UNIVERSIDAD NACIONAL DEL SUR

TESIS DE GRADO

Integración de LibreMesh para Incentived Mesh Networking

Autor:
Luciano BRUNA

Directores:
Dr. Claudio DELRIEUX
Dr. Eduardo PAOLINI
Dr. Carlos MATRÁNGOLO

Departamento de Ingeniería Eléctrica y Computadoras

6 de noviembre de 2018

«Cuando cambiamos la forma de ver las cosas, las cosas cambian (aca iria una frase como la gente, afín al trabajo y con la que me identifique, si es que llegase a encontrar una, sino esta pagina se omite).»

Wayne Dyer

UNIVERSIDAD NACIONAL DEL SUR

Resumen

Universidad Nacional del Sur
Departamento de Ingeniería Eléctrica y Computadoras

Ingeniero Electrónico

Integración de LibreMesh para Incentived Mesh Networking

por Luciano BRUNA

Acá iría el abstracto, un resumen de todo lo que hice. Podria tener un poquito de narrativa tensional tal y como menciono claudio. "se vio ante tal o cual eventualidad, las redes mesh son la posible solucion. Investigando redes mesh e incursionando en el termino incentived mesh, se detecto que althea y libremesh podrian integrarse y mejorar asi el avance de ambos grupos."Pero no mucho, la idea es que aca en el abstracto va todo mas resumido y serio. En cambio, mas adelante en la introduccion iria mas narrativa que atrape al lector no experto en el tema.

Agradecimientos

Esto para mi es muy importante, no solo agradecer a mis directores y advisors sino tambien a Nicolas Pace, Althea. Aca irian los agradecimientos

Agradezco a Pampa Energía, esto no sería posible de no ser por ellos

Índice general

Índice de figuras

Índice de cuadros

List of Abbreviations

SSID	**S**ervice **S**et **I**dentifier
BATMAN	**B**etter **A**pproach **T**o **M**obile **A**d-hoc **N**etworks
WCN	**W**ireless **C**ommunity **N**etworks
MANET	**M**obile **A**d-Hoc **NET**works
MPR	**M**ulti **P**oint **R**elay
OLSR	**O**ptimized **L**ink **S**tate **R**outing

Dedicado a Diana Sanchez...

Capítulo 1

Introduccion

Durante los últimos años, las redes mesh [1] han pasado de ser un concepto teórico de red a convertirse en dispositivos comercialmente disponibles que prometen crear redes distribuídas, capaces de crear links entre sí y mantenerlos autónomamente. El IEEE creó el grupo de trabajo 802.11s, que define cómo los dispositivos inalámbricos pueden interconectarse para crear una red mesh inalámbrica [1]. Esto tiene la intención de guiar a institutos de investigación y a la industria para desarrollar un estándar para redes mesh, con el objetivo de proveer interoperabilidad entre todos los dispositivos. Mientras el estándar 802.11 sigue bajo desarrollo, hay un número de protocolos de red que están actualmente disponibles.

De todas las aplicaciones de redes mesh, en este trabajo se destacan las redes mesh comunitarias no cableadas, o mesh WCNs (Mesh Wireless Comunity Networks), cuya topología está representada en la figura (1.1). Estas han sido desarrolladas como movimientos de entusiastas de redes no cabledas, quienes usando equipamiento de bajo costo para una interconexión gratuita, han creado redes completamente autónomas. Su aparición se debe principalmente a la búsqueda de estos grupos de lograr un servicio equivalente a los ofrecidos por redes 2.5G y 3G pero de manera gratuita[2].

Las razones para participar en una red mesh son muy variadas. Aquellas personas que creen que la conectividad de banda ancha debería ser libre y gratuita y que las barreras impuestas por los oligopolios de los ISPs deben ser eliminadas usualmente son los primeros en unirse a WCNs. Esto último se agrava, al existir muchas zonas en las que solo hay un ISP disponible [3]. En el cuadro (1.1), se reportan algunas de las mesh WCNs más significativas a nivel mundial categorizadas según el año en que han sido diseñadas y la cantidad de nodos. Cabe destacar que todas ellas parten de una iniciativa comunitaria, es decir que son el resultado de efuerzos colectivos de voluntarios individuales y funcionan sin fines de lucro.

En el estudio de este fenómeno, se destacó la iniciativa de la empresa Althea, con la cual se desarrolló este trabajo en conjunto. Althea propone un modelo de negocios basado en el incentivized mesh [2]. Y aca explicar un poquito brevemente qué hace althea (basicamente copipastear la primera parte de su white paper sin entrar en detalles tecnicos de que tipos de nodos tienen, etc.).

En este parrafo explicar un poco la problematica y el hecho de que me di cuenta (hablar siempre en voz pasiva xq sino me agarra toc) que podría integrar las plataformas althea con libremesh (y explicar qué es libremesh usando el pie de página,

[1] Una red mesh es una topología de red local en la que la infraestructura de nodos (routers, Access Points y otros dispositivos) se conectan directamente, dinámicamente y sin jerarquía a tantos otros nodos como sea posible cooperando entre todos para hacer ruteo de la forma más eficiente posible.

[2] El incentivized mesh es un modelo económico que motiva a las comunidades a construir infraestructura robusta e independiente de ISPs utilizando crypto monedas como pago de incentivo por hacerlo.

