1. Introducción . . . 19
4.2. Parámetros de simulación . . . 19
4.2.1. Parámetros de escenario . . . 21
4.2.2. Parámetros de interfaz . . . 21
4.2.3. Parámetros de grupo . . . 21
4.2.4. Parámetros de evento . . . 21
4.3. Escenarios y trazas reales . . . 22
4.4. Configuración de los parámetros . . . 23
4.5. Ejecución de simulaciones . . . 25

5. Resultados obtenidos 27
5.1. Introducción . . . 27
5.2. Parámetros evaluados . . . 27
5.3. Resultados MIXNET . . . 28
5.4. Resultados Onion Routing . . . 32

6. Conclusiones y líneas futuras 35

7. Bibliografía 37

Lista de Figuras 39

Anexos 40

A. Resultados de las simulaciones 43
A.1. Resultados para red MIX . . . 43
A.2. Resultados para red Onion . . . 56

Capítulo 1

Introducción

La evolución de la tecnología está permitiendo el desarrollo y la implantación de nuevos paradigmas con los que ofrecer servicios que mejoran la calidad de la información procesada hasta el momento. Estos paradigmas se basan en la descentralización de las redes, dejando de lado las estructuras jerárquicas conocidas para dar paso a modelos en los que no existe un único punto de control en la red.

El carácter distribuido de estas nuevas redes propicia que su implantación se lleve a cabo en entornos móviles. La no necesidad de una infraestructura jerárquica junto con el carácter móvil de los nodos permite despliegues en los que son los propios nodos los que se comunican cuando estos entran en contacto. Es decir, nodos que participan en redes oportunistas.

Este tipo de redes se denomina redes oportunistas. En estos despliegues, los nodos se comunican solamente con aquellos nodos dentro de su rango inalámbrico. De esta forma, un mensaje se va transmitiendo desde el nodo origen hasta el nodo final pasando por un número indeterminado de nodos intermedios. La particularidad de estas redes es que se desconoce el instante en el que dos nodos van a entrar en contacto, es decir, se basan en la oportunidad de establecer contacto entre pares de nodos para propagar mensajes.

Las redes oportunistas se están extendiendo ampliamente ya que ofrecen soluciones para entornos convencionales saturados o entornos muy limitados, como por ejemplo centros comerciales, rescate de alta montaña o el seguimiento de especies en extinción. No obstante, como en todas las nuevas tecnologías, y en las ya consolidadas, existen problemas que son necesarios solucionar si se quiere trasladar el uso de estas redes a otros muchos servicios.

1.1. Motivación

La idea que subyace en las redes oportunistas sobre la falta de infraestructura también se traslada a la capacidad de los nodos. En algunos entornos, y en ocasiones por problemas de costes, es imposible desplegar dispositivos de alta capacidad, optando por cumplir solamente con la funcionalidad del servicio sin poder atender aspectos tan importantes como la privacidad o el anonimato.

Para que las redes oportunistas sigan evolucionando y se trasladen a servicios de uso más común, es necesario solucionar los problemas de seguridad planteados. Solamente de esta forma, las redes oportunistas estarán preparadas para ser utilizadas a diario por millones de personas en todo el mundo.

Por ello, este trabajo se centra en la implantación y el análisis de soluciones *MIXNET* y *Onion Routing* en redes oportunistas. El uso de estas dos tecnologías es idóneo debido a su carácter distribuido. Ambas utilizan nodos intermedios para proporcionar privacidad y anonimato, sin tener que configurar ningún tipo de infraestructura física convencional. El análisis de ambas se llevará a cabo con el software The One, un simulador oportunista introducido en la sección 2.5 de este trabajo.

1.2. Objetivos

El objetivo principal de este trabajo es ofrecer un análisis sobre el impacto de *MIXNET* y *Onion Routing* en redes oportunistas como soluciones ante los problemas de seguridad planteados.

Para conseguir dicho objetivo principal, se han definido los siguientes objetivos específicos:

- Diseñar e implementar una red *MIXNET* con el fin de simular su comportamiento con el sofware The One.
- Diseñar e implementar una red *Onion* con el fin de simular su comportamiento con el sofware The One.
- Definir un escenario real oportunista y evaluar el impacto de las redes de privacidad previamente diseñadas

1.3. Metodología

Para asegurar la correcta realización de los objetivos propuestos se pretende seguir una metodología de fases.

La primera fase se enfoca hacia el marco teórico del trabajo. Esto es, la definición teórica de las redes *MIXNET* y *Onion*, su desarrollo y su implementación en el entorno de simulación The One. También se pretende incluir en esta fase el diseño del escenario a simular.

La segunda fase está formada por las distintas simulaciones de las redes ya implementadas, así como la evaluación de los resultados y las conclusiones oportunas. Es decir, una fase práctica que se apoya en la solución teórica de la primera fase.

1.4. Fases y Tiempos de Realización

En la Figura 1.1 se expone un diagrama de Gantt que recoge las fases de realización a lo largo de todo el desarrollo del trabajo fin de máster.

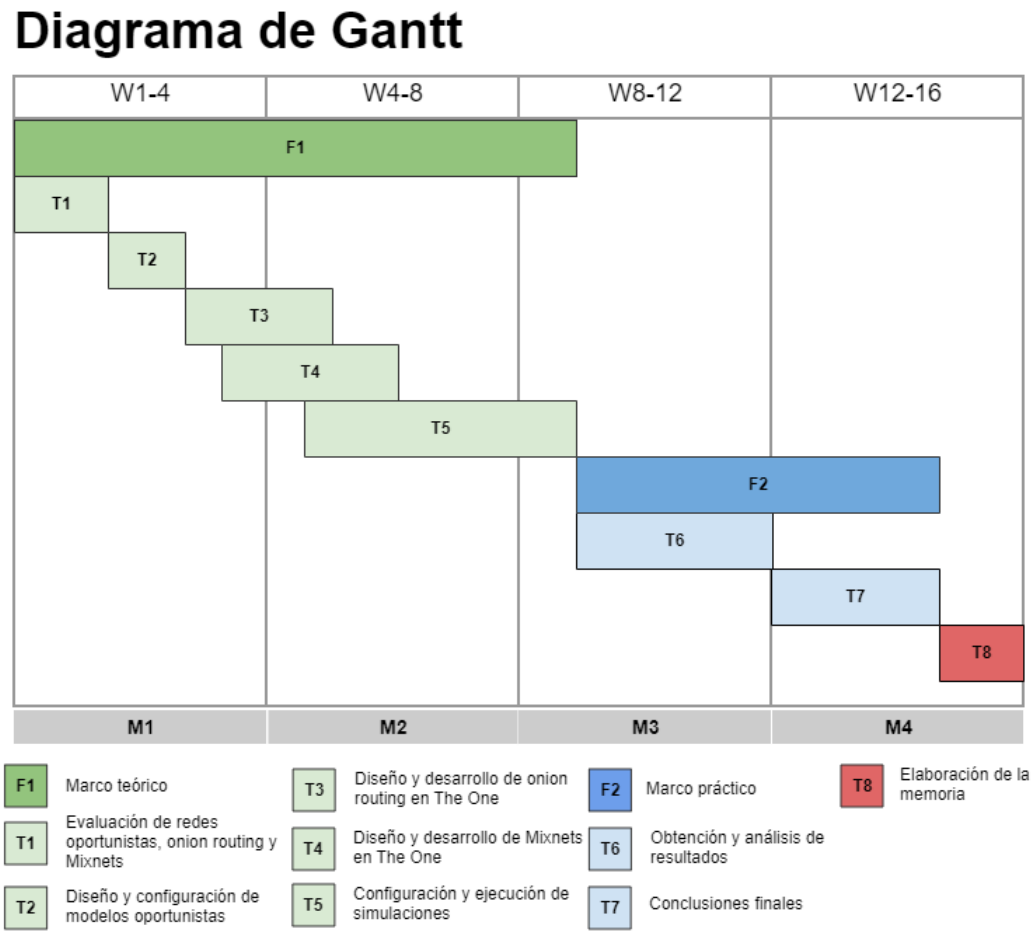

Figura 1.1: Diagrama temporal

1.5. Estructura del Documento

La estructura que da forma a este proyecto está dividida en cinco capítulos en los que se explica cada una de las fases comentadas con anterioridad. También se incluye una lista con las figuras utilizadas, la bibliografía que se ha seguido a lo largo del trabajo y los anexos con el resultado final.

El primer capítulo está dedicado al estado del arte. Se pretende ofrecer así una visión un poco más específica sobre las redes oportunistas, sus problemas de seguridad y como algunas tecnologías, como lo son *MIXNET* y *Onion Routing*, pueden ayudar a solucionarlos.

El segundo capítulo se centra en el desarrollo de estas dos tecnologías propuestas para su funcionamiento en redes oportunistas. Para esto, el desarrollo se ha tenido que integrar con el comportamiento por defecto del software The One.

El tercer capítulo detalla las configuraciones utilizadas en las distintas simulaciones llevadas a cabo. Para comprender correctamente los resultados obtenidos, se ofrece una descripción tanto de los distintos escenarios propuestos como de los datos utilizados en las simulaciones.

El cuarto capítulo se centra en el análisis de los resultados obtenidos en las simulaciones. Para esto, se ha evaluado el impacto de varios parámetros de configuración en cada una de las redes y como su modificación altera el rendimiento de las mismas.

Por último, el quinto capítulo expone tanto las conclusiones una vez analizados los sistemas como algunas líneas futuras respecto a las cuestiones planteadas para redes oportunistas.

Capítulo 2

Estado del arte

2.1. Introducción

En este capítulo se pretende ofrecer una visión más profunda sobre las redes oportunistas (*OppNets* o *Opportunistic Networks*) . Para esto, se incluye una definición general y un análisis de los problemas que este tipo de redes presentan, así como las soluciones elegidas en este proyecto para su solución. Finalmente, se expone la herramienta de simulación elegida para la evaluación de las redes oportunistas.

2.2. Definición de redes oportunistas

Las redes oportunistas son redes en las que no se garantiza la recepción de mensajes por parte de los nodos destino debido al carácter oportunista de las conexiones entre estos. En este tipo de redes no existe una infraestructura centralizada que controle el envío de mensajes, motivo principal por el cual las conexiones que suceden entre estos nodos pueden o no llevar el mensaje hasta su destinatario.

Esta incertidumbre sobre la fiabilidad en el envío de mensajes es lo que introduce complejidad a la hora de desplegar servicios en *OppNets*. La correcta recepción de los mensajes depende del número de conexiones entre nodos y de su duración, lo que directamente está determinado por el movimiento de los mismos. Este movimiento, aunque se puede basar en patrones, no deja de ser probabilístico, por lo que es necesario estudiar los patrones de movimiento de los nodos en redes oportunistas para aumentar la tasa de éxito.

Este tipo de redes, al depender directamente del movimiento de sus nodos, tienen que contar con mecanismos que garanticen un almacenamiento prolongado de los mensajes en los nodos hasta que sea posible el reenvío. Es por esto por lo que en ocasiones se las denomina *DTN* o *Delay-tolerant networks*. Es decir, son redes preparadas para soportar retardos elevados en contraposición a las redes actuales

TCP/IP desplegadas en Internet.

2.3. Análisis de los aspectos críticos

Tal y como ya se ha comentado, las *OppNets* suelen estar desplegadas en entornos desfavorables debido a su descentralización y gestión de retardos. Además de esto, al constituirse redes relativamente grandes o redes con pequeños dispositivos que faciliten la implantación, no suele ser posible incluir nodos de alta capacidad que ofrezcan una gran tasa de cómputo o de almacenamiento.

Estos problemas de capacidad provocan una falta de mecanismos en pro de aquellos que se utilizan para el envío y recepción de mensajes. Es decir, actualmente muchas de las redes oportunistas desplegadas no cuentan con capacidad para ofrecer lo que se espera de los servicios centralizados en redes TCP/IP comunes. Parte de estos mecanismos que faltan, y probablemente los que mas se hacer de notar, son los destinados a la seguridad en las comunicaciones.

