Bitcoin: Identificar nodos mineros

MISTIC: Máster Interuniversitario de Seguridad de las Tecnologías de la Información y de las Comunicaciones

Informe del trabajo de final de máster del Máster Interuniversitario de Seguridad de las Tecnologías de la Información y Comunicación presentado por Rubén Pérez Conte y dirigido por Cristina Pérez Solà.

Universitat Oberta de Catalunya, Sabadell, 2015

Abstract

En este trabajo inicialmente se ha realizado una investigación sobre Bitcoin con tal de obtener el conocimiento sobre sus componentes, interacción entre estos y el comportamiento y resultado del conjunto global. Posteriormente se ha buscado desarrollar una herramienta que permita detectar a potenciales usuarios “Mineros”. Esta herramienta consiste en scripts en python, sencillos de usar, que permiten buscar nodos en la red, identificar los accesibles y monitorizando su comportamiento determinar que nodos tienen mayor probabilidad de estar minando en la red a tiempo real.
Palabras clave: Bitcoin, red, bloque, minero.

In this work was initially carried out a research on about Bitcoin so to get knowledge about its components, interaction between them and the behavior and outcome of the overall set. Later it has sought to develop a tool to detect potential users ”Miners”. This tool consists of scripts in python, easy to use, allowing you to search nodes in the network, identify accessible nodes and monitoring their behavior to determine which nodes are more likely to be mining in the network in real time.
Keywords: Bitcoin, network, block, mining.

Agradecimientos

Me gustaría agradecer a los profesores del MISTIC de las diversas asignaturas cursadas que han respondido a las consultas surgidas durante la superación de estas.

En especial a Cristina Pérez Solà por su guía y supervisión en el desarrollo de este trabajo.

Índice

Lista de figuras

CAPÍTULO 1

Introducción

1.1 Motivación

Bitcoin es una de las primeras criptodivisas implementadas que han tenido éxito y la que más relevancia ha obtenido llegando a ser adoptada alrededor del mundo por miles de personas y grandes empresas. En los últimos años la criptodivisa ha ganado notoriedad mediática debido a sus características (potencialmente anónimo, exento de tasas y sin una autoridad de control), a sus fluctuaciones en el mercado y a su posible uso en negocios criminales. Estos aspectos junto al valor de cambio que se le da respecto otras divisas, más de 200 euros/Bitcoin, convierte al sistema que mantiene a esta criptodivisa en un objetivo interesante a ser estudiado desde multitud de puntos.

Bitcoin es un sistema complejo compuesto por multitud de componentes que permiten su utilización como un sistema de pago: protocolos de comunicación, software para su utilización, usuarios de la divisa que le dan valor y como no la divisa en sí entre otros. La divisa Bitcoin (BTC) a la que se le proporciona el valor respecto al Euro o al Dolar tiene la característica de emitirse cierta cantidad periódicamente por ciertos usuarios especiales de Bitcoin a los que se les llama "Mineros", proceso equiparable a la impresión de billetes y monedas de los bancos de los países. El número, o cantidad de unidades de la divisa Bitcoin, aún estando limitado por el sistema a 21 millones de unidades actualmente en el sistema no hay esta cantidad, sino una menor. Son los mineros los encargados expedir unidades e incluirlas en el sistema y que los usuarios puedan acumular y usar esta divisa sin que el sistema se vea afectado. Esto se ha demostrado difícil debido a que los usuarios también tienen un gran peso en el sistema, prueba de ello son sus grandes fluctuaciones en su valor respecto al resto de divisas [6].

Aun con todo, el simple hecho de que los mineros son los encargados de expedir las divisas, ya demuestra que son una pieza clave del sistema. Estas nuevas divisas son expedidas en una

cantidad variable en el tiempo (inicialmente 50 y actualmente 25, cada 4 años descenderá a la mitad,) "a nombre" del minero en cuestión [20]. Cualquier usuario que cumpla con los requisitos y exigencias puede convertirse en un minero. Este proceso de expedir cierta cantidad de divisas a nombre de un minero, que a partir de ahora nos referiremos a él como minar bloques, es un proceso vital para el sistema y no únicamente por aumentar la cantidad total disponible para ser usada. El proceso de minar bloques legítima intercambios de divisas entre usuarios (transacciones) al incluirlos en unas estructuras de datos, llamados bloques, y que el intercambio sea efectivo. Si bien el hecho de contribuir al sistema legitimando las transacciones no es un motivo suficiente para los usuarios, debido a que supone actualmente un alto coste de procesamiento, el hecho de poder expedir a nombre de uno mismo cierta cantidad de divisas del sistema si lo es para hacer participar a los usuarios.

Únicamente con lo obtenido con minar un bloque, 25 BTC, supone una ganancia de más de 5.000 EUR, cantidad suficiente para motivar su intento y generar una gran competitividad entre ellos. Esta competitividad puede llevar a realizar acciones de distinta índole desde la aparición de hardware cada vez más eficiente hasta otras dañinas contra estos. En cualquier caso poder identificar los mineros en la red puede abrir posibilidades que van desde la supervisión hasta el aprovechamiento egoísta de estos.

1.2 Objetivos

Este trabajo de final de máster dispone de un objetivo general dispuesto junto a la propuesta que es la identificación de los nodos mineros en la red bitcoin y como secundarios obtener información sobre ellos así como detectar comportamientos anómalos. En base a esta propuesta y por el estado de conocimiento en la materia se han expandido en:

1. Objetivos de obtención de conocimientos: Con estos objetivos se pretenderá lograr el conocimiento base necesario para abordar el resto.
    - Estudio y documentación de la historia y fenómeno Bitcoin.
    - Estudio de los elementos de Bitcoin (clientes, blockchain, mineros, etc.).
    - Realización de experimentos relacionados con el uso y minado.
2. Objetivo principal: Como objetivo principal se encuentra el diseño e implementación de una aplicación capaz de detectar los nodos mineros en la red bitcoin. Esta aplicación deberá buscar los nodos y determinar cuáles de ellos son mineros en base a ciertos criterios. Posteriormente se procederá a evaluar los resultados que esta aplicación provee.
3. Objetivos secundarios: Se perseguirá como meta idílica, una vez completada la aplicación de detección, los siguientes objetivos:
    - Modificación de la aplicación principal para añadir la funcionalidad de obtención de información de los nodos detectados como mineros.
    - Modificación de la aplicación principal para añadir la funcionalidad de comportamientos anómalos dentro de la red.

1.3 Planificación Inicial

La planificación inicial del trabajo ha sido realizada mediante Microsoft Project obteniéndose un diagrama de Gantt. Este diagrama de Gantt contiene los apartados principales que corresponden a los objetivos que en el inicio del TFM fueron pensados, tanto el principal como los secundarios por lo que se han forzado las tareas a entrar en el plazo de tiempo disponible. Estos se dividen a su vez en las tareas pertinentes con diferentes tipos de dependencias y fechas de inicio pero caracterizándose por terminar todas las tareas de un mismo bloque objetivo en la misma fecha. A más de estas tareas se han incluido las diferentes fechas de entrega lo que permite hacer una división del tiempo mucho más acertada. Esta planificación es incluida como un anexo bajo el nombre de: "Anexo 1: Planing".

1.4 Metodología

En una primera fase de este trabajo, debido a la falta de conocimiento sobre el tema, se ha realizado un proceso de documentación y el estudio del funcionamiento de Bitcoin mediante la realización de análisis de experimentos tales como el sniffing de una transacción o los pasos necesarios para la participación en el minado de bloques. Se ha buscado obtener el conocimiento previo necesario para la realización de las siguientes fases así como documentarlo y adjuntarlo con tal de proveer en el mismo trabajo, al lector, el mismo conocimiento necesario.

En una segunda fase se ha procedido a la realización del diseño de la aplicación de detección con lo conocimientos obtenidos anteriormente. Al estudio y comprensión del par de herramientas importadas incluidas en la propia herramienta que tienen por objetivo facilitar ciertas partes del desarrollo. A su implementación con el objetivo de la obtención de un producto capaz de cumplir con el objetivo principal del trabajo. Y posteriormente se ha estudiado mediante la utilización en repetidas ocasiones en escenarios distintos y evaluación crítica de tanto los resultados obtenidos como del funcionamiento de los procesos que se ejecutan.

Se ha finalizado con una autoevaluación del trabajo realizado con tal de determinar la aportación que ha supuesto realizar este trabajo para uno mismo. También, a partir de la evaluación de los resultados y la experiencia obtenida, unos nuevos caminos que deberían seguirse con tal de enmendar y evitar fallos.

1.5 Estructura del Documento

El resto del documento está estructurado de manera como se describe a continuación:

- El capítulo 2 contiene información que proporciona al lector conocimiento sobre Bitcoin. Se ha tratado de incluir la información imprescindible que el lector necesitará con tal de comprender las bases de los siguientes apartados. Contiene desde una introducción hasta explicaciones detalladas del comportamiento de los diferentes elementos presentes en Bitcoin.

- El capítulo 3 corresponde a la realización del objetivo principal, desarrollar una herramienta para la detección de los mineros. En el se concretara en mayor medida que se espera de esta herramienta, su desarrollo así como una evaluación de la herramienta.
- El capítulo 4 contiene los posibles caminos que se deberían seguir para la realización de los objetivos secundarios. No incluye una implementación y un análisis, restando en poco menos a una explicación del diseño, debido a que la herramienta desarrollada no ha proporcionado los resultados con la suficiente exactitud para ser implementados.
- Por último, en el capítulo 5 se presentan las conclusiones a las que se ha llegado en terminar el trabajo en relación a la realización del mismo así como del resultado general del trabajo. En este apartado también se incluyen sugerencias sobre mejoras posibles o futuras líneas de expansión.

CAPÍTULO 2

Bitcoin

2.1 Introducción

Cuando se habla de Bitcoin normalmente se piensa directamente en una abstracción de una moneda física de una divisa, como por ejemplo una moneda de 1 euro o un billete de 1 dolar, a una divisa digital. Si bien Bitcoin cumpliría esta definición lo cierto es que es bastante más y no unicamente por los diversos nuevos usos que se le puede dar a diferencia de los de las divisas tradicionales. Más que una simple moneda Bitcoin es un sistema con una multitud de elementos que lo forman, desde protocolos y tecnologías hasta usuarios. En este apartado se intentara introducir Bitcoin como conjunto.

Bitcoin apareció por primera vez en 2008 mediante la aparición del articulo "Bitcoin: A Peer-to-Peer Electronic Cash System" escrito bajo el seudónimo de Satoshi Nakamoto [13]. En este articulo se define la idea de un sistema de moneda electrónica descentralizada. La moneda electrónica ya había sido pensada anteriormente a la aparición del articulo pero hasta ese momento la norma era seguir diseños centralizados en que estructuras de poder controlasen su funcionamiento. Estos diseños centralizados eran débiles a ataques a los centros de control, evitándose con una estructura descentralizada.