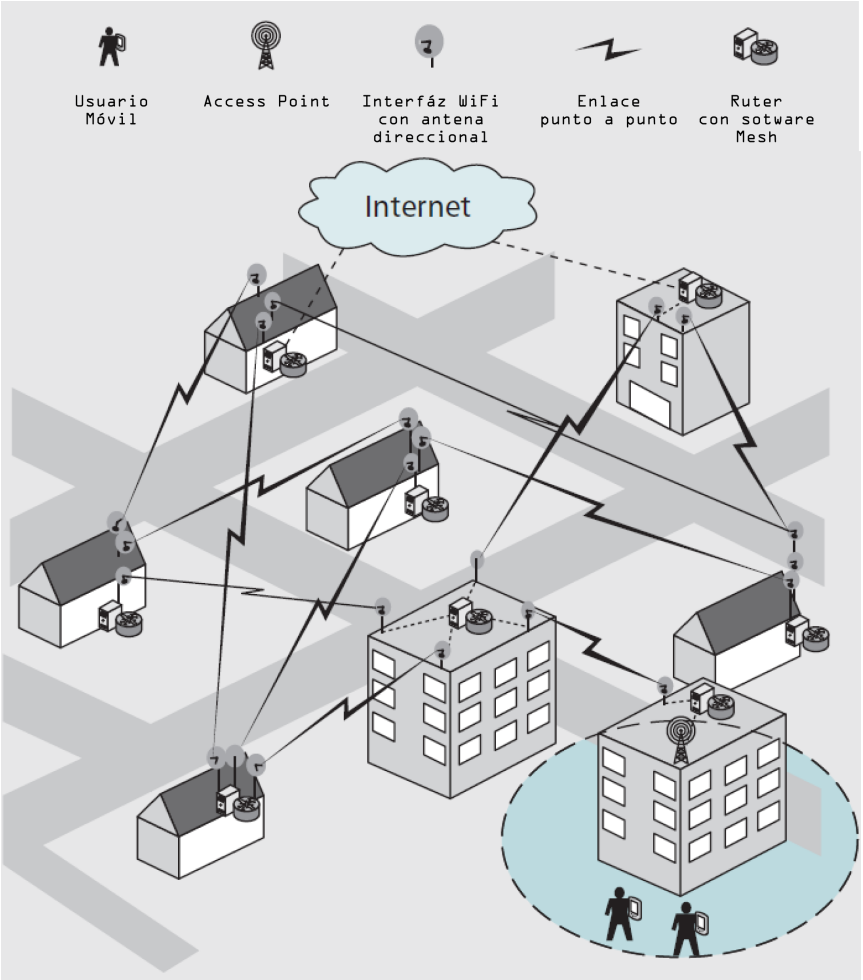

FIGURA 1.1: *Topología de redes mesh WCNs.*

no hace falta mas en esta instancia ya que lo explicare por completo en la seccion del marco teorico).

En este parrafo explicar brevemente (pero no tan breve como en el resumen) todo lo que hice en el trabajo y las cosas que quedan como propuesta para seguir trabajando en el futuro.

Y en este parrafo poner -en la seccion 1- se vera tal cosa. En el -capitulo 2- tal otra cosa y listo. Es una manera util de ocupar espacio (mas de la mitad de los papers que lei lo hacen asi que yo tambien lo hare).

CUADRO 1.1: Comunidades de redes mesh WCNs en todo el mundo.

Nombre de la Red	**Ubicación**	**Año de Fundación**	**Cantidad de nodos**
SeattleWireless	Seattle, WA, Estados Unidos	2000	80
AWMN	Athena, Grecia	2002	2473
CUWiN	Urbana, IL, Estados Unidos	2002	48
Berlin's Freifunk	Berlin, Alemania	2002	316
Wireless Leiden	Leiden, Países Bajos	2002	73
NetEquality Roofnet	Portland, OR, Estados Unidos	2007	126
NYCwireless	New York, Estados Unidos	2001	145

Capítulo 2

Marco Teórico

Este capítulo está pensado para dar el *background* necesario para entender a las redes mesh y algunos algoritmos de ruteo específicos. Estos han sido partícipes de discusiones por parte de las entidades protagonistas de este proyecto. Además se incursiona en temas como criptografía y ..

Mencionar aca o en algun lado que esto es muy importante para entender por que libremesh usa los protocolos que usa y por que althea usa lo que usa, etc.

2.1. Definiciones Iniciales

2.1.1. Redes Ad Hoc y Redes Mesh

En gran parte de la bibliografía consultada, se han utilizado los términos redes ad hoc y redes mesh de manera indistinta. Si bien esto no representa ningún error en esos casos, puede generar confusiones si no se conoce la diferencia entre ellos. Por eso es importante destacarla.

Una red Ad hoc implica el compromiso cooperativo de un conjunto de nodos wireless sin ningún tipo de intervención por parte de algún access point centralizado o de alguna infraestructura existente. Este tipo de red tiene características claves de ser auto-generativas y auto-curativas y no depender de ningún servicio centralizado de ningún nodo en particular. Una red Ad-hoc puede ser wireless, en cuyo caso sus dispositivos, tales como laptops o sensores, realizan una función de ruteo para enviarse datos creando una forma arbitraria de topología de red. Cuando todos los dispositivos son móviles, se habla de redes MANET [1], que pueden cambiar de topología muy rápidamente y de forma impredecible. Redes de sensores no cableados son un buen ejemplo de redes MANET.

Una red wireless mesh se caracteriza por tener routers estáticos o cuasi-estáticos dedicados que tienen el trabajo de realizar la función de ruteo de paquetes a lo largo de la red. Además, están los dispositivos clientes, quienes no tienen funcionalidad de ruteo. Estos están conectados a los ruters wireless.

Existen muchos protocolos de ruteo compatibles con estos tipos de redes. La gran mayoría de ellos parte de grupos académicos que buscan optimizar diferentes cuestiones como la velocidad de convergencia o la evitación de lazos de ruteo. A estos se les prestará una mayor atención en la sección (2.2). Además, el grupo de trabajo 802.11s de la IEEE define un estándar para las redes mesh en donde se define, entre otras cosas, el método de ruteo para este tipo de redes.

[1] MANET: Abreviación de Mobile Ad-Hoc Network.

2.2. Protocolos de Ruteo

Los protocolos para redes mesh se clasifican en tres categorías: *reactivos*, *proactivos* e *hibridos* [4]. Los protocolos reactivos solo buscan un camino entre nodos cuando hay datos para ser enviados. Este método tiene la ventaja de no gastar ancho de banda de la red con mensajes de control cuando la transmisión de datos no es requerida. Los protocolos reactivos idealmente se utilizan en redes Ad Hoc donde con los nodos móviles los caminos de datos podrían cambiar contínuamente.

Por otro lado, los protocolos proactivos actívamente establecen y mantienen caminos de datos para nodos tanto si los datos necesitan ser enviados o no. Esto permite una latencia menor a la hora de enviar datos a través de la red ya que el camino óptimo ya es conocido. De todas maneras, esto viene con un costo computacional mayor y con la necesidad de una administración de red mucho más compleja.

Los protocolos híbridos exhiben las propiedades de tanto los protocolos proactivos y reactivos. En general, intentan usar características proactivas o reactivas según el escenario, lo cual explota sus fortalezas y es por eso que pueden lograr un mayor nivel de escalabilidad. En general, son más complejos en comportamiento, lo cual hace que sean más complejos de implementar que los protocolos puramente proactivos o reactivos.

En (2.2.1) se explicará el algoritmo *Vector Distancia* que es la base de varios protocolos de ruteo muy conocidos como RIP y OLSR. Se pondrá en discusión la característica principal y la problemática de este último en (2.2.2). Luego se detallarán protocolos más eficientes para redes mesh en las secciones (2.2.3), (2.2.5) y (2.2.6). Finalmente, en (2.2.7) se compararán todos los protocolos previamente descritos.

2.2.1. Algoritmo Vector-Distancia

El algoritmo *Vector Distancia* se trata de uno de los más importantes junto con el *estado de enlace* [2]. Se basa en el algoritmo de Bellman-Ford para calcular las rutas. Requiere que un router informe a sus vecinos de los cambios en la topología periódicamente y en algunos casos cuando se detecta un cambio en la topología de la red.