Soluciones actuales que proporcionan privacidad, autenticidad o integridad, como por ejemplo el uso de *Public Key Infrastructure* y su aplicación en TLS/SSL, no son posibles llevarlas a entornos oportunistas. Esta problemática hace que los servicios en este tipo de redes no sean seguros y no lleguen a desplegarse, o se haga a favor del propio servicio obtenido comprometiendo la seguridad de los datos.

Así pues, es necesario revisar qué soluciones son posibles implantar en redes oportunistas para garantizar la seguridad en las comunicaciones. Además, y acorde a lo que en este trabajo se expone, es necesario evaluar el impacto de las nuevas soluciones de seguridad para que el nuevo paradigma que plantean estas redes no se vea limitado.

2.4. Soluciones propuestas

En este trabajo se plantean las tecnologías *MIXNET* y *Onion Routing* como soluciones ante la falta de seguridad en las comunicaciones de *OppNets*. Ambas aprovechan la descentralización de este tipo de redes para ofrecer privacidad y anonimato. Como usuarios de estas tecnologías se han configurado nodos Epidemic Router, también introducidos en la sección 2.4.3.

2.4.1. Redes Onion

Las redes de tipo *Onion* se basan en cifrar un mensaje tantas veces como nodos *Onion* existan en el path que va a seguir el mensaje. Este path es predefinido y los mensajes no pueden viajar a otro receptor que no sea el siguiente nodo *Onion* en el

path. Cada nodo *Onion* descifra su capa de cifrado y envía el mensaje al siguiente nodo del path. Finalmente, el mensaje se descifra en el nodo *Onion* frontera para enviarlo descifrado al nodo final.

En este tipo de redes se distinguen dos tipos de nodos, los destinados a ofrecer funciones de desencriptación (nodos intermedios) y los nodos emisores y/o receptores. La Figura 2.1 muestra un envío de mensaje dentro de una red *Onion*.

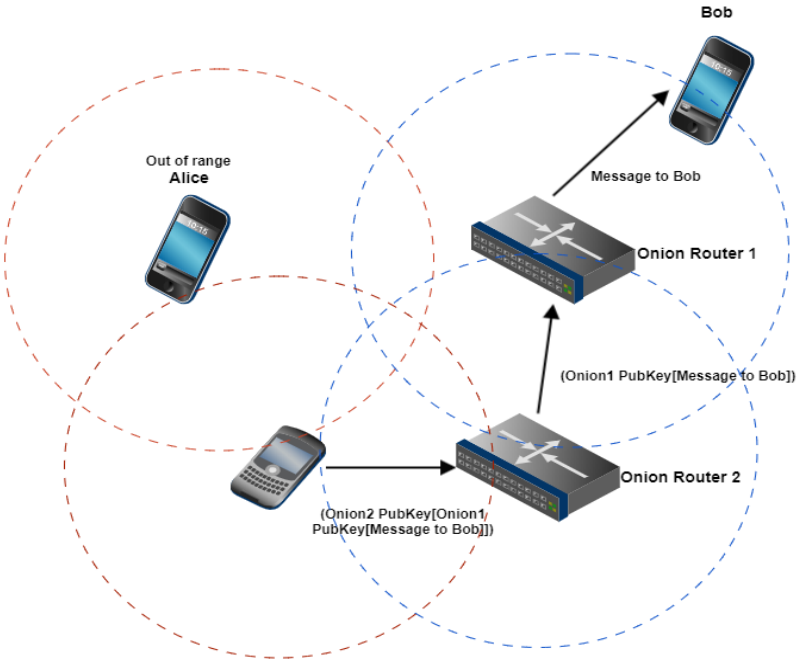

Figura 2.1: Envío de mensaje en red Onion.

2.4.2. Redes MIX

Las redes MIX o *MIXNET* son protocolos de enrutamiento que utilizan nodos *proxy* para ofrecer una capa de anonimato extra al cifrado convencional. Aunque estas redes siguen utilizando criptografía de clave pública tipo *Onion*, los *proxies* descifran la correspondiente capa de cifrado con su clave privada y reordenan los mensajes antes de reenviarlos con el fin de romper el enlace entre emisor y destino.

Otra particularidad de este tipo de redes es que, al contrario que en el protocolo *Onion routing* definido anteriormente, no siguen un path predefinido en el envío del mensaje. Los mensajes de reenviarán a través de toda la red hasta que lleguen al correspondiente nodo MIX.

Esta tecnología ofrece privacidad gracias a los cifrados de clave pública utilizados. Además, ofrece anonimato al reordenar los mensajes, dificultando las posibles escuchas que un tercero pudiera llevar a cabo.

Así pues, dentro de las redes MIX se distinguen dos tipos de nodos, los que ofrecen funciones de reordenación de mensajes y los nodos emisores y/o receptores. La Figura 2.2 muestra un envío de mensaje dentro de una MIXNET.

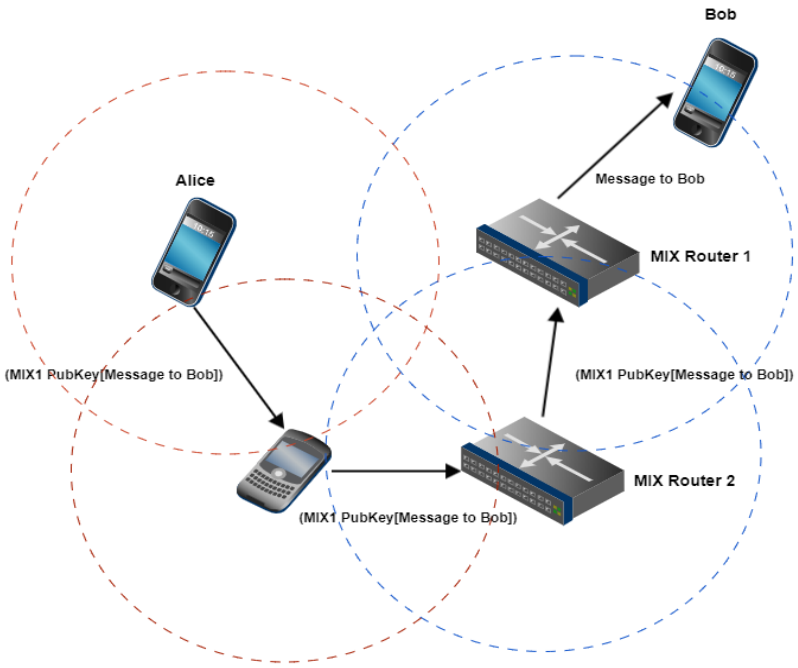

Figura 2.2: Envío de mensaje en red MIX.

2.4.3. Nodos Epidemic Router

Las simulaciones llevadas a cabo pueden contener tres nodos diferentes: MIX, *Onion* o Epidemic Router. Estos últimos son los que generan los mensajes hacia otros Epidemic Router haciendo uso de las redes *Onion* o MIX desplegadas.

Acorde a la definición del protocolo, un nodo Epidemic Router envía cada mensaje a todos los nodos con los que está conectado en un intento de aumentar la probabilidad de que este llegue al nodo final hacia el que va dirigido. The One implementa este protocolo añadiendo la restricción de que un nodo Epidemic Router solamente puede enviar un mensaje a la vez, dejando de lado el envío a la vez a múltiples nodos. Además, cuando un nodo Epidemic Router llena su buffer, libera espacio en base al borrado de los mensajes más antiguos que aún conserva.

2.5. Herramienta de simulación

La herramienta escogida para la simulación de los distintos escenarios es The One. Esta herramienta es un simulador para redes oportunistas capaz de simular el movimiento basado en patrones de los nodos, así como de generar los mensajes

aleatorios y visualizar el intercambio de los mismos en tiempo real a través de una interfaz Java incorporada.

Una característica muy importante de esta herramienta de simulación, y que afecta directamente al desarrollo llevado a cabo, es que las simulaciones se realizan a nivel de routing. Es decir, las *MIXNET* y *Onion Routing* desarrolladas se centran en el encaminamiento de paquetes, dejando de lado capas superiores como puede ser la de aplicación. Aunque estas capas están directamente ligadas a los protocolos descritos mediante aplicaciones como el cifrado TLS/SSL, esta característica solo se va a poder simular, por ejemplo, con el tamaño de paquete.

El desarrollo de The One ya cuenta con distintos tipos de nodos. No obstante, en este trabajo solamente se utilizarán nodos *epidemic router* como emisores y receptores de los mensajes. Además, se utilizarán trazas reales de conexiones oportunistas para dotar de más realismo a los resultados obtenidos. En la Figura 2.3 se ofrece una captura de dicho software.

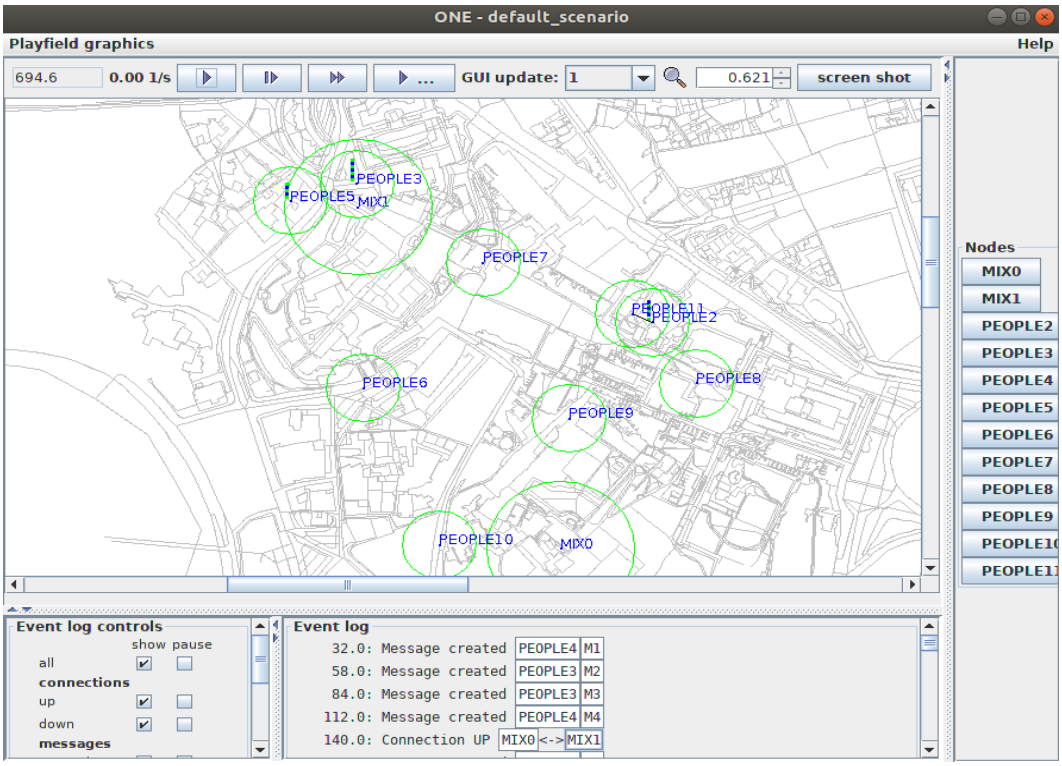

Figura 2.3: Captura del simulador The One.

Capítulo 3

Desarrollo e implementación de redes oportunistas

3.1. Introducción

En este apartado se pretende profundizar sobre el diseño, desarrollo e implementación de las tecnologías *MIXNET* y *Onion Routing* propuestas como solución para aumentar la seguridad en redes oportunistas.

Todos los desarrollos se han realizado en Java ya que es el lenguaje que utiliza el simulador The One. Además, para que las redes funcionen correctamente, se han tenido que diseñar mensajes y reportes específicos a parte de los nodos correspondientes.

3.2. Desarrollo de Onion Routing

El tráfico en las redes oportunistas evaluadas con nodos *Onion* está cifrado con tantas capas como nodos participan en una comunicación. Es decir, si un nodo emisor envía un mensaje a un nodo receptor cuyo path establecido consta de tres nodos *Onion*, el mensaje original estará tres veces cifrado con las correspondientes claves públicas.