La "robustez" que proporciona una estructura descentralizada, junto con otras medidas como la criptografía, no es la única característica del sistema. Las más apreciadas por sus usuarios, junto con la anterior, es el "anonimato" de las transacciones y la imposibilidad de controlarlas o bloquearlas. Mientras que en un banco te tienes que identificar para abrir una cuenta y es en última instancia el mismo banco el que decide llevarla a cabo o no, en Bitcoin no existe una asociación directa de persona a dinero o transacciones entre entidades. El dinero esta asociado en transacciones pasadas a una dirección donde la forma en que demuestras que posees el dinero es mediante una clave privada del sistema de clave publica, como el pin de una tarjeta de crédito,

que te permite generar el mensaje de transacción valida. La forma de realizar la transacción, lo que el banco realiza, es enviarles este mensaje y que terminen aceptándola como realizada. Con esta única necesidad de conocer una contraseña para "poseer" el dinero y hacer que los demás usuarios acepten un mensaje podemos realizar envíos de dinero a cualquiera, en cualquier parte del mundo, casi de inmediato, sin coste apreciable y sin necesidad de saber quien esta enviando el dinero o quien lo recibe.

Actualmente Bitcoin, como moneda, tiene un valor, en comparación al resto de divisas, bastante fluctuante. Las divisas normales están asociadas a países y con su poder económico respecto del resto le dan estabilidad. Bitcoin no tiene ningún país asociado sino un conjunto de usuarios que lo usan repartidos en muchos países. La confianza de estos usuarios puede variar mucho más drásticamente mediante la difusión de noticias y similares que la depositada en la economia de un país. Un ejemplo de este efecto es como de septiembre a diciembre de 2013 su valor paso de alrededor de 100 dolares a superar los 1000 dolares debido a la especulación y posteriormente, en apenas 4 meses, bajo de los 500 dolares debido en gran parte a la prohibición en China y el cierre de Mt. Gox [6, 1, 14].

La principal forma de cambiar los bitcoins a la divisa del propio país es mediante el uso de casas de cambio como BTC-e, las cuales retienen una comisión y como se ha visto pueden afectar en gran medida al valor de este. Esto lo convierte en una moneda de muy difícil uso cotidiano para la compra/venta de productos aunque cuenta con una comunidad de usuarios que ofrece productos a cambio de bitcoins, todos ellos principalmente a través de internet [22]. Si bien esos puntos de venta pueden no ser apenas usados mas que por la mera curiosidad, innegablemente, Bitcoin se ha ido extendiendo en el mercado suponiendo los ejemplos de más peso la adopción de Microsoft en la plataforma de pago de sus consolas o la aparición de cajeros con los que poder cambiar Bitcoins por dinero en la calle [10, 9]. Esto acerca cada vez más a una utilización real de Bitcoin como moneda de uso y no como valor especulativo, lo que hace interesante conocer su uso y funcionamiento.

2.2 Utilización y funcionamiento

La mejor forma de explicar el uso y funcionamiento de Bitcoin para un usuario que desee realizar compras utilizando esta divisa es mediante un ejemplo del mismo. En este apartado se detallará en que consisten los pasos para realizar pagos de bitcoins entre usuarios (que a partir de este momento llamaremos transacciones).

El primer requisito que un usuario ha de cumplir es disponer de un wallet (cartera) que abstraído al mundo físico podría considerarse un banco que crea uno mismo, a diferencia de inicialmente podría crearse que equivaldría a una cuenta o cartilla. El wallet esta integrado a un cliente bitcoin que es un software que permite realizar ciertas acciones. Las acciones mínimas que ha de incluir un cliente para realizar la función de wallet son: la creación de direcciones bitcoin, almacenar las direcciones generadas con sus claves (pueden codificarse), crear transacciones validas y enviarlas a la red. A parte de estas mínimas funciones se puede incluir otras que, aunque no son necesarias para realizar transacciones, son necesarias dentro del sistema Bitcoin. En base a las funciones y características del cliente puede considerar que hay 3 tipos:

1. Cliente Completo: Se caracteriza por guardar el registro de todas las transacciones realizadas hasta el momento (blockchain), el wallet y realizar las transacciones [11].

2. Cliente ligero: Solo guarda el wallet y delega la comprobación de que las transacciones y su realización a un tercero [8].

3. Cliente web: Todo es almacenado por un tercero y ofrecido como un servicio necesario para cualquier proceso [5].

Independientemente de la elección es necesario disponer de bitcoins. La forma mas común de obtener bitcoin es mediante casas de cambio o a través de usuarios directos. Pero para poder recibir estos bitcoins primero el wallet debe generar una dirección que deberá usarse como referencia para la recepción.

Una dirección es una cadena de caracteres (números y letras) que suele empezar por “1” o “3” (para la red en producción mientras que para la red de pruebas suele comenzar con “n” o “m”) y con una longitud de entre 29 y 33 dígitos, o como versión alternativa esta cadena también es codificada en QR. Un ejemplo de estas dos versiones de una misma dirección son:

- Dirección Bitcoin: 1HNspymX4sZxuTKTKRqYmdL8d6jmNDRLoU

Figura 2.1: Dirección Bitcoin QR

Estas direcciones, la secuencia de caracteres, no son aleatorios sino que son el producto de un proceso complejo que usa criptografía de clave publica, pero a diferencia de la tradicional, la usada en Bitcoin es criptografía de curva elíptica [12, 25]. El proceso de creación de una dirección se compone de los siguientes pasos [21, 15]:

- Obtener una clave privada ECDSA. Ej: ”18E14A7B6A307F426A94F8114701E7C8E774E7 F9A47E2C2035DB29A206321725”.

- Obtener la clave publica de 65 bytes a partir de la clave privada anterior. Ej: ”045086 3AD64A87AE8A2FE83C1AF1A8403CB53F53E486D8511DAD8A04887E5B23522CD470 243453A299FA9E77237716103ABC11A1DF38855ED6F2EE187E9C582BA6”.

- Realizar un hash SHA-256 sobre la clave publica. Ej: ”600FFE422B4E00731A59557A5C CA46CC183944191006324A447BDB2D98D4B408”.

- Realizar un hash RIPEMD-160 sobre el SHA-256. Ej: "010966776006953D5567439E5E39F86A0D273BEE".
- Añadir un byte al resultado del hash RIPEMD-160. El valor de este byte depende de para la red que este destinada, para la red de producción es el valor 0x00 mientras que para la red de pruevas es 0x6f. Ej: "00010966776006953D5567439E5E39F86A0D273BEE".
- Realizar un hash SHA-256 sobre la secuencia en el paso anterior. La secuencia con el byte de la red y el hash RIPEMD-160. Ej: "445C7A8007A93D8733188288BB320A8FE2DEBD2AE1B47F0F50BC10BAE845C094".
- Realizar otro hash SHA-256 sobre la secuencia en el paso anterior. El SHA-256. Ej: "D61967F63C7DD183914A4AE452C9F6AD5D462CE3D277798075B107615C1A8A30".
- Escojer los primeros 4 bytes de la secuencia anterior como el checksum de la dirección. El: "D61967F6".
- Unir el primer hash RIPEMD-160 mas el checksum del paso anterior para generar la dirección. Ej: "00010966776006953D5567439E5E39F86A0D273BEED61967F6".
- Convertir la cadena de bytes a una cadena base58. Ej: "16UwLL9Risc3QfPqBUvKofHmBQ7wMtjvM".

Una vez se dispone de una dirección Bitcoin es posible recibir transacciones en que nuestra dirección sea la destinataria o una de las destinatarias de cierta cantidad de BTC en una transacción. Una transacción esta constituida en por las entradas y las salidas de BTC. Concretamente el formato de una transacción es:

Una vez se dispone de una dirección Bitcoin es posible recibir transacciones en que nuestra dirección sea la destinataria o una de las destinatarias de cierta cantidad de BTC en una transacción. Realizar una transacción varia un poco de un cliente a otro pero en el cliente Bitcoin Core se dispone de una funcionalidad que busca automáticamente transacciones pasadas utilizables en las entradas de tal modo que únicamente debas incluir las salidas (dirección, cantidad y un comentario). El formato de una transacción es [23]:

- Número de versión: Son 4 bytes, que actualmente el valor marcado es 1, para especificar la versión del formato de la transacción.
- Número de entradas: Son de 1 a 9 bytes que especifican el número de entradas que tendrá esta transacción. El formato de este campo es "Variable length integer" que consiste en especificar el numero requiriendo la menor longitud de datos. Esto lo realiza cambiando la estructura según el valor numérico: uint8 (1 byte de dato) , 0xfd(byte de control)+uint16(2 bytes de dato), 0xfe(byte de control)+uint32(4 bytes de dato), 0xff(byte de control)+uint64(8 bytes de dato).
- Lista de entradas: El numero de bytes dedicado a este campo es variable dependiendo de la cantidad. Estas entradas a la vez están estructuradas:
  - Hash de la transacción origen: Son 32 bytes que contienen el hash SHA-256 de la transacción de la que provienen los BTC de entrada.

  - Índice de la salida origen: Son 4 bytes que nos indican el índice de la salida que se usa como esta entrada.
  - Longitud del Script: Son de 1 a 9 bytes con el formato “Variable length integer”.
  - Script: Son un numero variable de bytes que contienen la prueba de poder utilizar estos bytes. Número de secuencia: Son 4 bytes con valor irrelevante en la mayoría de los casos.

- Número de salidas: Son de 1 a 9 bytes que especifican el número de salidas que tendrá esta transacción con el mismo formato que el número de salidas.
- Lista de salidas: El numero de bytes dedicado a este campo es variable dependiendo de la cantidad. Estas entradas a la vez están estructuradas:
  - Valor: Son 8 bytes que indican la cantidad de Satoshis (0.00000001 BTC) a transferir.
  - Longitud del Script: Son de 1 a 9 bytes con el formato “Variable length integer”.
  - Script: Son un numero variable de bytes que especifican la dirección Bitcoin a la que son transferidos y como pueden ser gastados.
- Tiempo de bloqueo: Son 4 bytes que establecen un tiempo a partir del cual la transacción podrá ser usada. Esto puede estar especificado como una cantidad de bloques que la han confirmado o una fecha a partir de la cual sera válida.