Se basa en calcular la dirección y la distancia hasta cualquier enlace en la red. El costo de alcanzar un destino se lleva a cabo usando cálculos matemáticos como la métrica del camino. Una de las métricas más utilizadas es el número de *hops* [3].

Los cambios son detectados periódicamente ya que la tabla de enrutamiento de cada router se envía a todos los vecinos que usan el mismo protocolo. Una vez que el router tiene toda la información, actualiza su tabla e informa a sus vecinos de los mismos. Este proceso se conoce también como *enrutamiento por rumor* ya que los nodos utilizan la información de sus vecinos y no pueden comprobar a ciencia cierta si ésta es verdadera o no.

El algoritmo de Bellman-Ford se adapta perfectamente al modo de aprendizaje de los nodos que "nacen", es decir, cuando se conectan a la red. A medida que el algoritmo progresa, el nuevo nodo va adquiriendo más información sobre el resto de nodos de la red. Este algoritmo converge rápidamente cuando se conectan nuevos nodos. Por ello se suele decir que las buenas noticias viajan rápido por la red.

[2] Estado de enlace es un algoritmo de ruteo alternativo al de *Vector Distancia*. Fue implementado por primera vez en 1979 por ARPANET.

[3] En una red, se le dice *hop* a una porción de camino entre una fuente y su destino. La cuenta de *hops* se refiere al número de dispositivos intermediarios a través de los cuales deben pasar los datos.

Un problema del que padece este algoritmo es el de la transmisión de malas noticias por la red tales como la ruptura de un enlace o la desaparición de un nodo. Este algoritmo converge lentamente en estos casos. Aunque el principal inconveniente de este algoritmo es el de la cuenta a infinito. Para ilustrarlo, se puede tomar como ejemplo el de la figura (2.1).

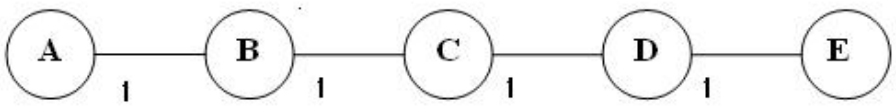

FIGURA 2.1: *Topología de red para ejemplificar la limitación del protocolo vector distancia.*

- Inicialmente A está desactivado. Cuando A se activa, B se entera de que A existe al recibir su vector distancia y actualizar su tabla indicando que A dista 1.
- El nodo C se entera de que A existe porque B le indica que tiene un enlace hacia A de coste 1. Entonces C actualiza su tabla registrando una trayectoria hacia A de coste 2.
- Si el nodo A se desconecta entonces B no recibe el VD de A. Sin embargo el nodo C le dice que tiene una trayectoria hasta A de distancia 2. B no sabe que la trayectoria de C a A pasa por el mismo y por tanto cree que puede llegar a A a través de C por lo que actualiza su tabla registrando la distancia 2 + 1 = 3 hasta A.
- En el siguiente intercambio, el nodo C comprueba que sus vecinos B y D tienen una trayectoria hasta A de distancia 3. C calcula su propia distancia hasta A en 3 + 1 = 4. En los siguientes intercambios, los nodos elevan ilimitadamente su distancia a A (cuenta a infinito).

Mientras no se interrumpa la cuenta a infinito, el algoritmo no converge. Aunque se han propuesto diversas soluciones a este problema. El protocolo RIP establece el siguiente criterio:

$$\infty = 16.$$

Es decir, cuando la cantidad de incrementos de distancia supera los 15 en un período de tiempo relativamente corto, RIP considera que se está en un loop infinito y corta las iteraciones. El protocolo OLSR utiliza un acercamiento más eficiente, como se verá en (2.2.2).

2.2.2. Protocolo OLSR

El protocolo de ruteo Optimized Link State Routing (o OLSR) emplea el algoritmo de *vector distancia* pero tiene muchos menos conteos a ínfinitos que el protocolo RIP. Para mantener informados a los nodos de la red de los cambios topológicos

de la misma, utiliza el mecanimo Multi-Point Relay (o MPR) para hacer flooding [4]. A diferencia del flooding convencional, el método MPR evita que todos los nodos vecinos retransmitan los paquetes. Todos los nodos de la red seleccionan entre sus vecinos un conjunto de retransmisores (o *multi point relays*) encargados de retransmitir los mensajes que envía el nodo inicial. Los demas vecinos del nodo no pueden retransmitir, lo cual reduce significativamente el tráfico generado respecto al flooding tradicional.

Hay varios criterios para elegir los MPRs de un nodo. En general, se considera que el conjunto de MPRs de un nodo debe verificar que son capaces de alcanzar a todos los vecinos situados a 2 *hops* del nodo que los define como MPR.

EL protocolo OLSR es un protocolo muy usado pero como fue desarrollado en los inicios del 2000 no está pensado para rutear información en redes mesh. De hecho, hay multiples fuentes que demuestran que OLSR sufre de muchos conteos a infinito en redes mesh [5] y que su performance con respecto a otros protocolos de ruteo como los que se presentarán en (2.2.3), (2.2.5) y (2.2.6) es mucho menor [4].

2.2.3. Protocolo BatMan

El protocolo BatMan (o Better Aproach to Mobile Ad-hoc Network) surge como la solución a todos los loops de ruteo que ocurren en una red mesh que utilice OLSR. Es un protocolo pro-activo ya que hace que periódicamente todos los nodos de la red envíen *hello packets* [5], a sus vecinos. Cada uno de estos mensajes tienen tres partes:

- Dirección de recepción
- Dirección de Remitente
- Secuencia única de números

Cada vecino que recibe el paquete, sustituye la dirección de remitente por su propia dirección. Luego, realiza un $re - broadcasting$ [6] del mensaje. Cada vez que un nodo recibe un mensaje cuya secuencia única de números fue generada por si mismo, verifica que ese mensaje fue emitido por él anteriormente. Esto le permite saber si la conexión con otros nodos es o no bi-direccional.

Para explicar el algoritmo con el que trabaja BatMan se modela a la red como $G = (N, E)$, donde N representa un set de nodos y E representa un set de links entre par de nodos. Para cada nodo $i \epsilon N$, existe un set de vecinos a 1-hop de distancia, K. El mensaje desde la fuente $s \epsilon N$ a un destino d es transmitido sobre el link $(s, d) \epsilon E$ si d es también un elemento de K. Si no fuera el caso, se transmitiría sobre una ruta de múltiples hops hecha por un link (s, i) y una ruta $[i, d]$, donde i es un nodo dentro de K y (s, i) es un link en E. La ruta $[i, d]$ representa una ruta del nodo i al nodo d a travéz de la *subnet* $S = (N - \{s\}, A - \{(s, i) : i \epsilon K\})$.