Estos nodos que participan en redes oportunistas (o en redes distribuidas) no tienen a su alcance una infraestructura de intercambio de claves como pueden ser las actuales *PKI*. La falta de una infraestructura de este tipo implica que los protocolos deben de adaptarse para intentar obtener de forma dinámica una clave pública. De esta forma, cuando un nodo *Onion* se añadiese a la red no sería necesario ningún mantenimiento en el resto de los nodos.

Aunque el intercambio de estas claves debería de formar parte del protocolo, su implementación queda fuera del alcance de este proyecto por ser una linea actual de investigación. Se asume que los nodos emisores conocen todas las claves públicas de los nodos *Onion* desplegados.

3.2.1. Cifrado de mensajes

Las redes de tipo *Onion* son redes en las que los mensajes se encapsulan bajo capas de cifrado, tantas como nodos vayan a participar en la comunicación. Estas capas de cifrado utilizan claves de cifrado asimétrico (clave pública y privada) para garantizar la privacidad del mensaje. Este comportamiento no tiene un desarrollo específico ya que la simulación llega hasta el nivel de routing, no a capas superiores de aplicación.

3.2.2. Reenvío de mensajes

El reenvío de mensajes solamente se produce si el siguiente nodo que va a recibir el mensaje es también el siguiente en el *path* preestablecido para el mensaje. Esto es debido a que en en estas redes se debe de establecer un circuito entre nodos intermedios de tipo *Onion* que reenvíen el mensaje hasta el nodo final.

Dicho circuito contiene todos los nodos *Onion* desplegados en la red y es precomputado de forma previa al envío del mensaje acorde a un algoritmo de aleatorización. De esta manera, se simula la creación de un circuito aleatorio en cada nodo emisor (ver Figura 3.1).

Los nodos *Onion* que forman los circuitos se especifican en la configuración de The One. Cada simulación, tal y como se indica en la sección 4.3, se configura con 304 nodos, de los cuales algunos de ellos se utilizan como nodos *Onion*. El número total de nodos es constante, por lo que cuanto mayor es el número de nodos *Onion*, menor es el número de nodos Epidemic Router (emisores). Es decir, los nodos *Onion* se configuran de forma previa para que actúen acorde a dicho protocolo.

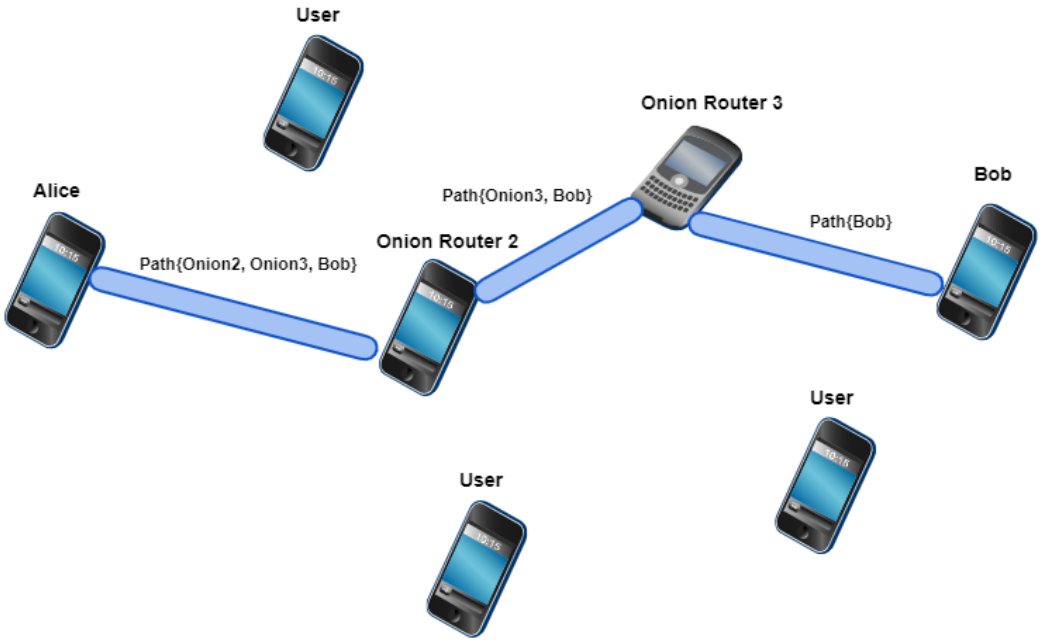

Figura 3.1: Path preestablecido en red Onion.

3.2.3. Gestión de buffer

Tras un envío exitoso, un nodo *Onion* elimina el mensaje del buffer local para una correcta gestión de la memoria. Además, se prioriza la admisión de nuevos paquetes en el buffer aún cuando no hay espacio disponible mediante el borrado de mensajes antiguos (comportamiento heredado de *Epidemic Router*). No admite un nuevo mensaje en buffer si este ya se encuentra en él.

3.2.4. Gestión de las conexiones

La tarea que desempeñan los nodos *Onion* dentro de las redes necesita de la gestión de múltiples conexiones de forma simultánea. Esto es, la capacidad de gestionar múltiples conexiones y enviar datos por todas ellas a la vez.

De esta manera, un nodo es capaz de aumentar su ancho de banda y no ser un cuello de botella durante el reenvío de mensajes.

La tabla 3.1 resume las características de las redes Onion comentadas hasta el momento.

Característica	Configuración
Cifrado de mensajes	Simulado mediante parámetros del mensaje.
Reenvío de mensajes	Reenvío condicionado a comunicación con siguiente nodo del *path*.
Gestión de buffer	Eliminación de mensajes correctamente mandados y prioridad a nuevos mensajes recibidos. No se pueden duplicar mensajes en el buffer.
Gestión de las conexiones	Múltiples conexiones simultáneas.

Tabla 3.1: Características implementadas en nodos Onion.

3.3. Desarrollo de MIXNET

Las redes *MIXNET* hacen uso de tecnología *Onion Routing* para ofrecer servicios de anonimato y privacidad. Además, reordenan los mensajes en los nodos MIX para desenlazar las conexiones entre emisor y receptor ante escuchas de terceros.

3.3.1. Cifrado de mensajes

Tal y como ya se ha comentado, el cifrado sigue el esquema marcado por las redes *Onion*. Así pues, las redes MIX desarrolladas tampoco se han configurado a nivel de aplicación, dejando este comportamiento a la simulación mediante parámetros como el tamaño de mensaje.

3.3.2. Reenvío de mensajes

Además de los cifrados tipo *Onion*, las redes MIX se caracterizan por ofrecer una reordenación de mensajes en sus nodos MIX con el fin de no reenviarlos instantáneamente tras su recepción. Los nodos MIX desarrollados tienen un atributo configurado que actúa como *threshold* de cara al reenvío de mensajes.

Cuando un nodo MIX comienza a almacenar mensajes en su buffer local, no comienza con el reenvío hasta que el número de mensajes alcanza el *threshold*. Este atributo es configurable desde un fichero de configuración externo.

Si el número de mensajes alcanza el límite marcado para el reenvío, un nodo MIX obtiene un número *threshold* de mensajes de su buffer de forma aleatoria y los reordena de la misma forma. Tras esto, comienza con el reenvío hacia nodos finales con los que esté actualmente conectado. De esta manera, se priorizan los mensajes que están en su tramo final del path con el fin de aumentar la tasa de mensajes entregados.

Una característica importante de las redes MIX es que los mensajes pueden viajar por toda la red con la obligación de que en algún momento pasen por ciertos nodos MIX. Así pues, tras el reenvío a nodos finales conectados, se eligen de forma aleatoria y secuencial las conexiones actuales para el reenvío del resto de mensajes. Esto es, el reenvío a cualquier nodo conectado de cualquier mensaje seleccionado para que este se propague y llegue en algún momento a su destino (ver Figura 3.2).

De la misma manera que para *Onion Routing*, los nodos Epidemic Router que utilicen una red MIX deben de precomputar un path de envío de mensajes que haga uso de todos los nodos *MIX* desplegados. Estos nodos deben de configurarse de forma previa a la simulación para que actúen como tal.

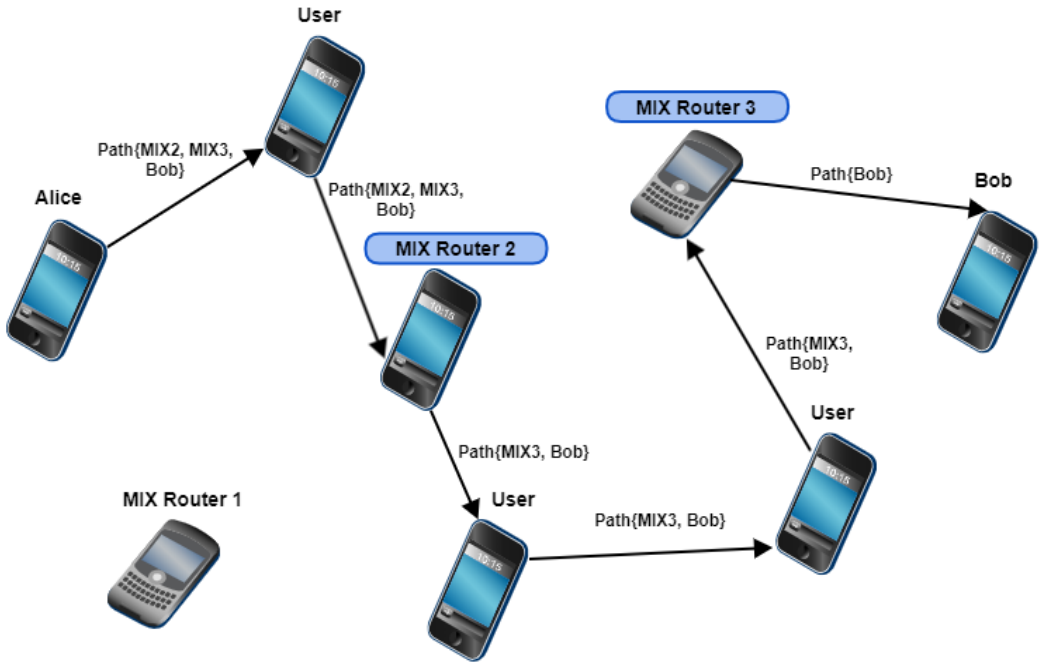

Figura 3.2: Path dinámico en red MIX.

3.3.3. Gestión de buffer

Los nodos MIX son nodos que van a almacenar una mayor cantidad de mensajes debido a su importancia en la red. Para que el buffer configurado en estos nodos no se llene de forma innecesaria, un MIX borra del buffer cualquier mensaje que se haya entregado de forma correcta, ya sea a un nodo final o en la propagación de un mensaje a un nodo cualquiera. No admite un nuevo mensaje en buffer si este ya se encuentra en él.

De la misma manera, un nodo MIX elimina el primer mensaje recibido de su buffer cuando no hay espacio suficiente para almacenar un nuevo mensaje. Esta medida

permite priorizar mensajes nuevos frente a mensajes antiguos, asumiendo que estos últimos han tenido más oportunidades de transmitirse por la red que los nuevos.

3.3.4. Gestión de las conexiones

Los nodos MIX también hacen uso de múltiples conexiones para el envío de mensajes (en contraposición a una configuración común de nodos *Epidemic Router*).

La tabla 3.2 resume las características de las redes MIX comentadas hasta el momento.

Característica	**Configuración**
Cifrado de mensajes	Simulado mediante parámetros del mensaje.
Reenvío de mensajes	Los mensajes no se reenvían en el nodo MIX hasta que el número de mensajes en buffer no haya superado el *threshold*.
Gestión de buffer	Eliminación de mensajes correctamente mandados y prioridad a nuevos mensajes recibidos. No se pueden duplicar mensajes en el buffer.
Gestión de las conexiones	Múltiples conexiones simultáneas.

Tabla 3.2: Características implementadas en nodos Onion.

3.4. Desarrollos complementarios

Además de los nodos anteriormente comentados, también se han desarrollado otras clases Java que son necesarias para la correcta implementación de las redes MIX y *Onion*.

3.4.1. Mensajes

La generación de mensajes es muy importante dentro del entorno propuesto. Aunque el simulador The One ya contiene alguna clase con la que crear mensajes y permitir el traspaso de información entre nodos, este modelo no funciona para las redes MIX y *Onion*.