Transaction

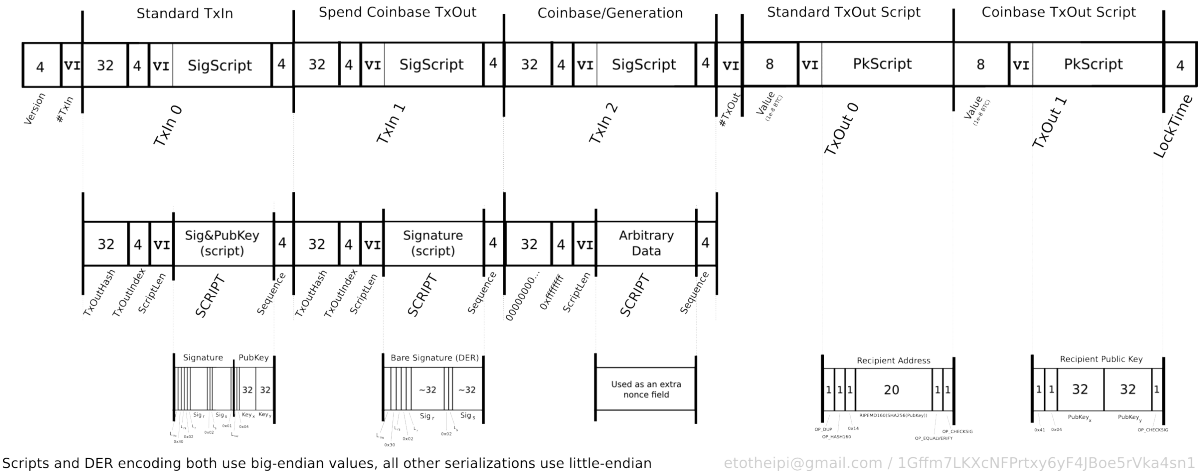

Figura 2.2: Formato transacción [23]

Toda esta información que contiene la transacción sera agrupada en un mensaje y transmitida por la red Bitcoin a los usuarios. Los usuarios receptores pueden comprobar las transacciones Bitcoin con salidas referenciando a sus direcciones en entradas y salidas y saber su saldo disponible. Para que la transacción sea válida esta ha de tener fondos en las entradas suficientes para la cantidades de la salidas más una pequeña diferencia en que esta cantidad es conocida como comisión de transacción. Esta comisión es una pequeña ganancia que obtienen los mineros al crear bloques en que esta transacción este incluida.

2.3 La red Bitcoin

La Red Bitcoin corre sobre internet y pose una arquitectura P2P, en concreto una topología de malla completamente descentralizada. Esta característica de completa descentralización es lo que le confiere su gran resistencia a ataques aunque los nodos de manera individual puedan no serlo. Al igual que los nodos de la red P2P están conectados, los clientes Bitcoin establecen conexiones entre ellos. A diferencia de la norma, estos nodos pueden tener funcionalidades muy diferentes que aun sin estar definido, unos nodos son más importantes que otros para garantizar el buen funcionamiento de la red. Las funcionalidades que un cliente puede tener son [3]:

- Wallet: El cliente puede operar transacciones. Concretamente se limita a la generación de direcciones y pagos. Con esta única funcionalidad el cliente no puede comprobar que los pagos se realicen, hacerlos públicos o saber su estado real.
- Enrutamiento: Esta esta es la funcionalidad imprescindible para cualquier cliente. Con ella se participa en la red estableciendo conexiones entre nodos y poder publicar transacciones.
- Blockchain: Esta funcionalidad esta únicamente para los clientes completos y consiste en disponer en memoria del registro completo de la blockchain. Esto permite al cliente por si mismo conocer su saldo y comprobar que cualquier pago que pueda recibir sea válido.
- Minero: Estos son los nodos que generan los bloques, mediante la computación intensiva. Pueden requerir para su funcionamiento tanto la blockchain como el enrutamiento en caso que sea un minero autónomo o en caso de pertenecer a una piscina de mineros le bastará con usar la función de enrutamiento de la piscina.

Como se ha visto la función de enrutado es obligatoria para participar en la red. Para establecer las conexiones se sigue el protocolo P2P bitcoin aunque no es el único utilizado. Recordemos que un minero puede actuar individualmente o pertenecer a una piscina. En este ultimo caso las conexiones no las realizará con los nodos de la red Bitcoin sino con la piscina. El protocolo que se acostumbra a usar en las piscinas es Stratum. También existe el caso de los wallet. Es posible disponer de únicamente un cliente con un wallet pero este no es capaz de verificar las transacciones sin la blockchain. También si no dispone del enrutado con el protocolo P2P bitcoin no podrá hacer públicas transacciones realizadas. Aun así los clientes que únicamente disponen de un wallet implementan sistemas de enrutado, con protocolos propios, que les comunica directamente con un tercero capar de realizar todas estas funcionalidades. Esta comunicación es exclusiva entre el cliente wallet y el servidor que provee el servicio y es un nodo con las funcionalidades de la blockchain y el enrutamiento con P2P bitcoin.

Como se ha visto se forma un gran número de pequeñas redes aisladas de la red Bitcoin con clientes de ella pero con estos clientes aislados del resto. Los exploradores de nodos de la red Bitcoin actualmente suelen tener identificados a unos 6.000 nodos. Pero estos nodos son únicamente los que disponen de la funcionalidad de enrutado con el protocolo P2P de Bitcoin. Los demás clientes dentro de estas subredes resultan inaccesibles pero claramente se puede determinar que el número de nodos es muy inferior al de clientes reales utilizados

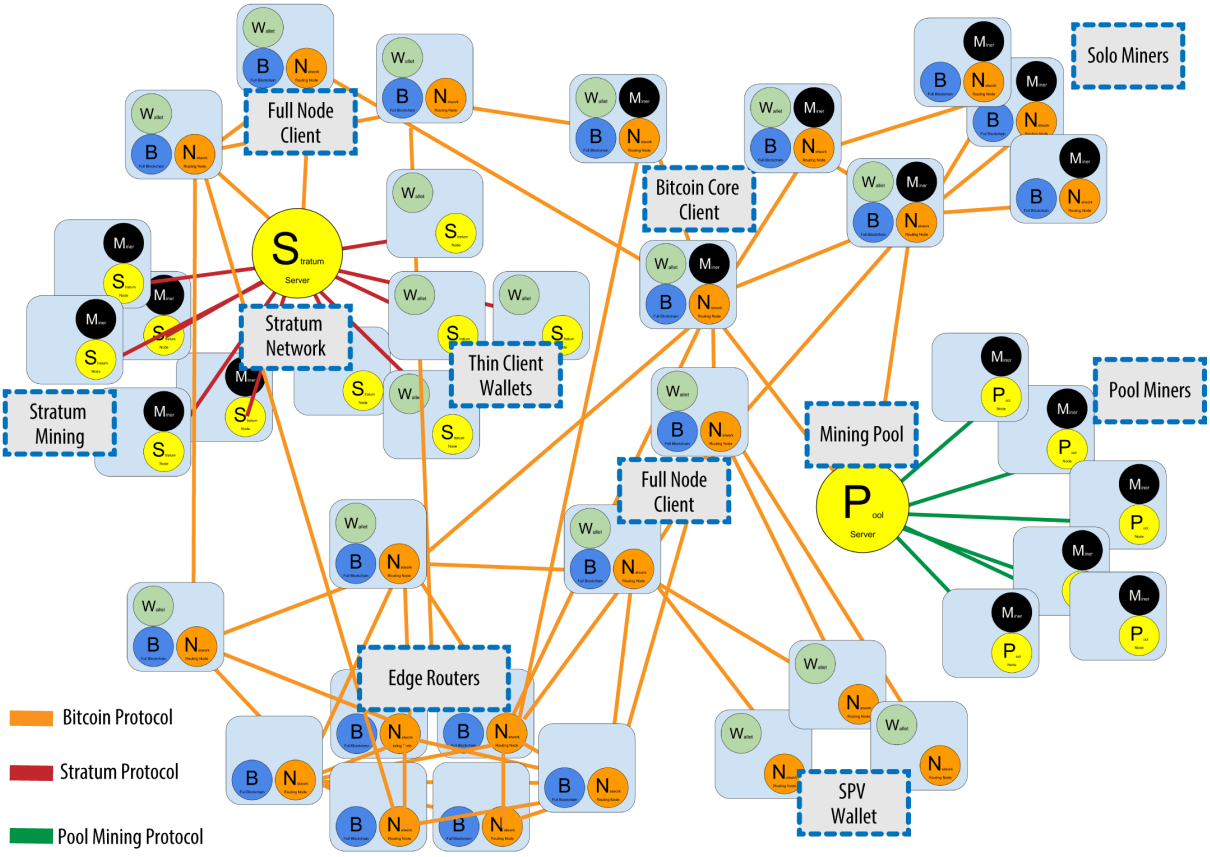

Figura 2.3: Red Bitcoin [3]

2.4 Los Bloques y la Cadena de Bloques

En los apartados anteriores se han explicado las transacciones, su estructura y utilidad, y la red Bitcoin, con su estructura y componentes. Se ha comentado como las transacciones son mensajes estructurados que son distribuidos por los clientes Bitcoin conectados a la red entre estos mismos clientes. Llegados a este punto el sistema Bitcoin aún contiene elementos que necesitan ser explicados para poder comprender su funcionamiento como sistema monetario. Estos elementos son los bloques y la Cadena de Bloques. Si bien es posible hacer la analogía de que los clientes dan acceso al sistema, la red da el escenario común sobre el que regirse y las transacciones son la utilización, los bloques y la cadena de bloques dan la fiabilidad y/o inmutabilidad a estas últimas.

Debido a la dificultad de explicar la Cadena de Bloques sin comprender al Bloque como elemento individual se procederá con él primeramente. Al igual que una transacción un Bloque es un conjunto de datos estructurados en base a un patrón. Estos datos son agrupados en el proceso de minado y una vez realizado son publicados en la red como se realiza con las transacciones. El objetivo individual, separado de la Cadena de Bloques, se puede considerar que és incluir,dentro del mismo, cierto número de transacciones entre otros datos y "sellarlos", esto se realiza mediante el proceso de minado que se explicará más adelante, con tal de poder asegurar que se han realizado esas transacciones en un cierto momento y no se han modificado. Con tal de cumplir

este objetivo la estructura del bloque es la siguiente [18]:

- Número: Son 4 bytes que por defecto tienen el valor 0xD9B4BEF9.
- Tamaño: Son 4 bytes que nos indican el número de bytes que determina el tamaño conjunto de los siguientes campos el bloque.
- Cabecera: Son 80 bytes que no proporcionan información sobre el bloque. Este campo esta a su vez compuesto por 6 subcampos que también siguen una estructura:
  - Versión: Son 4 bytes que contienen la información de la versión del software que estemos usando.
  - Bloque anterior: Son 32 bytes que identifican al bloque anterior de manera única mediante un hash. Este hash es construido de la siguiente manera: se concatenan los 6 campos de la cabecera de un bloque y a continuación se realiza un hash SHA256 sobre esta cadena y otro sobre el resultante del SHA256 (doble SHA256).
  - Raíz Merkle: Son 32 bytes tambien que se utilizan para determinar de forma rápida si una transacción está incluida o no en la lista de transacciones de este bloque una vez está "cerrado" . Estos bytes son determinados mediante la construcción de un árbol Merkle [26]. Este proceso consiste en aplicar un doble SHA256 a las transacciones, estos hashes son agrupados por pares (en caso de quedar el último suelto al ser un número impar es duplicado) y se aplica a estos pares el doble SHA256. Con esto el número de hash se ha visto reducido a la mitad. Este proceso se continúa hasta que únicamente reste un único hash doble SHA256 que es incluido en el campo. Posteriormente se puede usar un filtro de floración sobre la raíz y determinar recursivamente si una transacción puede o no estar incluida en un bloque [2]. Esto permite a ciertos clientes consultar si las transacciones se han realizado sin la necesidad de recibir el bloque con todas sus transacciones como prueba.