Se puede separar al algoritmo en cuatro pasos:

1. Considerar el mensaje de ruteo m desde el nodo s al nodo d en la red G. Eliminar todos los links $(s, i) \forall i \neq K$ para reducir el grafo.

[4] Se le dice *flooding* a la acción de enviar un paquete de actualización de métricas a *todos* los nodos de la red. Cuando un nodo desea hacer *flooding* primero le envía el paquete deseado a todos sus nodos vecinos. Luego, estos retransmiten el mensaje hasta que éste llegue a todos los nodos de la red.

[5] Se le dice *hello packet* a una corta cadena de texto que un nodo de una red le envía a otro para informarle de su existencia.

[6] En una red mesh, Broadcasting es la transmisión de un cierto mensaje a *todos* los nodos vecinos.

2. Asociar cada link con el peso w_{si} donde w_{si} es el número de *hello packets* recividos desde el destino a travéz del nodo i durante un período de tiempo reciente.

3. Encontrar el link con el peso w_{si} más grande en el sub-grafo y enviar m a travéz del link (s, i).

4. Si $i \neq d$ entonces repetir los pasos de 1 a 4 para rutear el mensaje desde i hasta d en el sub-grafo S.

Las figuras (2.2), (2.3) y (2.4) ilustran el funcionamiento del algoritmo en el siguiente escenario:

- Nodo 1 quiere enviar un mensaje al nodo 6. Solo considera este sets de links $\{(1,2),(1,3),(1,4)\}$ a sus vecinos $\{2,3,4\}$. Esto se aprecia en la figura (2.3).
- Determinar el mejor link como el link con el mayor número de *hello packets* recividos desde el nodo 6.
- Suponiendo que $(1,2)$ es el mejor link entonces enviar el mensaje a travéz de dicho link.
- Como el nodo 2 no es el destino, reducir el grafo N al grafo S y repetir los pasos del algoritmo de 1 a 4. Esto se aprecia en la figura (2.4).
- Nodo 2 solo considera este sets de links $\{(2,3),(2,5)\}$ hacia sus vecinos $\{3,5\}$.
- Determinar el mejor link como el link con el mayor número de *hello packets* recividos desde el nodo 6.
- Suponiendo que $(2,5)$ es el mejor link entonces enviar el mensaje a travéz de dicho link.
- Como el nodo 5 no es el destino, reducir el grafo N al grafo S y repetir los pasos del algoritmo de 1 a 4.
- Nodo 5 solo considera este sets de links $\{(5,6),(5,3)\}$ hacia sus vecinos $\{6,3\}$.
- Determinar el mejor link como el link con el mayor número de *hello packets* recividos desde el nodo 6.
- Suponiendo que $(5,6)$ es el mejor link entonces enviar el mensaje a travéz de dicho link.
- Nodo 6 es el destino.

Como se puede ver, usando el protocolo BatMan no se guarda la ruta completa a ningún destino. Cada nodo sobre una ruta solo mantiene información sobre el siguiente link a travéz del cual se puede encontrar la mejor ruta. Resulta curioso mencionar la similitud que tiene esta filosofía con un comportamiento particular de las termitas llamado *estigmergia* [7]. Las termitas dejan caminos de feromonas que otras termitas pueden sentir para permitirles saber en qué dirección se puede encontrar la

[7] Estigmergia significa colaboración a través del medio físico. Fue introducido por Pierre-Paul Grassé, un estudioso de las hormigas, para explicar cómo se lograban realizar las tareas en insectos sociales sin necesidad de planificación ni de un poder central. Se ha usado como inspiración en algoritmos de ruteo como *AntHocNet* [6]. BatMan también comparte muchas similitudes con su filosofía.

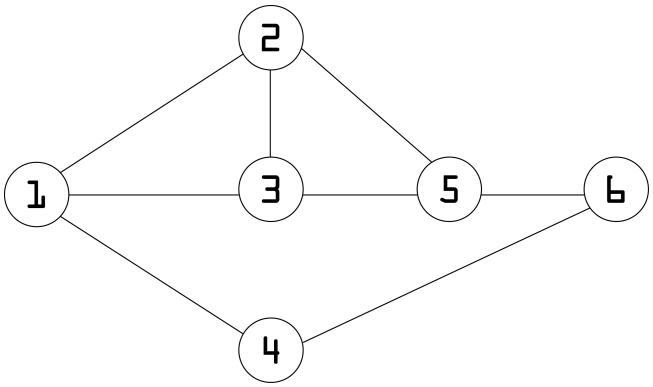

FIGURA 2.2: *Grafo inicial G.*

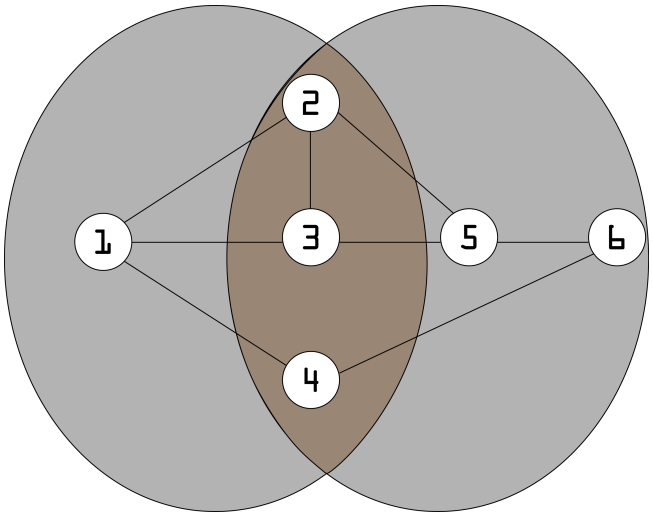

FIGURA 2.3: *Subconjunto de nodos formados por el algoritmo de BatMan luego de la primer iteración. Muestra la relación entre los tres subconjuntos que están referidos en el algoritmo.*

mejor ruta hacia la comida o hacia el nido. Pero las termitas nunca saben el camino completo hacia ninguno de ellos.

BMX (o BatMan-eXperimental) es un protocolo de ruteo para redes mesh hecho para sistemas operativos basados en linux. Comenzó siendo una rama del protocolo BatMan para diseñada para implementar y probar nuevas características y conceptos para superar ciertas limitaciones en el algoritmo de ruteo. Pero con el tiempo fue tomando una dirección totalmente diferente de manera que la re-integración se hizo imposible. Hoy en día, es un proyecto independiente que tiene su propia subsección en este trabajo (2.2.4).