Los modelos clásicos de mensajes ya desarrollados están preparados para viajar desde nodo emisor hasta nodo receptor sin tener en cuenta los nodos intermedios. Para las redes comentadas es necesario implementar de alguna manera el *path* que los mensajes deben de seguir. Asi pues, se ha desarrollado un tipo de mensaje que contiene en la cabecera el circuito a seguir.

Para el caso de redes MIX, el *path* del mensaje solamente indica por los nodos que debe de pasar en algún momento hasta llegar al nodo final. Por otro lado, el *path* en redes *Onion* es obligatorio completarlo de forma correcta en el orden indicado.

3.4.2. Creación de reportes

El simulador The One ya cuenta con algunos reportes interesantes con los que obtener datos estadísticos sobre las simulaciones realizadas. No obstante, se han implementado más reportes para aumentar esa información y que las mediciones sean más precisas.

3.4.3. Scripting para volcado de información

Los reportes ofrecen, de una manera clara y concisa, estadísticas sobre el tráfico y las conexiones que suceden durante la simulación. Sin embargo, esta información es difícil de tratar de cara a la obtención de reportes visuales (como por ejemplo los gráficos).

De cara a mejorar la eficiencia y obtener unos buenos resultados visuales, se han desarrollado algunos scripts en Bash con los que parsear los datos de las simulaciones y generar ficheros CSV. Estos ficheros son de fácil manejo para crear tablas o gráficos.

Capítulo 4

Configuración y ejecución de las simulaciones

4.1. Introducción

En este apartado se presentan los distintos diseños, configuraciones y ejecuciones que se han llevado a cabo con el fin de obtener la información necesaria para la conclusión de este trabajo.

4.2. Parámetros de simulación

Las simulaciones en el entorno de The One se presentan en formato de texto plato como archivos que contienen todos los parámetros necesarios para que esta se ejecute correctamente. Los parámetros más básicos necesarios para la simulación se pueden agruparen cuatro categorías acorde a la tabla 4.1.

Categoría	Parámetro	Descripción
Scenario	name	Nombre del escenario a simular
	simulate Connections	Valor booleano que indica si las conexiones han de ser simuladas por The One u obtenidas de un fichero externo.
	updateInterval	Tiempo de refresco de la simulación.
	endTime	Tiempo total de simulación.
	nrofHostGroups	Número de grupos (tipos de nodos) que participan en la simulación.
Interface	type	Tipo de interfaz.
	transmitSpeed	Velocidad de transmisión para una interfaz específica.
	transmitRange	Rango de transmisión para una interfaz específica.
Group	movementModel	Tipo de movimiento de los nodos.
	router	Tipo de Router a implementar en los nodos.
	nrofInterfaces	Número de interfaces en cada nodo.
	interface	Tipo de interfaz específica a configurar.
	speed	Velocidad de movimiento de los nodos.
	msgTtl	TTL de los mensajes generados por los nodos.
	nrofHosts	Número de nodos de un tipo concreto.
	groupID	Identificador de nodos.
	onlyRetransmit	Capacidad de los nodos para enviar mensajes o solo retransmitir.
	useOnionOrMix	Tipo de red a utilizar por los nodos.
	bufferSize	Tamaño del buffer de los nodos.
	dlvThres	Threshold para reenvío de mensajes en nodos MIX.
Events	nrof	Número de eventos distintos.
	class	Tipo de evento.
	size	Tamaño del mensaje en el caso de que un evento sea de este tipo.
	hosts	Rango de hosts para los cuales aplica el evento.
	Creation Interval	Tiempo en segundos para la creación de eventos.
	filePath	Path para la obtención de eventos en fichero externo.

Tabla 4.1: Parámetros de configuración en The One.

Los escenarios simulados se han diseñado en base a los parámetros descritos anteriormente. Se ha intentado que los valores elegidos para cada parámetro en cada simulación simulen diferentes tipos de tráfico y, con esto, diferentes tipos de aplicativos.

De esta manera, se podrá realizar un análisis un poco más intensivo sobre qué aplicaciones o qué patrones de tráfico pueden funcionar en redes MIX y *Onion*.

4.2.1. Parámetros de escenario

Estos parámetros identifican características generales de un escenario a simular. Entre ellos se destaca el parámetro *simulate Connections* y *nrofHostGroups*.

El primero de estos indica a The One si las conexiones entre nodos deben de ser simuladas o no. Para los escenarios descritos en este trabajo no deben de simularse ya que, como se comenta en el apartado 4.3, estas han sido obtenidas de sistemas reales.

El segundo parámetro indica el número de nodos distintos que aplican a un escenario concreto. En este caso, su valor es siempre dos a no ser que se simule el escenario sin la presencia de nodos *Onion* o MIX.

4.2.2. Parámetros de interfaz

Las interfaces son necesarias para establecer parámetros físicos de conexión en los nodos. En todos los casos se ha utilizado una interfaz con velocidad 10 MB/s. El rango, al simularse las conexiones con trazas reales, no aplica.

4.2.3. Parámetros de grupo

A cada tipo de nodo en una simulación se le denomina Grupo. Cada grupo será configurado con los parámetros que se desee acorde a lo que se espera conseguir.

Algunos de estos parámetros se han configurado con el mismo valor en todas las simulaciones ya que no aplicaban al análisis de las redes desarrolladas. Por otro lado, parámetros como *onlyRetransmit* o *useOnionOrMix* determinarán si los nodos de un grupo concreto pueden mandar mensajes o qué tipo de red deberán de utilizar, respectivamente.

4.2.4. Parámetros de evento

Se han definido dos tipos de eventos dentro de las simulaciones. El primero de ellos son las conexiones entre todos, obtenidas desde un fichero externo, tal y como se comenta en la sección 4.3.

El segundo tipo de eventos son los mensajes. Estos solamente aplican a los nodos emisores (los cuales también son receptores), ya que los nodos MIX y *Onion* solamente retransmiten.

4.3. Escenarios y trazas reales

Para dotar a las simulaciones de un carácter mucho más realista, se han utilizado trazas externas del repositorio *CRAWDAD*. Estas trazas describen las conexiones que sucedieron en la ciudad de Roma entre 304 taxis en un periodo de 30 días. No obstante, debido a la gran cantidad de escenarios a simular, solamente se ha utilizado un pequeño subconjunto del total de estas trazas. Más concretamente, las correspondientes a los primeros 30000 segundos de duración.

El uso de trazas reales que están almacenadas en ficheros externos a The One implica que este simulador ya no tiene que llevar a cabo las simulaciones de, en este caso, las conexiones. Debido a esto, si no se configura correctamente el tipo de movimiento de los nodos al utilizar trazas externas, el uso de la interfaz gráfica de The One no se recomienda ya que las conexiones sucederán aún cuando los nodos no estén dentro de rango de conexión.

No obstante, se ha configurado un mapa de Roma en The One para escenarios de pruebas y simulaciones que no conlleven trazas externas. Esta configuración se basa en el uso de un mapa OSM obtenido en *OpenStreetMap*. Para que The One lo reconozca y se pueda visualizar correctamente en la GUI, se ha convertido dicho mapa a formato WKT con el software *openJUMP*, donde se han mantenido las carreteras para que los taxis puedan circular por ellas. Se puede observar un ejemplo de simulación con trazas reales en la Figura 4.1.

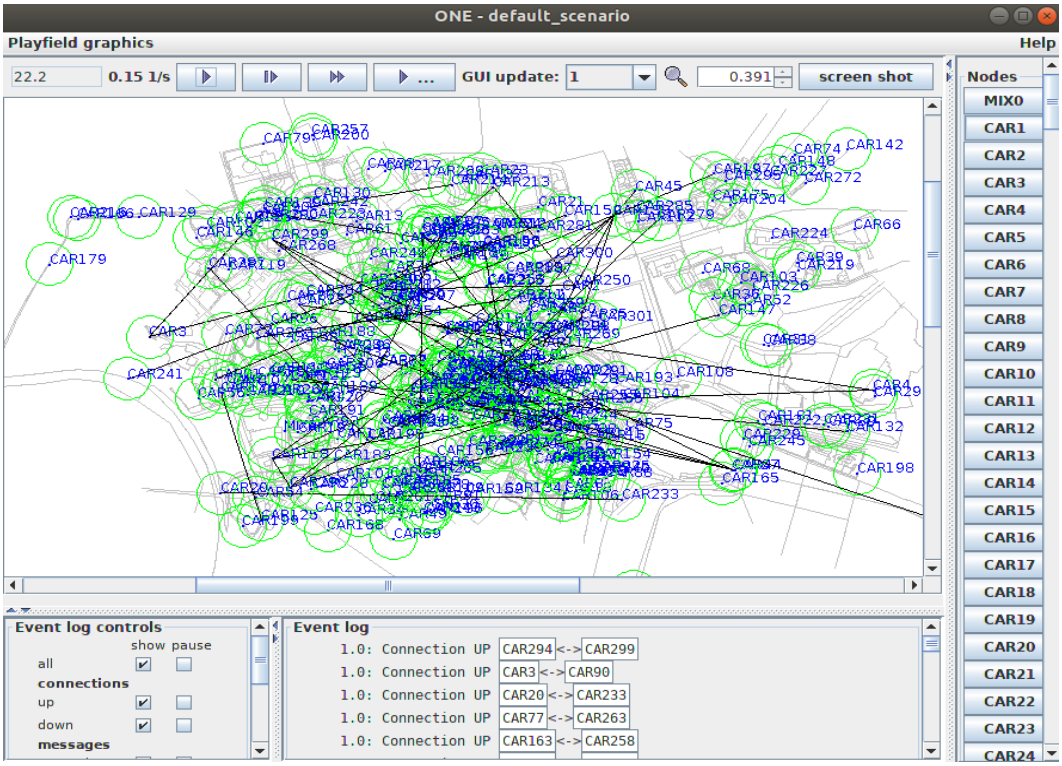

Figura 4.1: Simulación con trazas externas en Roma.

4.4. Configuración de los parámetros

Aunque existe un gran número de parámetros que pueden ser modificados para estudiar el comportamiento de las redes desarrolladas, se ha optado por realizar las simulaciones solamente con aquellos que podían tener una relación directa con los nodos nodos MIX y *Onion*. Esto es debido a la gran cantidad de escenarios posibles si se modifican todos los valores descritos.

En la tabla 4.1 se presentan tanto los valores que permanecen estáticos a lo largo de todas las simulaciones como aquellos específicos para ambos tipos de redes. Estos últimos, correspondientes a las tablas 4.3 y 4.4, contiene una relación de dichos parámetros con el número de escenario.

Categoría	**Parámetro**	**Valor**
Scenario	simulate Connections	False
	updateInterval	0.1 Seg
	endTime	30000 Seg
Interface	type	HighSpeedInterface
	transmitSpeed	10 MB/s
Group	nrofInterfaces	1
	interface	HighSpeedInterface
	msgTtl	300
	bufferSize	64MB en Epidemic Routers (nodos emisores y receptores)
Events	nrof	2
	class	ExternalEventsQueue y MessageEventGenerator
	filePath	Fichero de conexiones externo

Tabla 4.2: Parámetros estáticos en las simulaciones.

dlvThres	**Mix Buffer**	**Message Creation Interval**	**Message Size**	**Scen**
4 Messages	8MB	10, 15 Seg	64KB, 128KB	1
			1M, 2M	2
		300, 400 Seg	64KB, 128KB	3
			1M, 2M	4
	128MB	10, 15 Seg	64KB, 128KB	5
			1M, 2M	6
		300, 400 Seg	64KB, 128KB	7
			1M, 2M	8
16 Messages	8MB	10, 15 Seg	64KB, 128KB	9
			1M, 2M	10
		300, 400 Seg	64KB, 128KB	11
			1M, 2M	12
	128MB	10, 15 Seg	64KB, 128KB	13
			1M, 2M	14
		300, 400 Seg	64KB, 128KB	15
			1M, 2M	16

Tabla 4.3: Escenarios según configuración para simulaciones MIX.