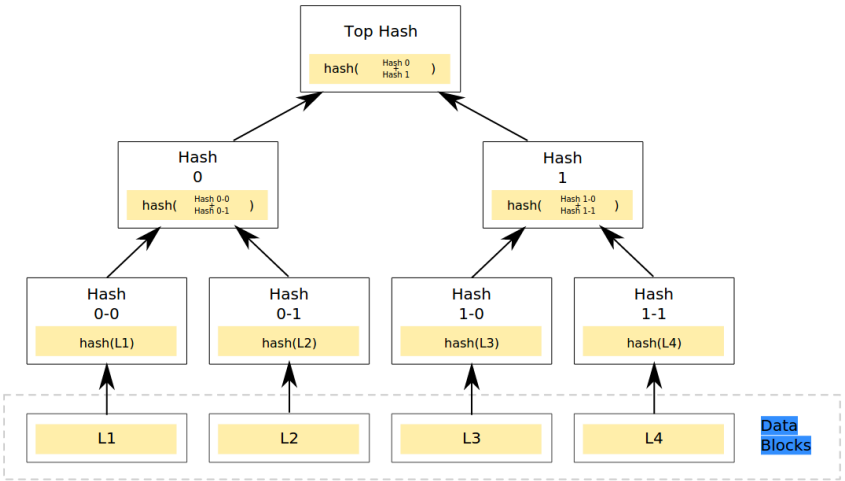

Figura 2.4: Arbol de Merkle [26]

  - Timestamp: Son 4 bytes que contienen un tiempo aproximado de cuando este bloque ha sido minado. El formato es el Unix Epoch, segundos desde el 1 de Enero de 1970.

  - Dificultad: Son 4 bits que determinan la dificultad con la que el bloque ha sido minado. Esta dificultad es determinada por el conjunto de la red cada 2016 bloques en función de potencia de cómputo actual y el objetivo de minar un bloque cada 10 min [19].
  - Número: Son 4 bytes que son aprovechados en el proceso de minado para "minar" los bloques.

- Contador: Son de 1 a 9 bytes, estructura VI, que nos indican la cantidad de transacciones que este bloque incluye.

- Lista de Transacciones: Con tamaño variable este campo incluye la lista de transacciones, con el formato descrito en el apartado 2.2, que son validadas al minar el bloque.

Realizada la explicación de la estructura del bloque se puede identificar el elemento clave que explica que es la Cadena de Bloques. Este elemento claves es el hash del bloque anterior en la cabecera lo que permite enlazar un bloque con uno anterior. Esta consecución de bloques enlazados es lo que forma la Cadena de Bloques. Los enlaces de bloques se realiza hacia atrás, el más joven contiene la referencia del más antiguo, lo que puede verse que esto permite que diversos bloques jóvenes apunten a un mismo bloque antiguo. Esto supondría un problema para el objetivo de la Cadena de bloques si no se hiciese nada al respecto.

Como se ha dicho anteriormente el objetivo del bloque es sellar un número de transacciones verificando que han sido realizadas para un tiempo concreto. Pero dado que no hay una autoridad que mine estos bloques, sino cualquier usuario los puede crear y enviar a la red Bitcoin, con tal de establecer una forma de validar a estos bloques, fuera de que estén correctamente formados o no, existe la Cadena de Bloques. Su objetivo primordial es establecer entre todos los usuarios de la red Bitcoin una secuencia de Bloques única que serán los que se consideran que contienen la lista de transacciones a tener en cuenta. Recordemos que anteriormente se comentó que para realizar una transacción es necesario referenciar a otra/as como entrada, estas transacciones deberán ser comprobadas dentro de la Cadena de Bloques antes de aceptar la transacción como válida y que posteriormente la misma sea incluida en la misma Cadena de Bloques. En definitiva la Cadena de Bloques es el consenso de los usuarios de la red Bitcoin sobre las transacciones realizadas realmente en el sistema.

La secuencia de bloques se puede dividir si aparece en la red dos bloques correctamente construidos que tengan en el campo de Bloque anterior el mismo. Esto podría secuencias de bloques diferentes las cuales tengan transacciones que han de ser aceptadas muy diferentes. Este problema se soluciona con el criterio que determina la Cadena de bloques a elegir. Este consiste en a partir del Bloque Génesis, que es el primer bloque que se minó de forma algo "artificial", que los clientes incluyen en el código del software que lo forman se determinara que la secuencia de Bloques más larga es la cadena de Bloques que el sistema considera aceptada y por transferencia las transacciones que contiene. Esto junto con el esfuerzo necesario con tal de minar un bloque con una estructura válida proporcionan la seguridad al sistema.

2.5 Minería

En los apartados anteriores ya se ha mencionado el proceso de minado de los bloques como un método para lograr algún fin pero en este apartado se explicará qué es la minería. La minería es un proceso esencial para el funcionamiento del sistema Bitcoin y a riesgo de resultar repetitivo es necesario remarcar los dos objetivos que tiene principalmente el proceso.

El primero es la producción de bitcoins de nueva circulación, expedir nuevos en el sistema, con tal de aumentar la cantidad disponible en el sistema. Esta cantidad tiene un máximo cerca de los 21 millones de bitcoins. Este hecho, que el número de bitcoins en el sistema sea finito ha ocasionado críticas debido al peligro de una moneda deflacionaria, el cual es esencialmente que la moneda no sea utilizada debido al hecho de los usuarios saben que su valor será cada vez mayor. Si bien se ha visto indicios de este posible problema en sus fluctuaciones de valor (caída del precio de mercado en Diciembre de 2013) no se discutirá su posibilidad. Este objetivo de expedir nuevos bitcoins al sistema con cada nuevo bloque que se añada a la Cadena de Bloques se materializa al permitir un tipo especial de transacción, única en cada bloque, que transfiere un número determinado de bitcoins sin origen a una dirección de destino. Esta dirección suele pertenecer al minero factor que es su motivación para invertir recursos en la minería, un proceso que como se verá es costoso, y así obtener algún beneficio. Aún así esta cantidad fija no es la única fuente de ingresos del bitcoin, ya que la comisión de transacción de todas las transacciones incluidas en el bloque también son desviadas a la dirección del minero.

El segundo objetivo de la minería es asegurar que las transacciones realizadas son válidas y no podrán ser negadas por el dueño de la dirección. Recordemos que las transacciones las expedían los mismos usuarios siendo un mensaje con un con unas entradas de bitcoins los cuales se demostraba ser el propietario y unas salidas hacia unos destinatarios. El mismo usuario podría usar las mismas entradas en otras transacciones sin un control de las emitidas a la red. Este control se realiza al comprobar que no hubiesen sido usadas anteriormente en alguna transacción incluida en algún bloque de la Cadena de Bloques. La minería es incluir bloques en la cadena, lo cual no es simple y requiere de un gran cantidad de cálculo y cuyo proceso veremos seguidamente. Este elevado coste de minar un bloque (calcular los parámetros para que sea aceptado según sus características) es el mecanismo de seguridad que cumple este segundo objetivo aunque no individualmente. Se logra también junto al hecho de incluir una referencia a un bloque predecesor y que la Cadena de Bloques con la sucesión de bloques valdos más larga es la aceptada, siendo las transacciones de sus bloques las únicas aceptadas. Si bien es cierto que es posible producir bloques de manera propia con el objetivo de ignorar cierta transacción resulta inviable ser capaz de producir el número suficiente de bloques para negar una transacción cuando esta hace cierto número de bloques que fue incluida en la Cadena de Bloques.

Una vez comentados los objetivos de la minería ya es posible empezar la explicación del proceso. Para ello debemos partir del instante en que una transacción creada por un cliente es enviada a la red. Los clientes recibirán esta transacción y si es válida la transmitirán a otros a la vez que la guardan en su memoria. Este acto de guardar una transacción válida en memoria se puede considerar el primer paso en el proceso de minado. Estas transacciones son guardadas en lo que se llama piscina de transacciones, un espacio de memoria volátil que contiene todas las transacciones que ha recibido este cliente desde su inicio y que aún no han sido incluidas en

un bloque las cuales desaparecen al cerrar el cliente. Cuando el mismo cliente reciba un bloque deberá de tratar de eliminar las transacciones, de la piscina de transacciones, que aparezcan en ese bloque. Esto es necesario porque el segundo paso en el proceso de minería es que el software minero, que es el encargado de calcular el bloque, solicite a la piscina transacciones para construir el bloque (la lista de bloques en la estructura de un Bloque). Estas transacciones, por defecto, serán proporcionadas primero aquellas consideradas de alta prioridad. En el cliente bitcoin core estas transacciones son las que obtienen un valor de prioridad superior a 57.6 millones en la siguiente fórmula [24]:
priority = sum(input value in base units * input age) / size in bytes .
El resto de transacciones serán escogidas en función de tener una comisión mayor.

Una vez tenemos la lista de transacciones del bloque a construir repleta procedemos a calcular el total de las comisiones de todas las transacciones. Este valor junto a la recompensa obtenida por el bloque serán incluidos en la primera transacción del bloque, la transacción generadora. La transacción generadora es la que añade nuevos bitcoins al sistema. Como cualquier transacción tiene una salida, que en este caso es la dirección del minero y como cantidad la calculada de la suma del total de comisiones y el valor de recompensa por bloque. La diferencia de esta transacción reside en la lista de entradas, donde los elementos de cada entrada son utilizados de forma diferente. Teniendo la estructura la utilización que se le hace es:

- Hash de la transacción origen: Son 32 bytes en que todos son 0.
- Índice de la salida origen: Son 4 bytes con el valor 0xFFFFFFFF.
- Longitud del Coinbase: Son de 1 a 9 bytes con el formato “Variable length integer” que indica la longitud del siguiente campo.
- Coinbase: Son un número variable de bytes con valor aleatorio que pueden ser usados por el minero.
- Número de secuencia: Son 4 bytes con el valor 0xFFFFFFFF.

Habiendo terminado de forma definitiva la lista de transacciones se procede a especificar los campos de la cabecera (Versión, bloque anterior, raíz Merkle, timestamp, y la dificultad). Restando el campo número, de 4 bytes, el cual es variado hasta que se obtiene un hash de la cabecera inferior a la dificultad.