Batman es uno de los protocolos más utilizados en redes mesh y ha sido un punto de partida para diversas investigaciones, no necesariamente limitadas a aplicaciones comunitaras [7].

2.2.4. Protocolo BMX

2.2.5. Protocolo Babel

El protocolo Babel fue creado para reducir al mínimo la cantidad de loops infinitos que se generan en las redes mesh. Conceptualmente, es como el protocolo RIP

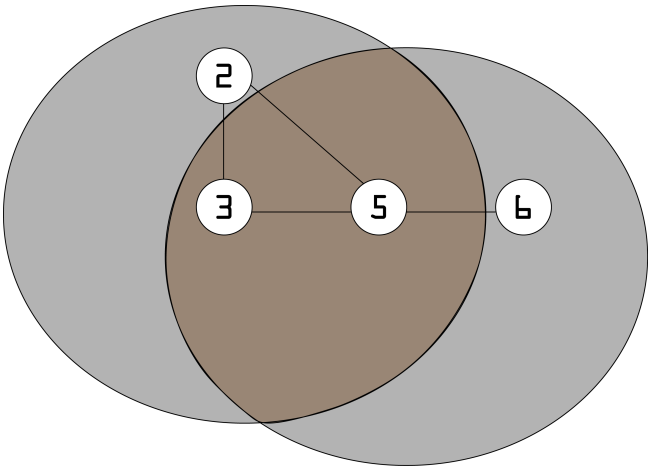

FIGURA 2.4: *Subconjunto de nodos formados por el algoritmo de BatMan luego de la segunda iteración.*

pero incluye una serie de refinamientos que previenen la formación de loops o al menos hacen que un loop generado desaparezca rápido y no se forme de nuevo [8].

La filosofía de este protocolo radica en saber que en una red mesh siempre van a existir constantes desconexiones y reconexiones. Entonces, no tiene sentido hacer un protocolo que converja muy rápido ya que es esto último lo que genera loops. Babel solo acepta actualizaciones provenientes de nodos que sean $fehacientes$. El concepto de $feasibilidad$ se ha utilizado en muchos protocolos, teniendo cada uno de ellos su propio método para verificarla.

En el caso de Babel, se considera $fehaciente$ a un nodo cuya ruta enviada tenga una métrica [8] menor estricta a las que siempre se tuvieron almacenadas. Cada vez que un nodo envía una actualización a otro se guarda el valor de la métrica y se mantiene un registro actualizado de la métrica más pequeña que jamás se haya enviado para esa dirección. Entonces, solo se acepta una ruta nueva si y solo si tiene una métrica mejor que el valor más chico que jamás se haya enviado. Con esto se evitan los loops fácilmente.

En la figura (2.5) se ilustra una topología de red similar a la analizada en (2.2.1) en la que en caso de apagarse el nodo A, podría generarse un loop infinito si se usase el algoritmo vector-distancia sin refinar. En cambio, al usar Babel, en el momento en que A se apaga y C le informa a B de una nueva métrica hacia el nodo A, B *nolecree* ya que esa nueva métrica es mayor a la anterior. Y de esa manera, nunca se origina ningún loop.

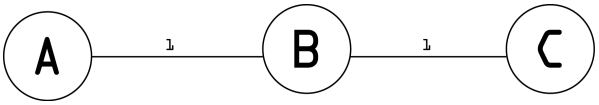

FIGURA 2.5: *Red mesh sencilla que podría generar loops infinitos con RIP pero no con Babel.*

Sin embargo, Babel enfrenta una problemática llamada *starvation* [9]. Considérese una topología de red como la ilustrada en la figura (2.6) donde todas las métricas son

[8] La métrica de una red es el valor que se utiliza para medir la calidad de las rutas. En general, se utiliza el número de hops entre el nodo origen y nodo destino.

[9] El término *starvation* o hambruna surge de la analogía entre una red mesh y un festín. Cuando un nodo tiene información sobre otro nodo pero no puede aceptarla por restricciones del protocolo, se dice que es como estar frente a una mesa llena de comida y no poder probarla.

iguales a 1. Supóngase que la conexión entre los nodos A y S se rompe, quedando una nueva topología de red ilustrada en (2.7). A partir de ese momento, B enviará una actualización a A informando una nueva métrica hacia S de valor 2. Como esta nueva métrica es mayor a la anterior, no será aceptada por A, evitando así un loop. Pero de esta manera A y S quedarán incomunicados indefinidamente.

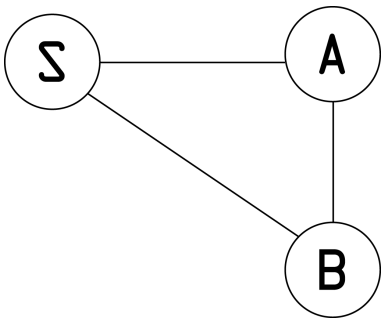

FIGURA 2.6: *Caso inicial de red mesh que utiliza Babel.*

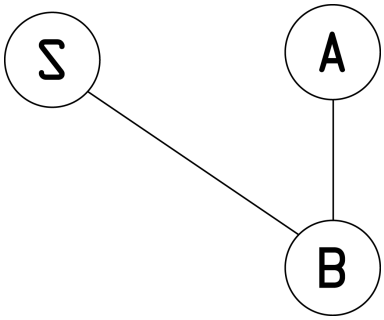

FIGURA 2.7: *Caso final de una red mesh que utiliza Babel. Se puede apreciar la starvation.*

Para evitar la *starvation* se pueden implementar diversas soluciones entre las que se destacan tres:

1. **Apagar toda la red y prenderla desde cero, reiniciando todas las memorias.**

   Esto es inviable en una red mesh ya que las desconexiones y reconexiones ocurren constántemente.

2. **Hacer una sincronización global y olvidar todos los previos valores de** $feasibilidad$ **con el nodo cuya conexión de mejor métrica se haya roto.**

   El problema de esta implementación es que el software necesario para sincronizar a los nodos suele ser muy complejo. Además, si bien puede funcionar muy bien en redes cableadas, no es el caso en redes *wireless*.

3. **Usar rutas secuenciadas.**

   Esto consiste en enviar un dato extra además de la métrica en las actualizaciones de los nodos. Esto es lo que implementa Babel y es lo que le da el mejor tiempo de reparación de ruta frente a otros protocolos [4].

Normalmente, las actualizaciones que envían los nodos de una red contienen solo la métrica. Usar rutas secuenciadas implica enviar un dato extra en cada *announcement* como sigue:

$$s, d(B),$$

siendo s un *secuence number* y $d(B)$ la métrica. s es un número arbitrario generado únicamente por el nodo que envía el *announcement*. Él es el único que lo puede cambiar mientras el resto solo propaga el *announcement* manteniendo s fijo. De esta manera, se lo puede pensar como un reloj ya que solo es modificado por la fuente de la actualización y además, cada vez que cambia lo hace crecientemente (es decir, siempre aumenta).