Onion Buffer	**Message Creation Interval**	**Message Size**	**Scen**
8MB	10, 15 Seg	64KB, 128KB	17
		1M, 2M	18
	300, 400 Seg	64KB, 128KB	19
		1M, 2M	20
128MB	10, 15 Seg	64KB, 128KB	21
		1M, 2M	22
	300, 400 Seg	64KB, 128KB	23
		1M, 2M	24

Tabla 4.4: Escenarios según configuración para simulaciones Onion.

4.5. Ejecución de simulaciones

Las simulaciones en The One pueden ejecutarse desde la GUI mostrada anteriormente o en modo *batch*. Se ha utilizado este último para ahorrar tiempo de simulación y favorecer el scripting con el que automatizar la ejecución de las mismas y el volcado de información.

Se han ejecutado un total de 24 escenarios, de los cuales 16 corresponden a redes MIX (escenarios del 1 al 16) y 8 corresponden a *Onion* (escenarios del 17 al 24).

Cada escenario cuenta con cinco simulaciones para poder comparar también el rendimiento de la red al utilizar varios nodos MIX y *Onion*. La primera de estas simulaciones no cuenta con ninguno de estos nodos, es decir, es el comportamiento ideal de la red solamente utilizando nodos emisores y receptores *Epidemic Router*. En las cuatro simulaciones restantes sí que se añaden, de uno en uno, nodos MIX y *Onion* hasta un total de cuatro.

Así pues, se han simulado un total de 120 simulaciones con hasta 24 configuraciones de red diferentes.

Capítulo 5

Resultados obtenidos

5.1. Introducción

En este capítulo se exponen los resultados obtenidos en las simulaciones de las redes MIX y *Onion*. Se ha intentado en todo momento justificar dichos resultados para que, además de conocer si ambas tecnologías tienen un impacto negativo en las redes oportunistas, se puedan mejorar en desarrollos posteriores.

5.2. Parámetros evaluados

En la tabla 5.1 se ofrece una descripción de los valores medidos con el fin de entender correctamente los resultados obtenidos. Los resultados analizados que han sido obtenidos en las simulaciones se encuentran en el anexo A.

Dato	Descripción
Created	Número de mensajes creados durante la simulación.
Started	Número de intentos de transmisión de mensajes entre nodos.
Relayed	Número de transmisiones exitosas de mensajes entre nodos. Su valor máximo es *Started*.
Aborted	Número de transmisiones de mensajes canceladas por pérdida de conexión.
Dropped	Número de mensajes eliminados de buffer por falta de espacio y por expiración del TTL.
Dropped by TTL	Número de mensajes eliminados exclusivamente por expiración del TTL.
Removed	Número de mensajes eliminados por haberse recibido en el pasado o tras una transmisión exitosa.
Delivered	Número de mensajes transmitidos correctamente al nodo final hacia el que iban dirigidos.
Delivery Prob	Porcentaje de mensajes trasmitidos correctamente respecto a los totales creados.
Overhead Ratio	Ratio entre el overhead de paquete y sus datos.
Latency Avg	Media de la latencia de todas las transmisiones.
Latency Med	Valor medio de la latencia de todas las transmisiones.
Hopcount Avg	Media del número de saltos de todos los mensajes entregados.
Hopcount Med	Valor medio del número de saltos de todos los mensajes entregados.
Buffertime Avg	Media del tiempo en segundos que un mensaje pasa en buffer.
Buffertime Med	Valor medio del número de segundos que un mensaje pasa en buffer.

Tabla 5.1: Descripción de los valores medidos.

5.3. Resultados MIXNET

Tal y como se ha expuesto en los apartados 4.4 y 4.5, son cinco los parámetros que se han ido variando a lo largo de todas las simulaciones de redes MIX. A modo de recordatorio, estos son:

- Message Size.

- Message Creation Interval.
- Mix Buffer.
- dlvThres (mensajes a almacenar hasta liberar buffer).
- Número de nodos MIX participantes.

De los resultados obtenidos mostrados en el Anexo A cabe destacar que siempre que se incluyen dos o más nodos MIX para cualquiera de las configuraciones, no se llega a entregar ningún mensaje a los nodos finales. Este resultado no es de extrañar ya que sigue la tendencia general de mensajes entregados con nodos MIX.

	0 MIX	1 MIX	2 MIX	3 MIX	4 MIX
dlvThres = 4	16.26 %	2.6 %	0 %	0 %	0 %
dlvThres = 16	16.26 %	0.95 %	0 %	0 %	0 %

Tabla 5.2: Media de ratios entre mensajes creados y entregados.

La tabla 5.2 muestra la media entre todas las simulaciones del porcentaje de mensajes entregados. El uso de un nodo MIX hace que la probabilidad de que un mensaje llegue a su destino disminuya drásticamente. Si esta tendencia se alarga hasta los escenarios con más de un nodo MIX, se puede concluir que es necesario más tiempo de simulación para que llegue al menos un mensaje debido a la baja probabilidad de entrega.

En los resultados también se puede observar como el tamaño de mensaje influye directamente en el comportamiento de una red MIX. A mayor tamaño de mensaje, menor es el porcentaje de mensajes recibidos.

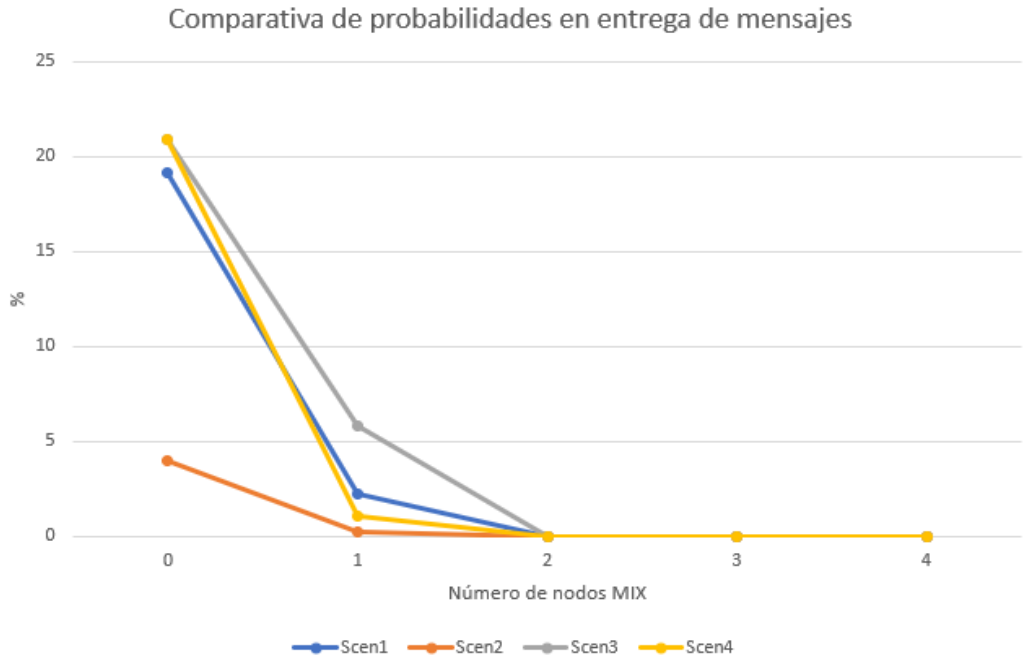

Figura 5.1: Comparativa entre probabilidades de entrega de mensajes MIX.

La gráfica 5.1 muestra los cuatro primeros escenarios resumidos en la tabla 4.3, es decir, para unos valores *dlvThres=4* y *buffer=8M*. Se han elegido estos escenarios ya que el aumento de buffer en el nodo MIX no tiene repercusión en el estado de la red.

Se puede observar como los escenarios uno y tres obtienen mejores resultados que los otros dos escenarios. Es decir, el envío de mensajes de pequeño tamaño hace que la red obtenga un mayor porcentaje de mensajes recibidos que aquellas redes con tráfico más pesado.

Entre estos dos escenarios que parecen ofrecer mejores resultados, también existe una clara diferencia a tener en cuenta. El escenario uno tiene la mitad de mensajes recibidos que el escenario tres si se comparan con el número total de mensajes enviados. Esto implica que las redes MIX se adaptan mejor a patrones de tráfico lentos en cuanto a creación de mensajes. Este comportamiento se puede ver en la tabla 5.3.

	Created	Delivered	Delivery prob	Dropped	Dropped by TTL
Escenario 1_1	2488	55	2.21 %	175404	12336
Escenario 3_1	86	5	5.81 %	2534	2534

Tabla 5.3: Datos obtenidos en simulaciones con dlvThres = 4.

Además, se puede diagnosticar el problema que existe en la red del Escenario 1_1 gracias a las estadísticas de paquetes perdidos. En este escenario, la gran mayoría de paquetes son eliminados de buffer por problemas de *overflow*. En el Escenario 3 no se han eliminado paquetes por *overflow* ya que la información que proporciona *Dropped by TTL* nos indica que, en realidad, ha expirado su TTL.

Estos resultados comentados sobre la mejora de capacidades en redes MIX al utilizar tráfico lento y de pequeño tamaño también se observan con un *threshold* mayor en los nodos MIX. La tabla 5.4 resume dicho comportamiento pero con el valor *dlvThres = 16*.

	Created	Delivered	Delivery prob	Dropped	Dropped by TTL
Escenario 9_1	2494	10	0.40 %	128508	11122
Escenario 11_1	86	2	2.33 %	2423	2423

Tabla 5.4: Datos obtenidos en simulaciones con dlvThres = 16.

Con un buffer para reenvío de mensajes mayor (*dlvThres*), los nodos MIX van a necesitar de mayor memoria donde almacenar dichos mensajes. Si este dimensionamiento no se tiene en cuenta, la capacidad de la red puede disminuir considerablemente, más especialmente en los Escenarios $9_1, 10_1, 11_1 y 12_1$ (es decir, en los escenarios donde se deberían de recibir mensajes).

En todos ellos, y acorde a la tabla 5.5, se puede apreciar un aumento de los envíos de mensajes entre nodos (*Started*), así como los mensajes eliminados de buffer por ya haberse recibido (*Removed*).

	Started	Removed	Delivered	Dropped	Buffertime Avg
Escenario 2	19109	1641	0.40 %	6	977.4612 Sec
Escenario 10	22471	4102	2.33 %	2	807.2029 Sec

Tabla 5.5: Comparación de mismas configuraciones con diferente dlvThres.

Cuando el buffer de los nodos MIX aumenta hasta 128MB, el comportamiento se normaliza hasta alcanzar casi los valores vistos en los Escenarios 4 a 8. Esta nueva configuración, aunque ofrece mayor seguridad al mezclar mayor cantidad de mensajes, hace que el porcentaje de envíos realizados con éxito disminuya un poco (entre dos y tres mensajes menos recibidos a excepción del escenario 14_1).

5.4. Resultados Onion Routing

Los escenarios evaluados han ofrecido los mismos resultados que las redes MIX en cuanto a los mensajes entregados con uno o más nodos *Onion* desplegados. En este caso, tampoco hay mensajes entregados para este tipo de configuraciones.

No obstante, se puede observar que un patrón de tráfico con un tiempo de creación de mensajes elevado, como por ejemplo 400 segundos, hace disminuir el rendimiento de la red en cuanto al número de mensajes entregados.

Este problema se debe a que los mensajes en redes *Onion* solamente se pueden enviar al siguiente nodo del path preestablecidos. Este comportamiento, junto con la baja tasa de creación de mensajes configurada, hace que sea muy complicado que suceda la comunicación necesaria entre un nodo n y $n+1$ del path para que el mensaje llegue a su destino final.

En la gráfica 5.2 se puede visualizar como para dos escenarios en los que solo varía el intervalo de creación de mensajes, el envío entre nodos *Onion* disminuye conforme estos aumentan en número.