Únicamente queda el proceso de minar el bloque que consiste en ir realizando hash SHA-256 de la cabecera. Este SHA-256, para considerar que el bloque ha sido minado, debe tener un valor inferior al especificado por el campo de dificultad. Los valores de los hash no pueden ser predichos y es por ello que no hay otra forma aparte de ir realizando hash continuamente para diferentes valores del campo número de la cabecera hasta encontrar un valor que permita obtener un hash con esta restricción. Momento en el cual el bloque puede ser emitido a la red y comprobado por todos los clientes si es valido o no simplemente calculando el hash con el valor número ya especificado en un valor que proporciona un hash que cumple el requisito de dificultad.

Como fin a este apartado resta explicar cómo es calculado el valor que el SHA-256 del bloque no debe superar. El campo de dificultad contiene con su primer byte el valor exponente y con

los tres siguientes el coeficiente, valores usados en la siguiente función para calcular la dificultad [4]:
$target = coeficiente * 2\hat{}(8 * (exponente - 3))$.

CAPÍTULO 3

Aplicación de detección

3.1 Alcance

La naturaleza de la arquitectura P2P de la red Bitcoin junto con los clientes y el tipo de redes bajo las que operan hace difícil su monitorización pero no imposible. El objetivo de esta segunda fase es desarrollar las herramientas necesarias para que un usuario pueda monitorizar la red Bitcoin y obtener una lista de nodos con una alta probabilidad de ser mineros. Se espera desarrollar un conjunto de herramientas de sencilla utilización mediante comandos de ejecución que permitan obtener directamente los resultados.

Se estima que una gran parte de los nodos de la red Bitcoin pueden encontrarse en muchas redes detrás de firewalls, NAT, etc y esto impida su visibilidad desde el exterior. Esta herramienta no pretenderá tener acceso a estas redes y poder identificar los nodos mineros internos sino únicamente identificar de entre todos aquellos nodos que puedan ser accesibles cuales de estos tienen más probabilidad de estar minando.

3.2 Diseño Conceptual

En base a los requisitos de la herramienta y el alcance del proyecto se ha realizado un diseño de dos herramientas así como la utilización de tres tipos de ficheros distintos que con su utilización buscan cumplir el objetivo principal del trabajo. Las herramientas han sido llamadas ObtenerNodos y MonitorizarNodos mientras que los tres ficheros son rangosout.txt, nodosfind.txt y nodoscandidatos.txt. A continuación se procederá a una explicación más detallada del objetivo, diseño y formato de cada uno de estos 5 elementos.

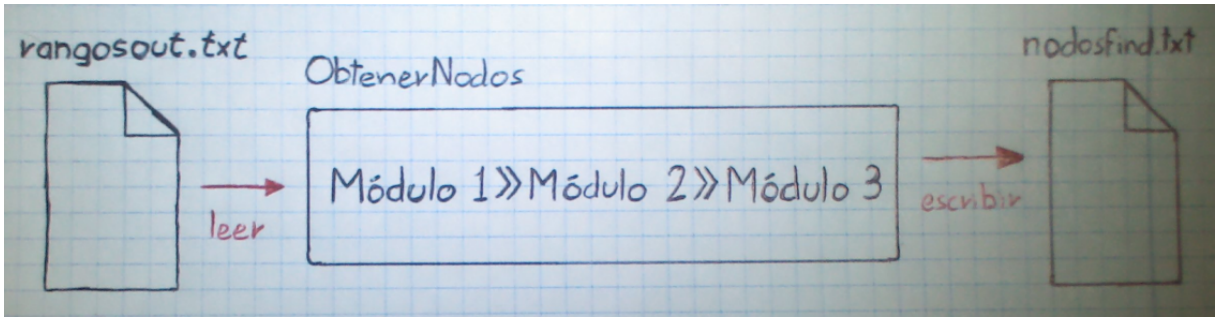

Figura 3.1: Aplicación: esquema A.

3.2.1 rangosout.txt

Este fichero ha de contener en cada línea una red (expresado con la IP y la máscara) en la cual no es necesario buscar nodos bitcoin a la hora del descubrimiento inicial. Esto es con el fin de reducir el inmenso número de Ip existentes. El archivo podrá ser modificado directamente por el usuario y sera leído por la herramienta ObtenerNodos. Un posible formato de una entrada de este archivo será por ejemplo: “192.168.0.0/24” (sin comillas).

3.2.2 ObtenerNodos

La primera herramienta necesaria para el cumplimiento del objetivo del trabajo. Esta herramienta tiene por objetivo proporcionar nodos que han de ser monitorizados. Con tal de cumplir este fin se ha hallado necesario realizar 3 acciones. Estas acciones, de cara a la implementación, se han separado en 3 módulos realizando cada uno una de las acciones. Estos son:

- Módulo 1: Realiza la lectura del archivo “rangosout.txt” y genera Ips de manera aleatoria, fuera de los rangos, y determina si tras esta se encuentra un cliente bitcoin.
- Módulo 2: El método anterior se presume extremadamente lento, por ello este módulo utiliza los nodos obtenidos en el anterior. Busca expandir de manera mucho mas rapida la lista de nodos conocidos. Para ello deberá preguntar al cliente de cada nodo encontrado los nodos que conoce. Tiene por objetivo obtener el mayor número posible de nodos, los cuales han sido comprobados por él al conectarse, preguntandoles que otros conocen. Estas consultas se repiten hasta haber comprobado todos los candidatos.
- Módulo 3: El anterior módulo, después de su finalización, ha de haber comprobado una inmensa cantidad de posibles nodos, y solo unos pocos de estos habían sido accesibles. Estos nodos accesibles serán guardados en el fichero ”nodosfind.txt”.

3.2.3 nodosfind.txt

Este fichero contendrá en cada línea un nodo (expresado con la IP y el puerto) de los accedidos en la ejecución de ObtenerNodos. Los nodos accesibles forman el conjunto en el que se ha de

buscar a los mineros por lo que la lista que los contiene debe estar disponible para la herramienta MonitorizarNodos. Para ello se ha elegido un archivo, que no deberá ser modificado directamente por el usuario a menos que tenga conocimiento fehaciente de nodos accesibles de la red Bitcoin.

Este archivo compone la base sobre la que la herramienta MonitorizarNodos comienza a trabajar, por lo que cuanto más recientes sean los resultados y mayor su número de entradas, mayor facilidad de inicio tendrá la herramienta. Un posible formato de una entrada de este archivo será por ejemplo: "199.8.89.23:8333".

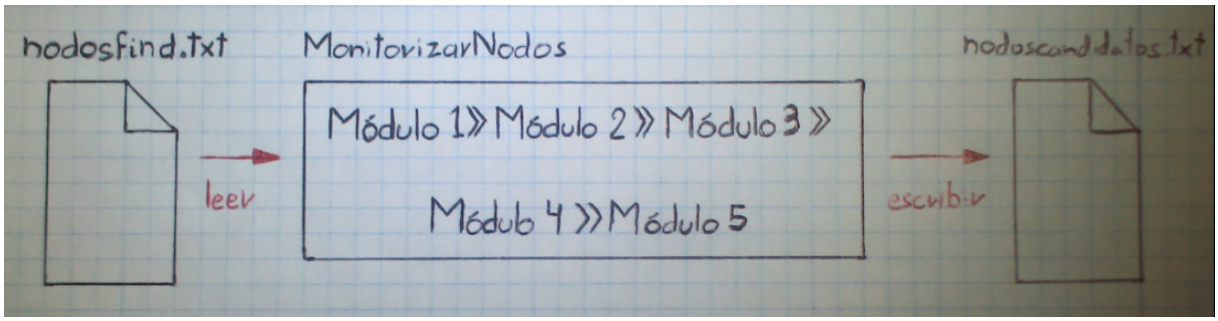

Figura 3.2: Aplicación: esquema B.

3.2.4 MonitorizarNodos

Esta segunda herramienta está diseñada con el fin de contribuir al cumplimiento del objetivo principal del trabajo. El objetivo de esta herramienta es realizar la monitorización de los nodos y mediante una heurística determinar de entre estos los mineros. Con tal de alcanzar esta meta se han identificado ciertas acciones que deberán realizarse, las cuales, como con la herramienta anterior, se han encapsulado en lo que he llamado un módulo. Un requisito y/o restricción muy importante que ha surgido al determinar su objetivo y las acciones que ha de realizar es que aparece la necesidad de realizar acciones en paralelo para cada nodo que se esté monitorizando. Continuando con el diseño, se han concebido los siguientes 5 módulos:

- Módulo 1: Su función es leer el fichero "nodosfind.txt" y crear los nodos que representan cada objetivo y que se utilizará como puente con el cliente del otro lado.
- Módulo 2: A partir de la lista de nodos creados anteriormente este módulo establece una conexión con cada cliente detrás de cada uno de los nodos. Estas conexiones deberán ser mantenidas en todo momento con tal de poder recibir, por parte de los diferentes clientes, mensajes del descubrimiento de un nuevo bloque.
- Módulo 3: Una vez la conexión con un nodo ha sido establecida es necesario monitorizar la actividad. La actividad consiste en el intercambio de mensajes, y de entre los posibles tipos de mensajes nos interesa monitorizar la anunciación de un nuevo bloque. Con este objetivo será necesario generar un timestamp de la recepción de cada bloque que recibe de cada nodo. Se requiere como mínimo asociar un identificador único por bloque independientemente del nodo que lo ha transmitido y del momento, y que este identificador esté asociado a su timestamp y a un nodo concreto.

- Módulo 4: Este módulo debe, cada cierto tiempo, realizar un control de los bloques recibidos por los nodos con el objetivo de determinar, mediante la heurística, la probabilidad de que los nodos sean mineros. Esto debe realizarse de la siguiente manera: de bloque nuevo descubierto determinar un número pequeño de los primeros nodos que lo transmitieron, aplicar una heurística a este grupo que proporciones pesos que proporcionen un valor numérico de las posibilidades de que este nodo sea minero, repetir este proceso para cada bloque que sea recibido, y intentar añadir nuevos nodos para monitorizar según los vecinos de los nodos que actualmente tengan más probabilidades de ser mineros.
- Módulo 5: Este módulo tiene como objetivo el detener la aplicación correctamente y proporcionar los resultados de la monitorización. Este módulo consta de dos fases. La primera fase es un hilo que está a la espera que el usuario mande la orden de detener la ejecución de la aplicación. La segunda fase consistirá guardar en el fichero "nodoscandidatos.txt" los resultados de la monitorización cada cierto lapso de tiempo con el fin de poder consultar los posibles candidatos a mineros en tiempo real.

3.2.5 nodoscandidatos.txt

Este fichero ha de contener una entrada por nodo (expresado datos de identificación como la IP, el puerto, el puntaje y el número de veces en haber sido el primero en publicar un bloque) que ha sido monitorizado y se encuentra siendo monitorizado. El archivo podrá ser visto directamente por el usuario y mediante el análisis de los valores del puntaje así como de la cantidad de número de veces en ser el primero en realizar un descubrimiento podrá extraer conclusiones acerca de con qué probabilidad este nodo es un minero y/o si está próximo a uno.