A partir de esto, la función de Babel que define si un nodo es o no $fehaciente$ se ve a continuación:

$$\begin{gathered} s1, d1 \text{ es una ruta mejor que } s2, d2 \\ \text{si y solo si} \\ s1 > s2 \\ \text{ó} \\ s1 = s2 \text{ y } d1 \leq d2. \end{gathered}$$

Entonces, con esta definición se evita el *starving* en la mayoría de los casos. Para el ejemplo de la figura (2.7) se puede ver que cuando el link entre S y A se rompe, solo hay que esperar a que S incremente su *secuence number*. De esa manera, A aceptará una actualización proveniente de B por más que la nueva métrica sea mayor que la anterior. Para este sencillo ejemplo, esto tomará una o dos iteraciones.

Pero en el caso de redes mucho más grandes en términos de cantidad de nodos, existe una *starvation* temporal ya que el *secuence number* puede tardar muchas iteraciones en llegar. Algunos protocolos como DSDV incrementan s en períodos muy cortos y se dice que eso no suele funcionar muy bien [9].

Babel incorpora la capacidad de enviar un *request*. Cada vez que un nodo sufre de *starvation*, puede enviarle a un vecino un *request* en el que indica su estado. De esta manera, el resto de los nodos sabe que debe incrementar su número s para reparar una ruta rota. Así se decrementa considerablemente el tiempo de reparación de ruta.

Babel es un protocolo "100 % loop-free" en el caso de que solo haya un *gateway*. En el caso de haber más de uno, puede tener problemas en casos muy patológicos como que haya un *crash* en varios de ellos al mismo tiempo. Aún así, el mecanismo de feasibilidad lo resuelve relativamente rápido.

2.2.6. Protocolo SSR

Scalable Source Routing (SSR) surge como una solución al problema de escalabilidad que enfrentan otros protocolos [10]. Es un protocolo híbrido, lo cual le hace consumir mucho menos ancho de banda cuando se lleva a cabo el ruteo. Hay dos consideraciones que definen a SSR:

1. Cada nodo mantiene un *cache* que guarda rutas fuentes usadas [10] recientemente. El *cache* se guarda de forma binaria con lo que ocupa poco espacio.

2. Cuando un nodo necesita enviar un paquete a un destino que no tiene guardado, esa ruta se construye iterativamente.

[10] El término "source routing" o ruteo desde la fuente es un sistema en el cual el paquete sale desde el emisor con la "hoja de ruta"que le indica por qué nodos debe pasar para llegar al destino. El primer problema es que como cada una elije su camino, puede ocurrir que muchos seleccionen los mismos tramos y haya sobrecarga o congestionamiento en algunos links y otros permancezcan ociosos.

2.2.7. Comparaciones

Aca podria copi-pastear el cuadro que uso el flaco de althea para su presentacion. El problema es que en ese cuadro hay 2 protocolos que no estudie. SSR porque no lo entendi y CJS... porque no estaba en ningun paper recomendado. Hay que achicar la tabla.

Aca tambien nombrar en algun momento que se puede demotrar que hay muchas brechas de seguridad (y nombrar el paper que dijo eso) y que para llegar a tener un protocolo seguro habria que definir y estandarizar un conjunto de criterios de protocolo seguro (y citar el paper que dijo eso). Pero que no se le va a dar mucha importancia en este trabajo porque para redes que no sean militares o de informacion fiscal no hace falta (citar al paper que dijo eso ultimo tambien) y yo voy a usar redes caseras.

2.2.8. Otros Protocolos

Esta subseccion podria no estar pero estaría bueno nombrar lo de los papers que hablan de la seguridad y de aplicaciones belicas. Porque sino me quedan esos papers sin mencionar.

Aca nombrar los protocolos SEAD [11] y SEMTOR [12] y referenciarlos, quizás en el pie de página explicando brevemente que son y ahi los referencio a la bibliografía.

2.3. Criptografía

La criptografía se define como el ámbito de la criptología que se ocupa de las técnicas de cifrado o codificado destinadas a alterar las representaciones lingüísticas de ciertos mensajes con el fin de hacerlos ininteligibles a receptores no autorizados. La aparición de la informática y el uso masivo de las comunicaciones digitales han producido un número creciente de problemas de seguridad. Las transacciones que se realizan a través de la red pueden ser interceptadas, y por tanto, la seguridad de esta información debe garantizarse.

Esto le ha dado a la criptografía una importancia muy grande en las últimas décadas, siendo una de las protagonistas principales en el ámbito del blockchain y las criptomonedas. Es por esto último que tiene un apartado propio en este trabajo.

Desde el año 1882 se habla de un método criptográfico *inhackeable* conocido como "libreta de un solo uso" [11]. Conceptualmente, es muy sencilla y se puede implementar con el uso de compuertas XOR. Considérese el caso de dos individuos: El remitente A y el receptor B, como se muestra en la figura (2.8). A desea enviar el mensaje m hacia B, donde m es el *string* "CS" que transformado al dominio binario es el número "1010011 1000011". Para encriptar el mensaje este método requiere de una clave k previamente compartida entre todos los participantes de la comunicación que debe ser tan arbitraria como sea posible. Para este ejemplo se supone $k = 11001000100110$. Entonces, el mensaje encriptado c [12] es la operación XOR entre m y k como se ve a continuación:

$$c = m \bigoplus k = 10100111000011 \bigoplus 11001000100110 = 01101111100101.$$

Finalmente, para que B decripte el mensaje, solo debe hacer la operación XOR entre c y k. Se verifica entonces que

[11] Hay muchos términos que aluden al mismo concepto que "libreta de un solo uso". Entre ellos están "cuaderno", "cifrado de Vernam" y su versión en inglés: "one-time pad".

[12] En criptografía se suele usar la letra c para referirse a un mensaje cifrado.