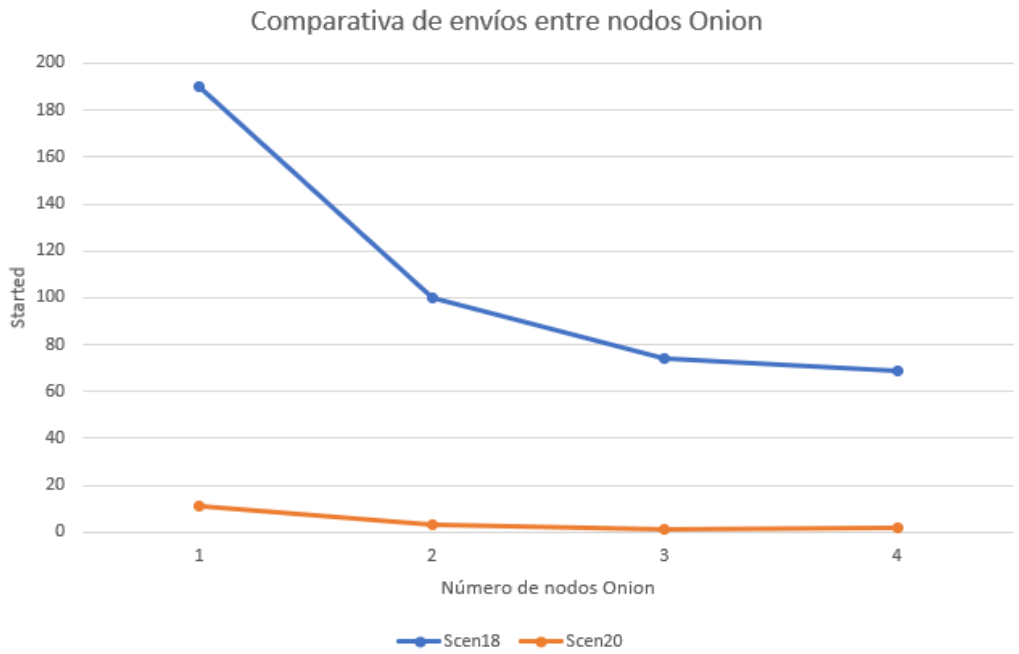

Figura 5.2: Comparativa entre envíos de mensajes en escenarios Onion.

Acorde a los resultados obtenidos y reflejados en el Anexo A, también se puede observar que un buffer pequeño disminuye el número de mensajes entregados a nodos finales. Al comparan los Escenarios 17 y 18 con 21 y 22 (ya que son iguales pero

modificando el buffer), se comprueba un aumento en el número de envíos de mensajes entre nodos *Onion* y una disminución de mensajes borrados de buffer para aquellos escenarios con un buffer mayor, tal y como se aprecia en la tabla 5.6.

	Buffer	Started	Dropped	Delivered
Escenario 17_1	8 MB	277	1009	18
Escenario 21_1	128 MB	281	984	22
Escenario 18_1	8 MB	190	1672	3
Escenario 22_1	128 MB	201	1602	17

Tabla 5.6: Análisis del impacto de Buffer en escenarios Onion.

Este tipo de redes, además de funcionar mejor en entornos con alta tasa de tráfico, también ofrecen mejores resultados con pequeños paquetes de datos. Por ejemplo, si se comparan los Escenarios 17 y 18, se observa cierta mejoría en cuanto al número de mensajes entregados respecto de los mensajes creados (ver Tabla 5.7).

	Message Size	Started	Dropped	Delivered	Delivered Prob
Escenario 17_1	64K, 128K	277	1009	18	0.72
Escenario 18_1	1MB, 2MB	190	1672	3	0.12

Tabla 5.7: Análisis del impacto de tamaño de paquete en escenarios Onion.

Capítulo 6

Conclusiones y líneas futuras

A lo largo de este trabajo se ha expuesto todo el desarrollo referente al diseño, configuración y análisis de tecnologías MIX y *Onion Routing* dentro de redes oportunistas. La idea principal que subyace tras esta investigación es intentar determinar si dichas tecnologías se pueden implantar en este tipo de redes sin que el rendimiento se vea afectado. De ser así, se podrían solucionar los problemas de seguridad que las redes oportunistas presentan, como por ejemplo la falta de privacidad o la incapacidad de ofrecer anonimato.

Del análisis de los resultados obtenidos para redes MIX se concluye que los mensajes entregados disminuyen cuando estos son de gran tamaño y su generación es rápida. Es decir, la probabilidad de que un mensaje llegue a su destino final en redes MIX depende directamente del tamaño de paquete y de la tasa de tráfico. El uso de esta tecnología para ofrecer privacidad y anonimato ofrecerá mejores resultados para aplicaciones con baja tasa de tráfico cuyos paquetes sean pequeños.

Además, una mejora del anonimato, la cual se consigue con un aumento del *dlvThres*, tiene consecuencias en cuanto al rendimiento general de la red. A mayor anonimato, menor ratio de mensajes entregados a nodos destino.

Al contrario que estas redes, las redes *Onion* evaluadas entregan más mensajes cuanto mayor es la tasa de tráfico. Esta probabilidad de entrega también se ve aumentada cuanto mayor es el buffer, parámetro que en las redes MIX no influye directamente acorde a los resultados obtenidos.

Otro resultado a tener en cuenta de cara al diseño de una red *Onion* es el tamaño de mensaje. Al igual que en redes MIX, el uso de mensajes de menor tamaño aumenta los mensajes entregados. Acorde a esto, las redes *Onion* oportunistas se comportan mejor con nodos intermedios que tengan un buffer elevado y aplicativos con una alta generación de mensajes pequeños. Para ambos escenarios, el uso de más de un nodo intermedio no está recomendado debido a los resultados negativos obtenidos.

Tras estas conclusiones, aún es necesario solucionar algunos problemas de

implementación de cara al uso de estas tecnologías en entornos reales oportunistas. Así pues, para líneas de investigación futuras, hay que evaluar el impacto de algoritmos de cifrado concretos con los que aplicar técnicas *Onion*. También es necesario, y con más urgencia que lo anterior, desarrollar un sistema de gestión y obtención de claves en entornos distribuidos para su implementación en situaciones oportunistas.

Capítulo 7

Bibliografía

[1] Barun Saha. The one tutorial. http://delay-tolerant-networks.blogspot.com/p/one-tutorial.html.

[2] Nikhitha. One simulator introduction. http://one-simuator-for-beginners.blogspot.com/2013/08/one-simulator-introduction.html.

[3] Netlab. The opportunistic network environment simulator. https://www.netlab.tkk.fi/tutkimus/dtn/theone/.

[4] Netlab. The one javadoc. https://www.netlab.tkk.fi/tutkimus/dtn/theone/javadoc_v141/.

[5] OpenStreetMap. Guía de principiantes. https://wiki.openstreetmap.org/wiki/ES:GuÃŋa_de_principiantes.

[6] OpenJUMP. The openjump wiki. http://ojwiki.soldin.de/index.php?title=Main_Page.

[7] Pierpaolo Loreti Giuseppe Bianchi Raul Amici Antonello Rabuffi Lorenzo Bracciale, Marco Bonola. The roma/taxi dataset. Technical report, July 2014. https://crawdad.org/roma/taxi/20140717/.

[8] OpenJUMP. The openjump wiki. http://ojwiki.soldin.de/index.php?title=Main_Page.

[9] Paul F. Syverson Michael G. Reed and David M. Goldschlag. Anonymous connections and onion routing. Technical report, 1998. https://www.onion-router.net/Publications/JSAC-1998.pdf.

[10] European Commission. What is a mix-net? https://ec.europa.eu/digital-single-market/en/news/what-mix-net-find-out-about-technology-core-panoramix-project.

[11] Wikipedia. Mix network. https://en.wikipedia.org/wiki/Mix_network.

[12] Oracle. Java^{TM} platform, standard edition 6 api specification. https://docs.oracle.com/javase/6/docs/api/.

[13] C. Pérez-Solà D. Chen, G. Navarro-Arribas and J. Borrell. Message anonymity on predictable opportunistic networks. Technical report, July 2019. https://link.springer.com/article/10.1007/s12652-019-01393-0.

Lista de Figuras

1.1. Diagrama temporal 3

2.1. Envío de mensaje en red Onion. 7
2.2. Envío de mensaje en red MIX. 8
2.3. Captura del simulador The One. 9

3.1. Path preestablecido en red Onion. 13
3.2. Path dinámico en red MIX. 15

4.1. Simulación con trazas externas en Roma. 23

5.1. Comparativa entre probabilidades de entrega de mensajes MIX. 30
5.2. Comparativa entre envíos de mensajes en escenarios Onion. 32

Anexos

Anexos A

Resultados de las simulaciones

A.1. Resultados para red MIX

A continuación, se presentan los distintos escenarios propuestos para redes MIX.

dlvThres	**Mix Buffer**	**Message Creation Interval**	**Message Size**	**Mix Num**	**Scen**
4 Messages	8MB	10, 15	64KB, 128KB	0	$scen1_0$
				1	$scen1_1$
				2	$scen1_2$
				3	$scen1_3$
				4	$scen1_4$
			1M, 2M	0	$scen2_0$
				1	$scen2_1$
				2	$scen2_2$
				3	$scen2_3$
				4	$scen2_4$
		300, 400	64KB, 128KB	0	$scen3_0$
				1	$scen3_1$
				2	$scen3_2$
				3	$scen3_3$
				4	$scen3_4$
			1M, 2M	0	$scen3_0$
				1	$scen4_1$
				2	$scen4_2$
				3	$scen4_3$
				4	$scen4_4$

Tabla A.1: Escenarios MIX con dlvThres 4 y Buffer 8MB.

dlvThres	**Mix Buffer**	**Message Creation Interval**	**Message Size**	**Mix Num**	**Scen**
4 Messages	128MB	10, 15	64KB, 128KB	0	$scen5_0$
				1	$scen5_1$
				2	$scen5_2$
				3	$scen5_3$
				4	$scen5_4$
			1M, 2M	0	$scen6_0$
				1	$scen6_1$
				2	$scen6_2$
				3	$scen6_3$
				4	$scen6_4$
		300, 400	64KB, 128KB	0	$scen7_0$
				1	$scen7_1$
				2	$scen7_2$
				3	$scen7_3$
				4	$scen7_4$
			1M, 2M	0	$scen8_0$
				1	$scen8_1$
				2	$scen8_2$
				3	$scen8_3$
				4	$scen8_4$

Tabla A.2: Escenarios MIX con dlvThres 4 y Buffer 128MB.

dlvThres	**Mix Buffer**	**Message Creation Interval**	**Message Size**	**Mix Num**	**Scen**
16 Messages	8MB	10, 15	64KB, 128KB	0	$scen9_0$
				1	$scen9_1$
				2	$scen9_2$
				3	$scen9_3$
				4	$scen9_4$
			1M, 2M	0	$scen10_0$
				1	$scen10_1$
				2	$scen10_2$
				3	$scen10_3$
				4	$scen10_4$
		300, 400	64KB, 128KB	0	$scen11_0$
				1	$scen11_1$
				2	$scen11_2$
				3	$scen11_3$
				4	$scen11_4$
			1M, 2M	0	$scen12_0$
				1	$scen12_1$
				2	$scen12_2$
				3	$scen12_3$
				4	$scen12_4$

Tabla A.3: Escenarios MIX con dlvThres 16 y Buffer 8MB.

dlvThres	**Mix Buffer**	**Message Creation Interval**	**Message Size**	**Mix Num**	**Scen**
16 Messages	128MB	10, 15	64KB, 128KB	0	$scen13_0$
				1	$scen13_1$
				2	$scen13_2$
				3	$scen13_3$
				4	$scen13_4$
			1M, 2M	0	$scen14_0$
				1	$scen14_1$
				2	$scen14_2$
				3	$scen14_3$
				4	$scen14_4$
		300, 400	64KB, 128KB	0	$scen15_0$
				1	$scen15_1$
				2	$scen15_2$
				3	$scen15_3$
				4	$scen15_4$
			1M, 2M	0	$scen16_0$
				1	$scen16_1$
				2	$scen16_2$
				3	$scen16_3$
				4	$scen16_4$

Tabla A.4: Escenarios MIX con dlvThres 16 y Buffer 128MB.

Las siguientes tablas muestran los resultados obtenidos para cada escenario presentado anteriormente.