3.3 Implementación y Funcionamiento

En este apartado, debido a que el código de la herramienta puede llegar a ser difícil de abordar sin una introducción, se iniciará con una introducción de la tecnología y recursos usados para poner en contexto al lector sobre los requisitos que se puede necesitar para utilizarla. Seguidamente se realizará una explicación de los componentes que forman la aplicación, sin entrar en detalle en el código, únicamente mencionando los componentes más relevantes, acciones o interacción. Se terminará con una explicación de la puesta en marcha de la aplicación y las consideraciones que hay que tener con tal de obtener resultados ajustados.

3.3.1 Tecnología y recursos usados

Con tal de introducir al lector sobre el código que ver en las herramientas primeramente es necesario anunciar el lenguaje de programación que se ha empleado, en este caso ha sido python. La elección de este lenguaje ha sido debido a diversos factores. El primero es el conocimiento superior en este lenguaje sobre C++ que es el empleado en el cliente oficial "Bitcoin Core". El siguiente es debido a que aunque el cliente oficial está desarrollado en C++ y se puede pensar que se encontraría una mayor cantidad de recursos, python es un lenguaje que ha sido muy usado

en trabajos y estudios relacionados con bitcoin, por lo cual la cantidad de recursos disponibles es extensa. El último y decisivo, apoyado por el anterior, es que los dos recursos ya creados usados para realizar funciones concretas de la herramienta han sido implementados en este lenguaje. Por lo cual integrarlas en la herramienta empleando el mismo lenguaje de programación resulta trivial.

Estas herramientas han sido "Bitnodes" y "Bitcoin Sniffer" [27, 7]. Explicarlas detalladamente queda fuera de documento pero comentar por encima las características de su funcionamiento y su utilización dentro de la herramienta creada si que parece ser necesario:

- Bitnodes: Esta herramienta se caracteriza por implementar una clase que permite comunicarse con clientes bitcoin mediante la utilización de mensajes. Construye mensajes correctamente estructurados y los envía mediante sockets. Después del envío serpera la respuesta concreta a la petición que contuviese el mensaje. Aunque esta compuesto por diversos scripts unicamente se ha requerido la utilización de "protocol.py". Su funcionamiento es secuencial y las funcionalidades que incluye se reducen, útiles en nuestro caso, al establecimiento de conexión y consulta de nodos conocidos.
- Bitcoin Sniffer: Se caracteriza por consistir en una clase orientando sus funcionalidades a ser capaz de establecer una conexión con un cliente y poder recibir mensajes, especialmente los de propagación de transacciones y bloques. Esta clase funciona como un servicio que en arrancar se mantiene a la escucha y reacciona a las recepciones de los mensajes a diferencia de la anterior. Esta contenido en un unico script el cual se ha debido adaptar eliminando y añadiendo funcionalidades.

Antes de terminar este apartado queda enumerar aquellas librerías de python más importantes o que se considera que pueden no poseerse, pero que son necesarias en la aplicación. La primera librería a mencionar usada es "time" usada para determinar el momento en que un nodo de la red comunica un nuevo bloque entre otras funciones y para los logs de las herramientas [17]. La siguiente es "threading" empleando tanto semáforos para bloquear la manipularon de ciertas variables así como crear threads para cada conexión con otros clientes [16]. Las últimas librerías a mencionar son "socket", "gevent" y "asyncore" usadas para la comunicación con los clientes y poder recibir mensajes y generar eventos, estas mayormente usadas por las 2 herramientas anteriores.

3.3.2 Implementación

A continuación, en este apartado, se describen las características de la implementación de las aplicaciones. Aunque no se pretende entrar en detalle del código de los más de 20 archivos utilizados señalar las características de los puntos más relevantes, como están construidos y el porque de ser así, se considera necesario. Esta descripción se realizará individualmente para cada herramienta.

ObtenerNodos.py:

La implementación de esta herramienta apenas a variado del diseño en casi nada salvo algun extra con tal de mejorar la usabilidad. Aparte de las funcionalidades de los módulos, con tal de

poder controlar el estado de la ejecución, se ha implementado un "log" el cual guarda el estado en que se encuentra la ejecución de la herramienta y lo avanzada que va la búsqueda. Es necesario este "log" debido a que la ejecución, aun con el uso de threads, ha resultado extremadamente lenta pudiendo requerir ejecutarse durante días. Poder controlar si realmente la aplicación está funcionando correctamente resulta ser esencial.

Respecto a la implementación de los módulos, como anteriormente se ha dicho, se ha sido estricto y cada módulo realiza la acción requerida siendo representado por una función. El módulo 1 es realizado mediante la función "buscarClienteBitcoin()" que generará ip aleatorias y tratará de conectarse a ellas para determinar si hay un cliente accesible. Aunque se ha implementado usando threads evitando asi el tiempo hasta que la conexion alcanza el timeout el método es extremadamente lento y desproporcionado. Los clientes de Bitcoin utilizan un sistema de DNS para encontrar nodos al inicio de la conexión pero desgraciadamente no ha sido posible replicar este sistema con lo que con tal de evitar esta acción es aconsejable proporcionar la dirección de un nodo conocido y activo. El módulo 2 consiste en un bucle "while" que utiliza threads lo cuales solicitan los nodos conocidos a aquellos que se les ha podido contactar. Aquellos obtenidos serán comprobados y los consultados almacenados. Este proceso es el "corazón" de la herramienta y la que provee los resultados. Su ejecución puede durar días, hasta que todos los nodos de la red hayan sido comprobados, por ser posible requerir parar la ejecución sin perder los resultados se ha implementado otra "finDeProceso()", funcionalidad que permite mediante la escritura en un archivo detener el proceso de ejecución y guardar unos resultados que pueden ser igualmente utilizados por la herramienta MonitorizarNodos.py. El módulo 3 ha sido implementado únicamente con la función "ficheroGuardarNodos()" sin ninguna complicación o variación de lo esperado.

MonitorizarNodos.py:

La segunda herramienta y que supone el centro del trabajo ha sido la más difícil de desarrollar y que más ha diferido la implementación respecto al diseño. Igual que la anterior herramienta ha se han añadido las 2 funcionalidades extra implementadas, el log y el mecanismo para parar la ejecución. Los motivos son también la necesidad de controlar la ejecución de la aplicación y que debido al largo plazo de ejecución, en realidad nunca terminará su ejecución, se necesita un proceso para pararla y poder obtener los resultados finales.

Las acciones del módulo 1 y del 2 descritas en el diseño se han mantenido comprobando la accesibilidad del nodo. El módulo 1 está representado por la función "ficheroLeerNodos()" y el extra de comprobar que los nodos sigan siendo accesibles. El módulo 2 es representado por la función "HiloEscucharYPodarConexiones()" el cual es un thread que establecerá las conexiones a los diferentes nodos y las mantendrá abiertas como hilos en ejecución. Como funcionalidad extra se ha añadido el hecho de cada 6 min comprueba si los nodos siguen conectados. Si siguen conectados no se realiza ninguna acción pero en caso contrario ese nodo es descartado y ya no será monitorizado liberando así recursos. el tiempo de comprobar nodos cada 6 min ha sido inferido de la observación de las conexiones que el cliente Bitcoin core mantiene. Se ha comprobado que únicamente una o dos se suelen mantener durante más de 5 min, desapareciendo incluso antes de un minuto. Este tipo de nodos son muy abundantes en la red y suponen un gran consumo de recursos si no son eliminados una vez desaparecen. Por ello, con tal de eliminarlos se ha elegido un tiempo algo superior a su tiempo máximo de conexión con tal de eliminar las conexiones pero tratando de maximizar el tiempo de espera sin consumir recursos al realizar la poda.

En el momento en que el módulo 2 establece un conexión el módulo 3 de monitorización tambien se inicia de inmediato para cada una. El módulo 3 se ha materializado como la clase "ClaseGestorConexion" que termina utilizando la clase "NodeConn" con unas diferencias a su estado al obtenerse de "Bitcoin Sniffer". La clase "ClaseGestorConexion" dispone de las variables donde se almacenara información a emplear para determinar los nodos que son mineros. Estas variables son: listaBloques, puntosMineros y vecesMinero. Aprovechando el paso por referencia de python la primera de listaBloques es compartica con el sniffer. Este sniffer ha sido modificado de tal forma que cuando recibe un mensaje se genera un timestamp. Si el mensaje recibido es la propagación de un bloque, la información de este junto con el timestamp de la recepción, son almacenados en la variable. Esto se realiza simultanemanete para todos los nodos, al recorrelos periodicamente en el módulo 2, que usa el método "comenzarEscucha()" de "ClaseGestorConexion" que activa el sniffer como un loop asincrono (asincore.loop()) en caso de no estarlo ya.

El módulo 4 ha sido desarrollado con un bucle "while" el cual utiliza la anterior función de detener la ejecución explicada anteriormente como condición de parada. Este bucle ejecuta el resto de funcionalidades que han de implementarse cada 15 min. La razón de escoger este tiempo es debido al tiempo medio de minado. Recordemos que el tiempo medio/deseado al desarrollar bitcoin es que se minase un bloque cada 10 min con tal de que las transacciones pudiesen validarse en un tiempo razonable. Este tiempo de 10 min nunca es cumplido con exactitud sino que puede diferir en bastante medida, de pocos segundos lo mínimo a cerca de 1 hora máximo. Aun así los extremos no son lo más común y en todo caso la mayor cantidad de bloques son minados antes de 15 min (la dificultad asegura esto) con lo que el realizar una iteración cada 15min permite tener la mayoría de las veces alguna transacción recibida durante la espera. Como se ha dicho, este bloque contiene el resto de funcionalidades, y las que pertenecen al mismo módulo son: "vaciarBloquesConexiones()", "puntuarMineros()" y "sumarVecinosDelPrimero()". Mientras que la primera función se limita a extraer los bloques de las conexiones para poder tratarlos con independencia de la conexión y la tercera añade, de los nodos que han obtenido mejor puntuación como mineros en esta ronda, sus nodos conocidos al conjunto de nodos conocidos con el afán de, en caso de que estos no sean los minero y que el minero vuelva a publicar un bloque, estar conectado directamente a este para detectarlo. Las segunda función, "puntuarMineros()", es junto con el método "comenzarEscucha()" los que contienen la esencia de la herramienta. Esta función escoge por cada nodo las cinco conexiones que lo recibieron antes y los puntúa. La puntuación se realiza a más de anotar el número de bloques que ha sido el primero en publicar con una heurística que ha terminado siendo unos valores fijos según la posición en la que quedaron: 5000, 1500, 700, 300, 70.