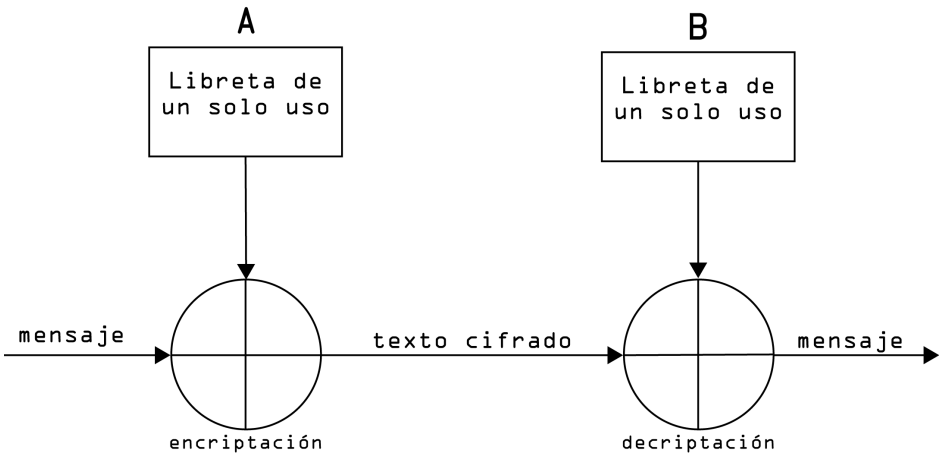

FIGURA 2.8: *Ejemplo de encriptación con el método Libreta de un solo uso.*

$$m = c \bigoplus k = 10100111000011.$$

Se ha demostrado que si k es suficientemente arbitrario, entonces este método es *inhackeable*. En la historia, se ha usado el concepto de Libreta de un solo uso en múltiples contextos, como el de la figura (2.9) que se corresponde con la maquina Enigma que usaban los nazis para encriptar sus mensajes en la Segunda Guerra Mundial [13]. El problema de este tipo de encriptación es que generar un k con suficiente aleatoriedad es complejo. Además, por definición k siempre tiene el mismo tamaño (o un tamaño mayor) que el mensaje. Esto es una limitación cuando se desea enviar mensajes relativamente largos.

FIGURA 2.9: *Máquina de encriptación de uso militar "Enigma". Fue usada a finales de los 1930 y durante la Segunda Guerra Mundial.*

Hoy en día, la criptografía enfrenta el desafío de lograr una buena seguridad en la comunicación manteniendo un tamaño de claves relativamente bajo. En las

[13] La maquina enigma tenía la desventaja de poseer un k con baja aleatoriedad y por ende, muy predictible.

subsecciones (2.3.2) y (2.3.3) se discutirá sobre las criptografías simétricas y no simétricas respectivamente. Pero antes, es imprescindible tener presente el concepto de Hashing (2.3.1).

2.3.1. El concepto de Hashing

Mencionar que se puede hacer de forma distribuida y hacer referencia al paper que habla del protocolo kademlia [13].

2.3.2. Criptografía Simétrica

La criptografía simétrica se basa en algoritmos que usan la misma clave tanto para encriptar como para desencriptar un mensaje [14]. Las claves pueden ser idénticas o tener una simple transformación matemática entre ellas. Un sistema que utilice este tipo de encriptación requiere que todas las partes tengan acceso a la clave antes de comenzar la comunicación.

Hay dos tipos de encriptación simétrica:

1. Cifrado por Streaming, con el que se ecriptan los digitos (tipicamente bytes) uno a la vez.

2. Cifrado por Bloque, con el que se toman un conjunto de bits y se los encripta como una única unidad.

Independientemente del tipo, es necesario que tanto el remitente como el receptor tengan la misma clave. Esto es un problema ya que es inseguro enviar la clave por un medio virtual pero puede resultar costoso enviarla por un medio físico. Además, la generación de claves se suele hacer con algoritmos de pseudoaletoriedad que a veces otorgan claves predecibles. La criptografía No Simétrica aparece como la solución a estos problemas (2.3.3).

2.3.3. Criptografía No Simétrica

También llamada criptografía de Clave Pública, la criptografía No Simétrica utiliza un par de claves: una clave pública que puede ser difundida ampliamente y una clave privada que es conocida solo por su dueño. Estas claves están relacionadas matemáticamente, con la suficiente complejidad para que no se pueda deducir una a partir de la otra.

En un sistema con este tipo de encriptación, cualquier individuo puede encriptar un mensaje utilizando la clave pública del receptor. Luego, el mensaje solo puede ser desencriptado utilizando la clave privada. Esto se ilustra en la figura (2.10) en la que el individuo B le envía un mensaje al individuo A. Con esto se soluciona el problema que tiene la criptografía Simétrica de tener que recurrir a un medio físico para compartir la clave para garantizar seguridad. Además, la sustentabilidad de este sistema radica en el alto costo computacional requerido para encontrar una clave privada a partir de su clave pública.

Normalmente, estos sistemas se apoyan en algoritmos basados en problemas matemáticos que actualmente no admiten una solución eficiente. Entre ellos se destacan los inherentes a la factorización de enteros, el logaritmo discreto y curvas elípticas.

[14] La "libreta de un solo uso" es de hecho, un tipo de criptografía simétrica que usa como método de encriptación y decriptación la operación XOR.

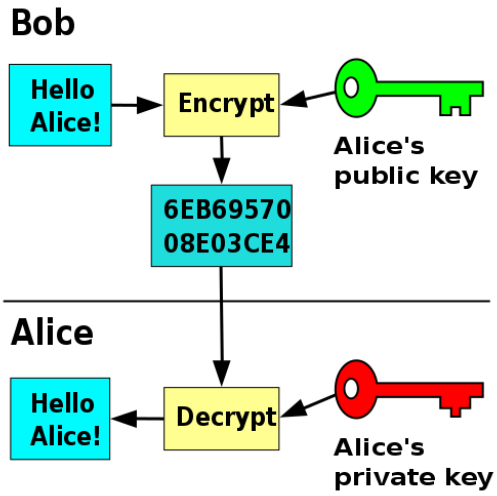

FIGURA 2.10: *Esquema que representa el envío de un mensaje desde el individuo B al individuo A utilizando un sistema de criptografía No Simétrica. (CAMBIAR LA FOTO PARA QUE QUEDE IGUAL AL DISEÑO DE LAS ANTERIORES FOTOS).*

Debido a esta complejidad, las comunicaciones son mucho más lentas que en la criptografía Simétrica. Es por esto, que se suele utilizar solo para pequeños bloques de información como cadenas de texto o para transmitir seguramente uns clave de sesión [15].

En un sistema de firma de clave pública, un individuo puede combinar un mensaje con una clave privada para crear una corta firma digital en el mensaje. Cualquier otro individuo que posea la correspondiente clave pública puede verificar que efectivamente fue firmado por el dueño de la clave privada. Esto se ilustra en la figura (2.11) en la que un individuo B firma un mensaje y un individuo A lo verifica. Cambiar el mensaje, incluso reemplazar solo un bit, causaría un fallo en la verificación. Esto logra un nivel de autenticidad que es clave en sistemas distribuídos como *blockchain* como se verá en la subsección (2.3.4).