Scen	**Created**	**Started**	**Relayed**	**Aborted**	**Dropped**	**Dropped TTL**	**Removed**	**delivered**	**delivery prob**	**overhead ratio**
$scen1_0$	2494	538819	538454	362	63657	63657	409238	479	0.1921	1123.1211
$scen1_1$	2488	203038	202876	161	175404	12336	21711	55	0.0221	3687.6545
$scen1_2$	2476	206640	206496	144	193517	12146	7062	0	0.0000	
$scen1_3$	2473	215131	214986	145	201931	11612	7067	0	0.0000	
$scen1_4$	2464	204688	204537	141	192326	11651	6318	0	0.0000	
$scen2_0$	2494	134052	133863	188	123833	9182	5759	99	0.0397	1351.1515
$scen2_1$	2488	19109	19090	19	18852	184	1641	6	0.0024	3180.6667
$scen2_2$	2481	36331	36313	18	36394	181	1329	0	0.0000	
$scen2_3$	2479	34736	34717	18	35012	181	956	0	0.0000	
$scen2_4$	2471	14494	14466	21	13954	173	1591	0	0.0000	
$scen3_0$	86	13834	13817	17	2767	2767	8283	18	0.2093	766.6111
$scen3_1$	86	6269	6261	8	2534	2534	1476	5	0.0581	1251.2000
$scen3_2$	86	7046	7039	7	2507	2507	1418	0	0.0000	
$scen3_3$	85	5915	5907	8	1993	1993	561	0	0.0000	
$scen3_4$	84	5907	5886	12	2384	2384	584	0	0.0000	
$scen4_0$	86	10624	10595	29	2755	2755	5088	18	0.2093	587.6111
$scen4_1$	86	6715	6702	13	5297	693	951	1	0.0116	6701.0000
$scen4_2$	86	4561	4557	4	4074	692	102	0	0.0000	
$scen4_3$	85	3178	3169	9	2542	640	373	0	0.0000	
$scen4_4$	84	5730	5726	4	4332	633	1010	0	0.0000	

Tabla A.5: Resultados para escenarios MIX con dlvThres 4 y Buffer 8MB.

Scen	**latency avg**	**latency med**	**hopcount avg**	**hopcount med**	**buffertime avg**	**buffertime med**
$scen1_0$	7616.8862	6827.1000	4.7453	5	1313.6081	0.1000
$scen1_1$	8569.8164	6562.1000	19.4727	19	899.7838	64.9000
$scen1_2$				0	886.6408	55.9000
$scen1_3$				0	818.6665	43.0000
$scen1_4$				0	845.9530	31.3000
$scen2_0$	5886.1919	4611.2000	6.9697	6	1126.7232	198.0000
$scen2_1$	4835.6333	6252.2000	12.3333	10	977.4612	33.1000
$scen2_2$				0	509.3616	0.6000
$scen2_3$				0	534.1234	0.6000
$scen2_4$				0	1237.2900	161.8000
$scen3_0$	6250.4556	6448.1000	4.5556	5	2548.2314	0.1000
$scen3_1$	9777.8000	8291.1000	15.6000	14	6155.4285	4478.1000
$scen3_2$				0	5867.6764	3964.8000
$scen3_3$				0	6522.0877	5539.6000
$scen3_4$				0	7825.3793	7574.7000
$scen4_0$	6250.6278	6448.2000	4.5556	5	3581.2887	0.1000
$scen4_1$	6588.2000	6588.2000	14.0000	14	1287.4458	91.0000
$scen4_2$				0	2129.1948	185.8000
$scen4_3$				0	2365.6513	627.8000
$scen4_4$				0	1690.4636	67.7000

Tabla A.6: Resultados para escenarios MIX con dlvThres 4 y Buffer 8MB.

Scen	**Created**	**Started**	**Relayed**	**Aborted**	**Dropped**	**Dropped TTL**	**Removed**	**delivered**	**delivery prob**	**overhead ratio**
$scen5_0$	2494	538819	538454	362	63657	63657	409238	479	0.1921	1123.1211
$scen5_1$	2488	185303	185155	148	169932	12520	9514	56	0.0225	3305.3393
$scen5_2$	2476	188452	188297	155	174504	12186	7941	0	0.0000	
$scen5_3$	2473	201812	201672	140	188990	11587	6939	0	0.0000	
$scen5_4$	2464	174676	174540	131	162008	11826	6786	0	0.0000	
$scen6_0$	2494	134052	133863	188	123833	9182	5759	99	0.0397	1351.1515
$scen6_1$	2488	23438	23418	20	22924	178	1823	6	0.0024	3902.0000
$scen6_2$	2481	35863	35845	18	36186	197	1097	0	0.0000	
$scen6_3$	2479	33681	33661	19	33983	246	1085	0	0.0000	
$scen6_4$	2471	14406	14361	21	13875	215	1712	0	0.0000	
$scen7_0$	86	13834	13817	17	2767	2767	8283	18	0.2093	766.6111
$scen7_1$	86	6269	6261	8	2534	2534	1476	5	0.0581	1251.2000
$scen7_2$	86	6165	6158	7	2510	2510	563	0	0.0000	
$scen7_3$	85	5866	5856	10	1994	1994	496	0	0.0000	
$scen7_4$	84	6106	6092	12	2384	2384	813	0	0.0000	
$scen8_0$	86	10624	10595	29	2755	2755	5088	18	0.2093	587.6111
$scen8_1$	86	6976	6966	10	5438	721	1023	1	0.0116	6965.0000
$scen8_2$	86	8512	8498	14	7148	684	852	0	0.0000	
$scen8_3$	85	4567	4564	3	4172	651	91	0	0.0000	
$scen8_4$	84	4219	4215	4	3780	673	101	0	0.0000	

Tabla A.7: Resultados para escenarios MIX con dlvThres 4 y Buffer 128MB.

Scen	**latency avg**	**latency med**	**hopcount avg**	**hopcount med**	**buffertime avg**	**buffertime med**
$scen5_0$	7616.8862	6827.1000	4.7453	5	1313.6081	0.1000
$scen5_1$	8657.9125	7999.1000	22.4643	20	986.2778	110.0000
$scen5_2$				0	971.0156	116.0000
$scen5_3$				0	872.9366	65.0000
$scen5_4$				0	1001.1164	99.9000
$scen6_0$	5886.1919	4611.2000	6.9697	6	1126.7232	198.0000
$scen6_1$	3259.0667	2188.6000	11.1667	14	798.6234	1.5000
$scen6_2$				0	530.2533	0.6000
$scen6_3$				0	554.6431	0.6000
$scen6_4$				0	1238.3840	167.8000
$scen7_0$	6250.4556	6448.1000	4.5556	5	2548.2314	0.1000
$scen7_1$	9777.8000	8291.1000	15.6000	14	6155.4285	4478.1000
$scen7_2$				0	7451.8179	6670.5000
$scen7_3$				0	6684.6196	5947.6000
$scen7_4$				0	7275.3902	6611.0000
$scen8_0$	6250.6278	6448.2000	4.5556	5	3581.2887	0.1000
$scen8_1$	6588.2000	6588.2000	14.0000	14	1246.4870	41.8000
$scen8_2$				0	1119.4371	1.0000
$scen8_3$				0	1615.5393	67.8000
$scen8_4$				0	2402.6272	113.8000

Tabla A.8: Resultados para escenarios MIX con dlvThres 4 y Buffer 128MB.

Scen	**Created**	**Started**	**Relayed**	**Aborted**	**Dropped**	**Dropped TTL**	**Removed**	**delivered**	**delivery prob**	**overhead ratio**
$scen9_0$	2494	538819	538454	362	63657	63657	409238	479	0.1921	1123.1211
$scen9_1$	2488	148311	148195	116	128508	11122	14307	10	0.0040	14818.5000
$scen9_2$	2476	177480	177335	145	160008	12322	10838	0	0.0000	
$scen9_3$	2473	203547	203404	143	187748	11538	9327	0	0.0000	
$scen9_4$	2464	172677	172407	134	156548	11744	9714	0	0.0000	
$scen10_0$	2494	134052	133863	188	123833	9182	5759	99	0.0397	1351.1515
$scen10_1$	2488	22471	22453	18	19913	163	4102	2	0.0008	11225.5000
$scen10_2$	2481	38359	38344	15	38238	209	1650	0	0.0000	
$scen10_3$	2479	31741	31723	17	32489	195	771	0	0.0000	
$scen10_4$	2471	11761	11747	14	12284	190	958	0	0.0000	
$scen11_0$	86	13834	13817	17	2767	2767	8283	18	0.2093	766.6111
$scen11_1$	86	8552	8542	10	2423	2423	3892	2	0.0233	4270.0000
$scen11_2$	86	8379	8368	11	2432	2432	2852	0	0.0000	
$scen11_3$	85	7813	7803	10	1921	1921	2524	0	0.0000	
$scen11_4$	84	7690	7681	9	2337	2337	2486	0	0.0000	
$scen12_0$	86	10624	10595	29	2755	2755	5088	18	0.2093	587.6111
$scen12_1$	86	11512	11502	10	7814	588	3343	0	0.0000	
$scen12_2$	86	5165	5160	5	3595	733	1231	0	0.0000	
$scen12_3$	85	4387	4379	8	2976	651	1120	0	0.0000	
$scen12_4$	84	3788	3786	2	3157	609	285	0	0.0000	

Tabla A.9: Resultados para escenarios MIX con dlvThres 16 y Buffer 8MB.

Scen	**latency avg**	**latency med**	**hopcount avg**	**hopcount med**	**buffertime avg**	**buffertime med**
$scen9_0$	7616.8862	6827.1000	4.7453	5	1313.6081	0.1000
$scen9_1$	6478.4100	5487.3000	8.8000	7	1155.9975	174.1000
$scen9_2$				0	1045.8150	145.1000
$scen9_3$				0	868.3590	70.9000
$scen9_4$				0	1000.9372	106.0000
$scen10_0$	5886.1919	4611.2000	6.9697	6	1126.7232	198.0000
$scen10_1$	564.6500	918.2000	5.5000	7	807.2029	1.0000
$scen10_2$				0	486.1294	0.6000
$scen10_3$				0	576.6407	0.6000
$scen10_4$				0	1464.5642	309.0000
$scen11_0$	8 6250.4556	6448.1000	4.5556	5	2548.2314	0.1000
$scen11_1$	11872.7500	16882.4000	10.0000	17	3564.7526	450.7000
$scen11_2$				0	4132.2645	942.2000
$scen11_3$				0	3501.0528	574.1000
$scen11_4$				0	4673.3419	1222.8000
$scen12_0$	6250.6278	6448.2000	4.5556	5	3581.2887	0.1000
$scen12_1$				0	677.1534	0.7000
$scen12_2$				0	1833.0243	78.8000
$scen12_3$				0	1704.2127	175.4000
$scen12_4$				0	2617.3584	135.0000

Tabla A.10: Resultados para escenarios MIX con dlvThres 4 y Buffer 128MB.

Scen	**Created**	**Started**	**Relayed**	**Aborted**	**Dropped**	**Dropped TTL**	**Removed**	**delivered**	**delivery prob**	**overhead ratio**
$scen13_0$	2494	538819	538454	362	63657	63657	409238	479	0.1921	1123.1211
$scen13_1$	2488	186245	186082	163	167168	12035	12610	53	0.0213	3509.9811
$scen13_2$	2476	179353	179204	147	163166	12654	10057	0	0.0000	
$scen13_3$	2473	198829	198682	147	182455	11443	9930	0	0.0000	
$scen13_4$	2464	166467	166274	127	150054	12089	10199	0	0.0000	
$scen14_0$	2494	134052	133863	188	123833	9182	5759	99	0.0397	1351.1515
$scen14_1$	2488	19342	19328	14	18021	237	2779	9	0.0036	2146.5556
$scen14_2$	2481	40313	40291	22	40057	230	1703	0	0.0000	
$scen14_3$	2479	33180	33162	17	33424	279	1272	0	0.0000	
$scen14_4$	2471	13305	13288	17	13140	233	1651	0	0.0000	
$scen15_0$	86	13834	13817	17	2767	2767	8283	18	0.2093	766.6111
$scen15_1$	86	8552	8542	10	2423	2423	3892	2	0.0233	4270.0000
$scen15_2$	86	8859	8846	13	2396	2396	3352	0	0.0000	
$scen15_3$	85	7623	7612	11	1944	1944	2301	0	0.0000	
$scen15_4$	84	8116	8106	10	2339	2339	2901	0	0.0000	
$scen16_0$	86	10624	10595	29	2755	2755	5088	18	0.2093	587.6111
$scen16_1$	86	11487	11477	10	7785	591	3350	0	0.0000	
$scen16_2$	86	6445	6440	5	3248	632	2834	0	0.0000	
$scen16_3$	85	5865	5861	4	2747	597	2837	0	0.0000	
$scen16_4$	84	3898	3892	6	3044	600	505	0	0.0000	

Tabla A.11: Resultados para escenarios MIX con dlvThres 16 y Buffer 128MB.