Estos valores se han inferido de la estructura de la red, el comportamiento de los nodos y los objetivos deseados. Esta puntuación busca proporcionar un valor numérico de lo cerca que estan de un nodo minero y la potencia que tiene este nodo. Analicemos primero como esta puntuación nos puede proporcionar información de lo cerca que está de un nodo. En caso de no estar monitorizando un nodo minero que en un determinado momento publica un bloque este bloque nos llegará primero por aquellos nodos a los que los que inicialmente él se los envío en caso de estar conectado a ellos. En este caso alguno de sus vecinos será al que asignaremos como ganador y al que le preguntaremos por nuevos nodos que monitorizamos. Es probable que realizando esto obtengamos el nodo que lo extendió. En las sucesivas publicaciones de bloques

por este nodo el sera el ganador mientras que sus vecinos quedaran detrás múltiples veces por lo que los puntos se equilibran rápidamente siendo la del minero superior a los pocos envíos y contando con un indicador elevado de ser el primer nodo en transmitir un bloque. Por el contrario los vecinos, aunque tendran tambien puntuaciones altas ellos tendrán el indicador de bloques transmitidos primero muy bajo por lo que se podrá determinar que está conectado a un minero. La puntuación puede equivaler a la potencia del nodo comparada con el resto de nodos con un valor alto de bloques que publicaron primero. Y finalizando ya resaltar que el módulo 5 es englobado con la simple función "guardarPuntuacionMinero()" que nos guardara en un archivo indicando en su nombre las puntuaciones obtenidas.

3.3.3 Utilización

Como anteriormente se ha comentado la herramienta se encuentra dividida en dos: ObtenerNodos.py y MonitorizarNodos.py. Su utilización requiere de la ejecución de la primera, esperar el fichero que genera y utilizarlo por el segundo. A continuación se comenta cómo utilizar cada uno para tratar de agilizar los resultados.

ObtenerNodos.py puede ser ejecutado desde un terminal, si se dispone en el sistema de las librerías necesarias, con un simple comando. Esta herramienta tiene ciertos parámetros que son recomendables ajustarlos manualmente por el usuario con tal de acelerar la ejecución y obtener un resultado aunque este pierda precisión. Estos parámetros están incluidos en la inicialización de variables en la primera línea del código y son:

- maxThreads= : Cuando mayor sea el número de threads que se le permita mayormente se verá reducido el tiempo de ejecución debido a que los hilos son usados para realizar las consultas a los nodos, las cuales pueden llegar a tardar en establecerse, realizar la consulta y cerrar la conexión.
- modo= : Esta variable, que es un número entero, determina si la aplicación buscará todos los nodos conectados de todos los conocidos de todos los consultados o únicamente se ejecutará la búsqueda de nodos conectados durante un periodo corto (5-6 min aprox.). Ejecutar la aplicación para comprobar todos los nodos requiere darle el valor 0 mientras que cualquier otro, 1 por ejemplo, será en modo rápido.

Aparte de estas dos variables se incluye en el código un par de líneas para incluir un Nodo del cual se tiene constancia que esta conectado ya sea mediante el uso de terceras aplicaciones o servicios web. Esto permite saltar el proceso de encontrar una primera ip que aloje un cliente Bitcoin rastreando la red y probando combinaciones.

La finalización del comando "python ObtenerNodos.py" en el modo más se realiza en aprox 10min obteniéndose el fichero "nodosfind.txt" con las Ip comprobadas de los nodos conectados. El proceso de ejecución de la herramienta puede ser comprobado mediante el fichero de log que crea durante su ejecución y que tiene como nombre "logDel:[fecha de inicio del proceso].txt". Terminado hemos de mover el fichero al directorio donde se halle la herramienta "MonitorizarNodos.py". Otra forma es usar el fichero "Fin.txt" para parar la ejecución en cualquier momento

y seguir obteniendo los resultados, siendo esta última opción la más aconsejable a seguir para poder parar una vez obtenemos una cantidad suficiente de nodos con los que probar.

La herramienta que determina los mineros también es iniciada con un comando "python MonitorizarNodos.py" pero esta herramienta no finaliza nunca su ejecución a priori. Los resultado de estar monitorizando los nodos son copiados cada 15 minutos aproximadamente en el fichero "nodoscandidatos[date].txt". En este fichero se podrán ver los puntos y el número de veces en que cada nodo de los que está siendo monitorizado ha sido el primero en publicar un bloque. Por lo que a la calidad del resultado respecta esta mejora cuanto más tiempo el proceso haya estado en funcionamiento. Debido a esto es recomendable dejar la aplicación unas 24 horas funcionando antes de mirar los nodos de la red que determinamos como mineros. Como se ha dicho anteriormente la ejecución de la aplicación no tiene fin a no ser que se indique. El mecanismo para poner fin a la ejecución de la aplicación es mediante el cambio del valor 0 de la primera línea del archivo "Fin.txt" por 1.

3.3.4 Análisi

Multitud de ejecuciones de ambas herramientas han sido realizadas tanto durante su desarrollo como después de estar terminadas con el afán de estudiar el comportamiento que tienen. Mediante dichas ejecuciones y la observación de sus resultados se puede evaluar ambas herramientas en relación a la calidad de los resultados proporcionados y sus limitaciones.

Iniciaremos este apartado con el análisis de la herramienta ObtenerNodos. Por lo que a la calidad del resultado que proporciona cabe destacar que aún siendo relativamente buenos nos son ideales. Esto es debido en gran medida al rendimiento de la búsqueda de nodos. El tiempo que consume para encontrar todos o un número muy alto de nodos disponibles es demasiado elevado. ejemplo de esto es como la ejecución de la herramienta durante 1-2 horas únicamente se obtiene alrededor de 100 nodos o de 1000 durante 12 horas. Esto provoca que conexiones que encontró como disponibles desaparezcan y que de nuevas aparezcan y puedan no ser detectadas. Degradando mucho el resultado que afirma que esos nodos son accesibles aunque igualmente pueden ser utilizados por la herramienta siguiente. En resumidas cuentas hay que destacar que aunque la calidad del resultado es aceptable está limitado por el rendimiento que tiene esta aplicación en cuanto a su velocidad de consultar nodos.

El análisis que resta es a MonitorizarNodos. Hay que destacar que los resultados que se obtiene en esta herramienta se ve afectado por la calidad de los obtenidos con la herramienta anterior con lo cual su calidad ya se ve en parte reducida. Diversas ejecuciones se han realizado variando la cantidad de tiempo empleado en la anterior ejecución. Cabe destacar que en ninguna de las ejecuciones se han obtenido buenos resultados y que la ejecución del programa, aunque en ciertos casos ha detectado candidatos a nodos mineros la aplicación termina por eliminar estas conexiones. Esto se estima que es debido a que el tiempo de vida de una conexión es bajo, junto con la gran mayoría que no aceptan más conexiones que permitan usar el sniffer y controlar los envíos de paquetes lleva a un final de la ejecución desastroso en que la aplicación se queda sin ningún candidato posible que pueda ser monitorizado y por lo tanto no determina, de los nodos en la red, aquellos que son mineros.

Lamentablemente hay que determinar que la aplicación diseñada no cumple con el objetivo del trabajo debido a un diseño deficiente. Se teoriza que el diseño ha resultado deficiente en realizarse sin la suficiente experiencia previa en intentos de interactuar con la red Bitcoin y monitorizar la. Lo cual aún ha supuesto mayor dificultad en haber comenzado el trabajo sin poseer ningún conocimiento o haber tenido contacto con Bitcoin en un trabajo para el que previamente ya debería disponerse de experiencia.

CAPÍTULO 4

Objetivos Opcionales

A parte del objetivo principal del trabajo que era desarrollar una herramienta que detectase los nodos mineros de la red existían otros dos objetivos. Estos objetivos si bien en un principio se veía complicado poder realizarlos ya se pensó en las vías posibles de abordaje. Aunque con la herramienta desarrollada no se ha logrado los resultados esperados por lo que ha invalidado la implementación de los diseños ideados para cada uno de estos objetivos, los diseños continúan siendo válidos a ser considerados en vías futuras.

Un primer objetivo era obtener información sobre los nodos mineros. El protocolo Bitcoin implementa un tipo de mensaje, "version", que comunica información tanto del software del cliente, como del tamaño de su cadena de bloques, hora del sistema, etc. Esto puede incrementarse si la monitorización se realiza correctamente. Esto haría posible teorizar el hashrate de los nodos mineros. El proceso consistiría y tendría los requisitos siguiente: Se añadiría la necesidad, a la herramienta MonitorizarNodos, del tiempo en que los nodos han estado conectados. Se hace necesario controlar el número total de nodos mineros conectados durante la propagación de cada bloque. A partir del número de veces en que este nodo minero ha propagado un bloque por primera vez y de las veces en que ha estado conectado pero no ha sido el primero en propagarlo de determinar su tasa de éxito como: nºminado / (nºminado + nºnominado). Esta tasa de éxito puede ser calculada para todos los nodos mineros y comparada. Mediante la comparación con tasas de éxito de mineros de los cuales sepamos su hashrate con la de los desconocidos se pude inferir su potencia de cálculo. Esto es posible debido a que: HashRateConocido x TasaExitoConocido ¡ HashRateDesconocido x TasaExitoDesconocido cuand el nodo conocido es una piscina. Estableciéndose un nivel tope probabilístico de su hashrate. Con esto el primer objetivo secundario quedaría resuelto.

A continuación restaría el segundo objetivo secundario que busca detectar comportamientos anómalos. La detección de este comportamiento anómalo se podría realizar mediante el control

de si algún nodo de la red ha provocado en más de una ocasión alguna intersección y si ha sido en respuesta siempre a la publicación de otros bloques. También si el mismo nodo ha llegado a decidir el camino aceptado por la red al publicar inmediatamente nuevos nodos siguiendo su camino. Esto podría ser identificado como un ataque contra el sistema si se ha repetido varias veces demostrando una competencia desleal en la minería por su parte. Otros mecanismos de detección pueden ser implementados como controlar la propagación de bloques falsos por clientes, los cuales no tienen relación con la cadena o determinar el número de conexiones que mantiene un cliente con otros clientes y si esta es muy alta identificarlo como un nodo que trata de monitorizar la red.

CAPÍTULO 5

Conclusion

Durante la realización de este trabajo se ha podido sumergirse en el mundo de Bitcoin el cual se ha podido ver unicmente una parte de su gran alcance. Se ha realizado tanto el primer contacto con la parte superficial de él como con aspectos más avanzados y obtenido una experiencia y conocimientos muy valiosos. Entre los conocimientos obtenidos destacan las caracteristicas de los principales elementos que forman el sistema Bitcoin (clientes, transacciones, red, bloques, mineros, etc.) e incluso se ha obtenido un mayor conocimiento del lenguaje de programación Python, del cual aunque era conocido previamente no hasta el grado en que se ha hecho necesario. La experiencia obtenida en trabajar con la red Bitcoin real tambien ha sido muy grande obteniendose una comprension mayor de los distintos escenarios posibles que pueden presentarese y que aunque se pueda no trabajar más con esta red la experiencia puede ser trasladada a otras también.