El problema central que enfrenta este tipo de criptografía radica en la dificultad de probar si una clave pública es o no auténtica y si pertenece al individuo que la anuncia y no a una tercera parte maliciosa. En general, esto se soluciona por medio de una infraestructura de clave pública en la que una o varias terceras partes, conocidas como *autoridades certificadoras*, certifican la propiedad de un par de claves.

RSA

Elliptic curve Cryptography

2.3.4. Bitcoin

Aca tambien va lo de lighning network paper.

2.4. MOE Token Valuations

Cambiar titulo claramente por uno mas pochoclero

[15] Una clave de sesión es una clave simétrica de un solo uso utilizada para encriptar todos los mensajes en una sesión de comunicación.

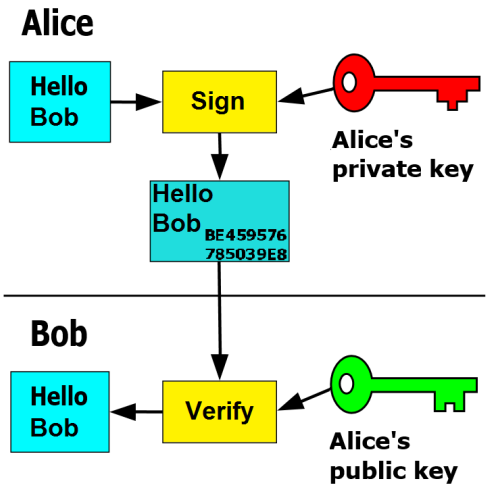

FIGURA 2.11: *Esquema que representa la firma de un mensaje del individuo B y su respectiva verificación por parte del individuo A utilizando un sistema de firma de clave pública. (CAMBIAR LA FOTO PARA QUE QUEDE IGUAL AL DISEÑO DE LAS ANTERIORES FOTOS).*

MOE = Medium of exchange (agregar a lista de abreviaciones)

En esta seccion iria lo de los ultimos 3 papers (puede que se añadan nuevas subsecciones.

Capítulo 3

Aplicaciones Reales de Redes Mesh

3.1. LibreMesh

3.2. Althea

3.3. Ammbr

Ammbr es como althea pero pinta mas a largo plazo, por ahi estaria bueno mencionar algo. y luego en el siguiente capitulo mencionar por que se decidio trabajar con althea en vez de ammbr. La razon es que althea esta mas avanzado y sería mas "directa"la integracion.

Capítulo 4

Trabajo Realizado

Esto probablemente tendrá muchas secciones y subsecciones que ire agregando conforme vaya haciendo cosas. Por lo pronto lo unico que tengo para poner aca es que hice una red de 4 ruters con libremesh y anda. Ponerle algunas screenshots y una foto pochoclera como la de la figura (4.1).

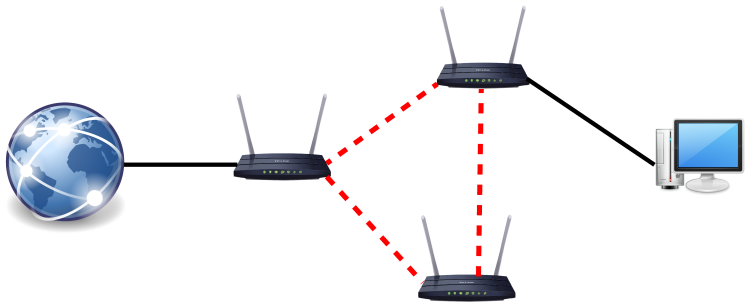

FIGURA 4.1: *Topología de primer red mesh realizada.*

Capítulo 5

Conclusiones

Apéndice A

Preguntas Frecuentes

A.1. Qué hace a un protocolo de ruteo mejor que otro?

blablabla... esto no se si va. Lo deje porque me gusto como se veia en el template. Pero probablemente el apendice vuele al carajo.

Bibliografía

[1] IEEE. «IEEE 802.11s». En: *IEEE Standars Association* (sep. de 2011). URL: https://es.wikipedia.org/wiki/IEEE_802.11s.

[2] Pantelis A. Frangoudis y George C. Polyzos. «Wireless Community Networks: An Alternative Approach for Nomadic Broadband Network Access». En: *IEEE* (mayo de 2011). URL: https://www.cs.columbia.edu/~vpk/papers/wcn.commag11.pdf.

[3] Federal Communications Comission. «Internet Access Services». En: *Industry Analysis and Technology Division Wireline Competition Bureau* (jun. de 2016). URL: https://www.fcc.gov/general/iatd-data-statistical-reports.

[4] J. C.-P. Wang M. Abolhasan B. Hagelstein. «Real-World Performance of Current Proactive Multi-hop Mesh Protocols». En: *IEEE APCC* (oct. de 2009).

[5] Corinna Aichele David Johnson Ntsibane Ntlatlapa. «A simple pragmatic approach to mesh routing using BATMAN». En: *2nd IFIP International Symposium on Wireless Communications and Information Technology in Developing Countries* (oct. de 2008).

[6] F. Ducatelle y L.M. Gambardella G. Di Caro. «AntHocNet: an ant-based hybrid routing algorithm for mobile ad hoc networks». En: *Proceedings of Parallel Problem Solving from Nature (PPSN VIII)* 3242 (jul. de 2004), págs. 461-470.

[7] Matthew Britton y Andrew Coyle. «Performance analysis of the B.A.T.M.A.N. wireless ad-hoc network routing protocol with mobility and directional antennas». En: *IEEE* (nov. de 2011). URL: https://ieeexplore.ieee.org/document/6470393.

[8] J. Chroboczek. «The Babel Routing Protocol». En: *Independent Submission* (abr. de 2011). URL: https://tools.ietf.org/html/rfc6126.

[9] Juliusz Chroboczek. «Babel Doesn't Care». En: *Conferencia BattleMesh* 8 (ago. de 2015). URL: https://youtu.be/1zMDLVln3XM.

[10] Thomas Fuhrmann y col. «Pushing Chord into the Underlay: Scalable Routing for Hybrid MANETs». En: *Fakultät für Informatik, Universität Karlsruhe* (jun. de 2006). URL: http://i30www.ira.uka.de/research/publications/p2p/.

[11] Yih-Chun Hu, David B. Johnson y Adrian Perrig. «SEAD: secure efficient distance vector routing for mobile wireless ad hoc networks». En: *Ad Hoc Networks* 1 (jun. de 2003). URL: http://computerscienceweb.com.

[12] Axel Neumann y col. «Securely-Entrusted Multi-Topology Routing for Community Networks». En: *Universitat Politècnica de Catalunya, Barcelona, España* (jun. de 2016).

[13] Petar Maymounkov y David Mazières. «Kademlia: A Peer-to-peer Information System Based on the XOR Metric». En: *Universidad de New York* (mar. de 2002). URL: http://kademlia.scs.cs.nyu.edu.