Scen	**latency avg**	**latency med**	**hopcount avg**	**hopcount med**	**buffertime avg**	**buffertime med**
$scen13_0$	7616.8862	6827.1000	4.7453	5	1313.6081	0.1000
$scen13_1$	8730.2208	7999.1000	24.1698	22	980.4897	100.4000
$scen13_2$				0	1024.1566	139.5000
$scen13_3$				0	886.7185	78.7000
$scen13_4$				0	1040.9099	124.3000
$scen14_0$	5886.1919	4611.2000	6.9697	6	1126.7232	198.0000
$scen14_1$	6379.1444	3070.6000	22.3333	10	967.5160	45.0000
$scen14_2$				0	468.5020	0.6000
$scen14_3$				0	567.0728	0.6000
$scen14_4$				0	1325.6988	235.8000
$scen15_0$	6250.4556	6448.1000	4.5556	5	2548.2314	0.1000
$scen15_1$	11872.7500	16882.4000	10.0000	17	3564.7526	450.7000
$scen15_2$				0	3603.6307	333.0000
$scen15_3$				0	3716.0669	596.5000
$scen15_4$				0	4283.1242	262.0000
$scen16_0$	6250.6278	6448.2000	4.5556	5	3581.2887	0.1000
$scen16_1$				0	681.9388	0.7000
$scen16_2$				0	1456.3672	0.5000
$scen16_3$				0	1209.0824	0.4000
$scen16_4$				0	2564.5378	166.8000

Tabla A.12: Resultados para escenarios MIX con dlvThres 4 y Buffer 128MB.

Scen	Created	Relayed	Aborted	Dropped	Dropped TTL	Removed	delivered	delivery prob	overhead ratio	latency avg	
4 Messages	8MB	10, 15	64KB, 128KB	0	2494	538819	538454	362	63657	409238	
				1	2488	203038	202876	161	175404	12336	21
				2	2476	206640	206496	144	193517	a	
				3	a	a	a	a	a	a	
				4	a	a	a	a	a	a	
			1M, 2M	0	a	a	a	a	a	a	
				1	a	a	a	a	a	a	
				2	a	a	a	a	a	a	
				3	a	a	a	a	a	a	
				4	a	a	a	a	a	a	
		300, 400	64KB, 128KB	0	a	a	a	a	a	a	
				1	a	a	a	a	a	a	
				2	a	a	a	a	a	a	
				3	a	a	a	a	a	a	
				4	a	a	a	a	a	a	
			1M, 2M	0	a	a	a	a	a	a	
				1	a	a	a	a	a	a	
				2	a	a	a	a	a	a	
				3	a	a	a	a	a	a	
				4	a	a	a	a	a	a	

Tabla A.13: Escenarios según configuración para simulaciones MIX.

A.2. Resultados para red Onion

A continuación, se presentan los distintos escenarios propuestos para redes *Onion*.

Onion Buffer	**Message Creation Interval**	**Message Size**	**Onion Num**	**Scen**
8MB	10, 15	64KB, 128KB	0	$scen17_0$
			1	$scen17_1$
			2	$scen17_2$
			3	$scen17_3$
			4	$scen17_4$
		1M, 2M	0	$scen18_0$
			1	$scen18_1$
			2	$scen18_2$
			3	$scen18_3$
			4	$scen18_4$
	300, 400	64KB, 128KB	0	$scen19_0$
			1	$scen19_1$
			2	$scen19_2$
			3	$scen19_3$
			4	$scen19_4$
		1M, 2M	0	$scen20_0$
			1	$scen20_1$
			2	$scen20_2$
			3	$scen20_3$
			4	$scen20_4$

Tabla A.14: Escenarios Onion con Buffer 8MB.

Onion Buffer	**Message Creation Interval**	**Message Size**	**Onion Num**	**Scen**
128MB	10, 15	64KB, 128KB	0	$scen21_0$
			1	$scen21_1$
			2	$scen21_2$
			3	$scen21_3$
			4	$scen21_4$
		1M, 2M	0	$scen22_0$
			1	$scen22_1$
			2	$scen22_2$
			3	$scen22_3$
			4	$scen22_4$
	300, 400	64KB, 128KB	0	$scen23_0$
			1	$scen23_1$
			2	$scen23_2$
			3	$scen23_3$
			4	$scen23_4$
		1M, 2M	0	$scen24_0$
			1	$scen24_1$
			2	$scen24_2$
			3	$scen24_3$
			4	$scen24_4$

Tabla A.15: Escenarios Onion con Buffer 128MB.

Scen	**Created**	**Started**	**Relayed**	**Aborted**	**Dropped**	**Dropped TTL**	**Removed**	**delivered**	**delivery prob**	**overhead ratio**
$scen17_0$	2494	538819	538454	362	63657	63657	409238	479	0.1921	1123.1211
$scen17_1$	2488	277	277	0	1009	874	273	18	0.0072	14.3889
$scen17_2$	2476	126	126	0	992	984	126	0	0.0000	
$scen17_3$	2473	88	88	0	986	986	88	0	0.0000	
$scen17_4$	2464	74	73	0	981	981	74	0	0.0000	
$scen18_0$	2494	134052	133863	188	123833	9182	5759	99	0.0397	1351.1515
$scen18_1$	2488	190	190	0	1672	84	190	3	0.0012	62.3333
$scen18_2$	2481	100	100	0	1675	115	100	0	0.0000	
$scen18_3$	2479	74	74	0	1679	121	74	0	0.0000	
$scen18_4$	2471	69	66	0	1659	115	69	0	0.0000	
$scen19_0$	86	13834	13817	17	2767	2767	8283	18	0.2093	766.6111
$scen19_1$	86	11	11	0	34	34	11	0	0.0000	
$scen19_2$	86	1	1	0	34	34	1	0	0.0000	
$scen19_3$	85	3	3	0	33	33	3	0	0.0000	
$scen19_4$	84	3	3	0	32	32	3	0	0.0000	
$scen20_0$	86	10624	10595	29	2755	2755	5088	18	0.2093	587.6111
$scen20_1$	86	11	11	0	35	29	11	0	0.0000	
$scen20_2$	86	3	3	0	34	34	3	0	0.0000	
$scen20_3$	85	1	1	0	33	33	1	0	0.0000	
$scen20_4$	84	2	2	0	32	31	2	0	0.0000	

Tabla A.16: Resultados para escenarios Onion con Buffer 8MB.

Scen	**latency avg**	**latency med**	**hopcount avg**	**hopcount med**	**buffertime avg**	**buffertime med**
$scen17_0$	7616.8862	6827.1000	4.7453	5	1313.6081	0.1000
$scen17_1$	7540.4944	7813.1000	1.9444	2	13740.3761	17953.6000
$scen17_2$				0	16085.4115	17963.0000
$scen17_3$				0	16565.0830	17965.2000
$scen17_4$				0	16798.0837	17966.0000
$scen18_0$	5886.1919	4611.2000	6.9697	6	1126.7232	198.0000
$scen18_1$	1067.2000	918.2000	2.0000	2	7115.6604	6518.0000
$scen18_2$				0	7801.2368	6917.0000
$scen18_3$				0	8132.7159	7325.0000
$scen18_4$				0	8083.3979	7420.0000
$scen19_0$	6250.4556	6448.1000	4.5556	5	2548.2314	0.1000
$scen19_1$				0	14163.2111	17947.3000
$scen19_2$				0	17454.3086	17961.4000
$scen19_3$				0	16636.6139	17967.7000
$scen19_4$				0	16438.8571	17953.8000
$scen20_0$	6250.6278	6448.2000	4.5556	5	3581.2887	0.1000
$scen20_1$				0	13719.7413	17947.3000
$scen20_2$				0	16659.0270	17960.9000
$scen20_3$				0	17555.1206	17970.9000
$scen20_4$				0	16814.7765	17961.3000

Tabla A.17: Resultados para escenarios Onion con Buffer 8MB.

Scen	**Created**	**Started**	**Relayed**	**Aborted**	**Dropped**	**Dropped TTL**	**Removed**	**delivered**	**delivery prob**	**overhead ratio**
$scen21_0$	2494	538819	538454	362	63657	63657	409238	479	0.1921	1123.1211
$scen21_1$	2488	281	281	0	984	984	276	22	0.0088	11.7727
$scen21_2$	2476	135	135	0	992	992	135	0	0.0000	
$scen21_3$	2473	94	94	0	986	986	94	0	0.0000	
$scen21_4$	2464	87	84	0	979	979	87	0	0.0000	
$scen22_0$	2494	134052	133863	188	123833	9182	5759	99	0.0397	1351.1515
$scen22_1$	2488	201	201	0	1602	112	197	17	0.0068	10.8235
$scen22_2$	2481	84	84	0	1639	167	84	0	0.0000	
$scen22_3$	2479	57	57	0	1656	152	57	0	0.0000	
$scen22_4$	2471	64	60	0	1642	145	64	0	0.0000	
$scen23_0$	86	13834	13817	17	2767	2767	8283	18	0.2093	766.6111
$scen23_1$	86	11	11	0	34	34	11	0	0.0000	
$scen23_2$	86	4	4	0	34	34	4	0	0.0000	
$scen23_3$	85	3	3	0	33	33	3	0	0.0000	
$scen23_4$	84	84	0	0	0	32	32	0	0.0000	
$scen24_0$	86	10624	10595	29	2755	2755	5088	18	0.2093	587.6111
$scen24_1$	86	11	11	0	34	34	11	0	0.0000	
$scen24_2$	86	6	6	0	34	34	6	0	0.0000	
$scen24_3$	85	4	4	0	33	33	4	0	0.0000	
$scen24_4$	84	2	2	0	32	31	2	0	0.0000	

Tabla A.18: Resultados para escenarios Onion con Buffer 128MB.

Scen	**latency avg**	**latency med**	**hopcount avg**	**hopcount med**	**buffertime avg**	**buffertime med**
$scen21_0$	7616.8862	6827.1000	4.7453	5	1313.6081	0.1000
$scen21_1$	8557.2909	9007.6000	1.9545	2	14399.7244	17954.6000
$scen21_2$				0	15984.3971	17962.6000
$scen21_3$				0	16545.1712	17965.5000
$scen21_4$				0	16604.2126	17965.8000
$scen22_0$	5886.1919	4611.2000	6.9697	6	1126.7232	198.0000
$scen22_1$	7530.2824	7813.2000	2.0000	2	8015.4394	7440.0000
$scen22_2$				0	8386.4424	7404.0000
$scen22_3$				0	8558.7922	7737.0000
$scen22_4$				0	8392.0485	7752.0000
$scen23_0$	6250.4556	6448.1000	4.5556	5	2548.2314	0.1000
$scen23_1$				0	14163.2111	17947.3000
$scen23_2$				0	16403.8421	17960.9000
$scen23_3$				0	16498.4444	17967.7000
$scen23_4$				0	17966.7156	17962.6000
$scen24_0$	6250.6278	6448.2000	4.5556	5	3581.2887	0.1000
$scen24_1$				0	14163.2267	17947.3000
$scen24_2$				0	15680.7850	17953.5000
$scen24_3$				0	16212.9270	17961.4000
$scen24_4$				0	16920.1353	17961.3000

Tabla A.19: Resultados para escenarios Onion con Buffer 128MB.