El trabajo tenia como objetivo principal la realización de una herramienta de facil uso que permitiese monitorizar los nodos de la red bitcoin por cualquier usuario. Si bien un duo de herramientas han sido dessarrolladas con este fin, la calidad de los resultados que proporcionan como conjunto es demasiado deficiente como para poder considerar haber logrado el objetivo fijado. Es por este motivo que entre las posibles vias de trabajo futuro, la primera y esencial sera la mejora del funcionamiento de la herramienta MonitorizarNodos.

5.1 Trabajos Futuros

- Como anteriormente se ha dicho y debido a los malos resultados el primer y esencial trabajo futuro que deberia ser realizado es, sino el rediseño, si la reimplementación de la herramienta MonitorizarNodos. La reimplementación buscaria un producto que generase resultados aceptables.

- Una pequeña parte del problema con los resultados ha vendio del poco rendimiento presentado por las aplicaciones. Con tal de corregir este fallo es posible que fuese necesario rediseñar la aplicaccion con tal de permitir que esta aumentase su rendimiento.

- Separado del objetivo principal se encontraban dos de secundarios los cuales unicamente han sido presentados superficialmente. Trabajar, una vez los dos anteriores hubiesen sido completados, en estos y abrirn nuevas posibles vias de actuación puede suponer un buen reto en esta materia.

- El ultimo trabajo futuro consiste en la satistfacción de la propia curiosidad surgida durante la inmesion en el tema del trabajo en ciertos temas. Estos temas si bien tratan los aspectos más maematicos y los de más bajo nivel del sistema Bitcoin y esta autosatisfacción puede considerarse que no merece ser incluida como un trabajo, debido a que por experiencia el obtener la comprension de partes priori no relacionadas con el tema pueden llegar a abrir nuevas vias y mejores de las disponibles sin haber sido abordado.

Bibliography

[1] AFP. China prohíbe el bitcoin en las tiendas 'online' y su valor cae un 65%. http://tecnologia.elpais.com/tecnologia/2013/12/18/actualidad/1387361060_150101.html. Online; 18-Diciembre-2013.

[2] Lorenzo Alberton. Modern algorithms and data structures - 1. bloom filters, merkle trees. http://es.slideshare.net/quipo/modern-algorithms-and-data-structures-1-bloom-filters-merkle-trees. Online; 17-Abril-2011.

[3] Andreas M. Antonopoulos. Chapter 6. the bitcoin network. http://chimera.labs.oreilly.com/books/1234000001802/ch06.html. Online; 28-Mayo-2015.

[4] Andreas M. Antonopoulos. Chapter 8. mining and consensus. http://chimera.labs.oreilly.com/books/1234000001802/ch08.html. Online; Diciembre-2014.

[5] BLOCKCHAIN.INFO. Mi monedero sé tu propio banco. https://blockchain.info/es/wallet. Online; 20-Mayo-2015.

[6] BLOCKCHAIN.INFO. Precio de mercado (usd). https://blockchain.info/es/charts/market-price?timespan=all. Online; accedido 20-Mayo-2015.

[7] Sebastian Castro. Bitcoin p2p network sniffer v0.0.2. https://github.com/sebicas/bitcoin-sniffer. Online; 20-Mayo-2015.

[8] Bitcoin Wallet developers. Bitcoin wallet. https://play.google.com/store/apps/details?id=de.schildbach.wallet. Online; 20-Mayo-2015.

[9] EFE. Una empresa formada por españoles lleva los bitcoins a los cajeros tradicionales. http://www.eleconomista.es/emprendedores-innova/noticias/6569561/03/15/Una-empresa-espanola-lleva-los-Bitcoins-a-los-cajeros-tradicionales.html#.Kku83qUdvhcT2BY. Online; 20-Marzo-2015.

[10] ÁNGEL LUIS SUCASAS FERNÁNDEZ. Microsoft permite el 'bitcoin' en estados unidos. http://tecnologia.elpais.com/tecnologia/2014/12/11/actualidad/1418315457_741748.html. Online; 11-Diciembre-2014.

[11] Bitcoin Foundation. Descargar bitcoin core. https://bitcoin.org/es/descargar. Online; 20-Mayo-2015.

[12] Dirk Merkel. Bitcoin for beginners, part 3: The bitcoinj api. http://www.javaworld.com/article/2078482/java-web-development/bitcoin-for-beginners--part-3--the-bitcoinj-api.html. Online; 10-Enero-2012.

[13] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. https://bitcoin.org/bitcoin.pdf, 2009.

[14] SANDRO POZZI. El colapso de mt. gox expone los riesgos de un mercado sin regular. http://tecnologia.elpais.com/tecnologia/2014/02/25/actualidad/1393336959_827678.html. Online; 25-Febrero-2014.

[15] Alex Preukschat. ¿cómo se crea una dirección o clave pública en bitcoin? (x). https://www.oroyfinanzas.com/2014/01/como-crea-direccion-clave-publica-bitcoin/. Online; 21-Enero-2014.

[16] pyspanishdoc. 7.5 threading – interfaz multihilo de alto nivel. http://pyspanishdoc.sourceforge.net/lib/module-threading.html. Online; 20-Mayo-2015.

[17] Python. 15.3. time — time access and conversions. https://docs.python.org/2/library/time.html. Online; 20-Mayo-2015.

[18] Bitcoin wiki. Block. https://en.bitcoin.it/wiki/Block. Online; 17-Mayo-2015.

[19] Bitcoin wiki. Difficulty. https://en.bitcoin.it/wiki/Difficulty. Online; 27-Mayo-2015.

[20] Bitcoin wiki. Mining. https://en.bitcoin.it/wiki/Mining. Online; accedido 28-Mayo-2015.

[21] Bitcoin wiki. Technical background of version 1 bitcoin addresses. https://en.bitcoin.it/wiki/Technical_background_of_version_1_Bitcoin_addresses. Online; 21-Enero-2015.

[22] Bitcoin wiki. Trade. https://en.bitcoin.it/wiki/Trade. Online; accedido 20-Mayo-2015.

[23] Bitcoin wiki. Transaction. https://en.bitcoin.it/wiki/Transaction. Online; 28-Mayo-2015.

[24] Bitcoin wiki. Transaction fees. https://en.bitcoin.it/wiki/Transaction_fees#Including_in_Blocks. Online; 6-Mayo-2015.

[25] Wikipedia. Criptografía de curva elíptica. http://es.wikipedia.org/wiki/Criptograf%C3%ADa_de_curva_el%C3%ADptica. Online; 17-Diciembre-2014.

[26] Wikipedia. Merkle tree. https://en.wikipedia.org/wiki/Merkle_tree. Online; 3-Mayo-2015.

[27] Addy Yeow. Bitnodes. https://github.com/ayeowch/bitnodes. Online; 20-Mayo-2015.

Appendices

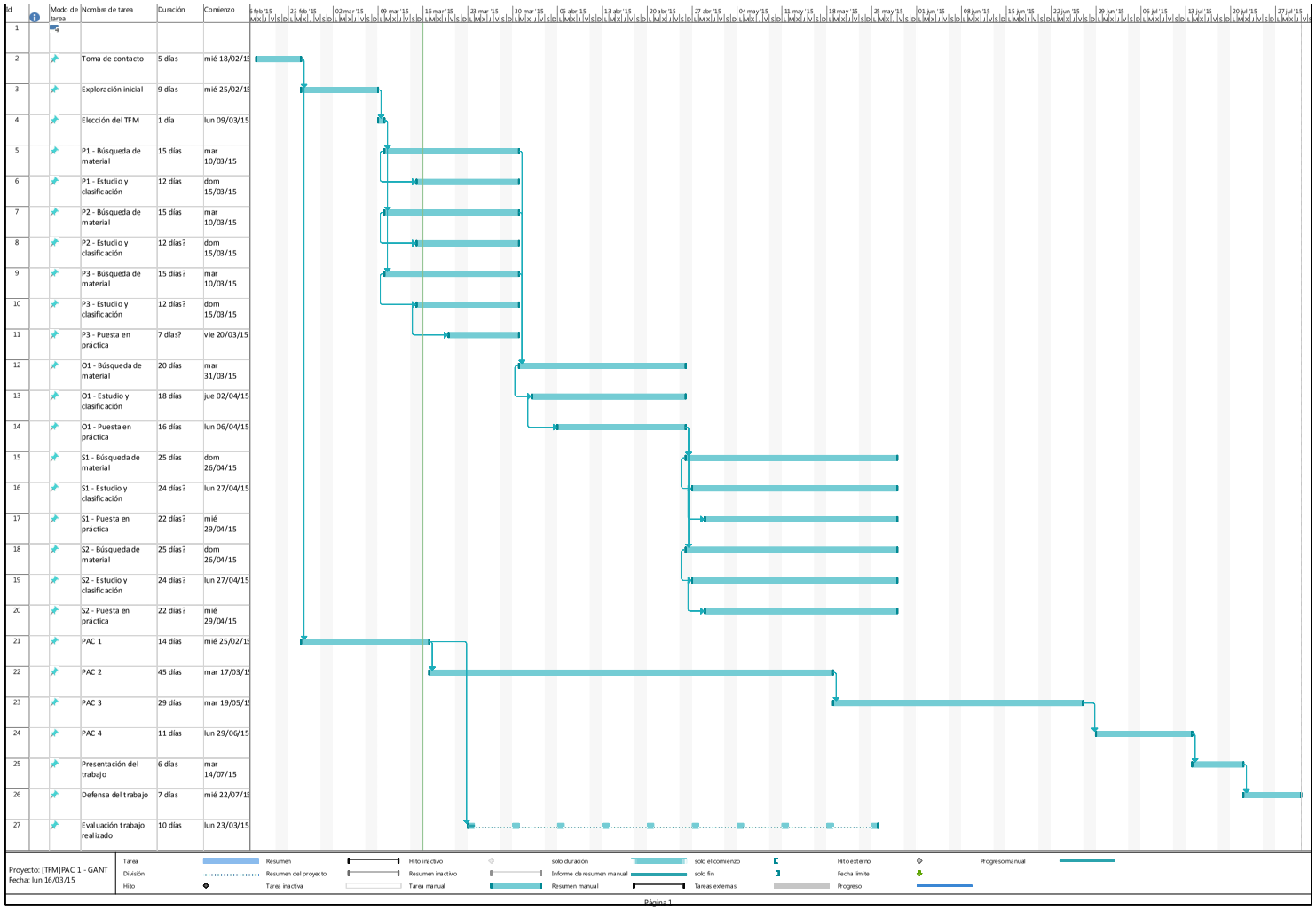

Figura 5.1: Anexo 1: Planing

Rubén Pérez
Sabadell, 2015