Proyecto de Grado
Relevamiento de técnicas de rastreo y entintado de dinero en Bitcoin

Lucas Bouissa - 4784424 – 1

Fabricio Garín - 4582413 – 0

Alex Rostán - 4377187 – 2

Tutores: Germán Ferrari - Alfredo Viola

Informe de Proyecto de Grado presentado al Tribunal Evaluador como requisito de graduación de la carrera Ingeniería en Computación

Ingeniería en Computación
Montevideo 2019

Agradecimientos

Este trabajo no hubiese sido posible sin el apoyo y la contribución brindada por varias personas que de alguna manera formaron parte de su culminación. En especial, quisiéramos agradecer a nuestros directores de tesis, el Dr. Ing. Alfredo Viola y al Ing. Germán Ferrari por la confianza y la guía constante en este trabajo, y por el interés y la motivación que en todo momento nos brindaron. Con gran cariño también queremos agradecer a nuestras familias y amigos que han sido un pilar fundamental en nuestras vidas y en nuestros años de estudio, por su paciencia y apoyo constante en los momentos más difíciles.

Resumen

Bitcoin se presentó como una solución descentralizada (o sea sin utilizar un Banco que actúe como intermediario) para hacer transacciones monetarias, presentando maneras interesantes y muy ingeniosas para resolver el problema del doble gasto. La estructura de datos distribuída se le llamó blockchain, y el protocolo de uso tiene la gran virtud de que es mucho menos costoso para los usuarios actuar honestamente que intentar quebrarlo.

Por otro lado una de sus propiedades radica en que las transacciones no se identifican con nombres de usuarios, sino que tanto quienes transfieren como quienes reciben bitcoins se identifican con el hash de una clave pública. Es importante recalcar que un usuario puede tener muchas claves públicas, y por tal motivo puede transferir dinero a si mismo, usando diferentes identificadores, y tratando de evitar así cualquier tipo de correlación entre cada usuario y las monedas que recibe o transfiere. En este contexto estamos tentados a decir que bitcoin ofrece anonimato.

Sin embargo, si bien no hay una relación directa, dada que todas las transacciones son públicas, se puede hacer un rastreo de las transacciones, y por tal motivo se dice que lo que presenta es un pseudoanonimato (las transacciones son anónimas, pero el comportamiento puede indicar que hay una relación fuerte entre ciertas direcciones y ciertas personas o instituciones).

En este proyecto, se relevan diferentes propuestas algorítmicas de rastreo, y al final (con el fin de entender más claramente su comportamiento) se analiza uno de estos algoritmos. Más específicamente se analiza la implementación de un algoritmo presentado por un grupo de investigación de la Universidad de Cambridge que simula el entintado de monedas, para rastrear las direcciones por las cuales pasa en su camino. A partir del código generado por ellos se realizaron pruebas para validar empíricamente este algoritmo.

Palabras clave: Bitcoin, Entintado FIFO, Rust, Anonimato, Clusterización, Rastreo.

Contenido

1 Introducción

Las formas de pago han variado a lo largo de la historia, partiendo desde el trueque y evolucionando hasta el actual sistema controlado por entidades centralizadas como los bancos. En la actualidad, gran cantidad de las transacciones monetarias se realizan en internet y, para ésto, se deben basar en instituciones financieras que actúan procesando dichas transacciones. No existe forma de hacer un pago en un canal de comunicación sin la intervención de una tercera parte. Históricamente han existido numerosos intentos de resolver este problema mediante sistemas criptográficos [1], tanto soluciones basadas en tecnologías de tarjetas de crédito como soluciones del estilo moneda electrónica. Algunos de estos sistemas son propuestas académicas mientras que otros han sido casos desarrollados y testeados. Decenas son los casos listados tan sólo en [1], de los cuales el único que se puede destacar que ha logrado prevalecer gracias a mutar rápidamente es Paypal.

En el año 2008, es presentada una nueva forma de moneda por medio de un artículo llamado "Bitcoin: A Peer-to-Peer Electronic Cash System" [2] a cargo de Satoshi Nakamoto, de quien en la actualidad poco se sabe, ni siquiera si es una persona, un pseudónimo o una comunidad. Nakamoto hace pública la implementación descrita en el artículo en 2009, dándole nacimiento a la moneda virtual Bitcoin, también fue el encargado de la participación en los foros y las primeras actividades de minería lo cual se estima que le proporcionó alrededor de 980000 BTC (Bitcoins), una verdadera fortuna a día de hoy, aunque en el año 2010 desapareció, dejando sin utilizar todas y cada una de sus Bitcoins.

Bitcoin es una moneda virtual que gracias a su novedoso mecanismo de pseudo anonimato, su transparencia reflejada en la *blockchain* y su inteligente forma de comprobar que las transacciones nuevas no tengan problemas de doble gasto ni inconsistencias, ha logrado posicionarse entre las principales criptomonedas, concepto dado ya que es una moneda virtual que hace uso de la criptografía. Todo esto, sumado a que puede ser usada como forma de pago, ha despertado el interés en ser utilizada para para transferir dinero proveniente de ilícitos. Dentro de estos ilícitos, los principales son por ejemplo el robo de Bitcoins, lavado de dinero, extorsión e innumerables servicios que se encuentran en la deep web [3] que hacen uso del pseudo anonimato de Bitcoin para ofrecer las cosas más variadas que una mente pueda pensar, desde pornografía infantil a asesinos a sueldo y todo tipo de drogas sólo por citar algunas. Esto presenta la pregunta, si dado un Bitcoin que se le compra a cambio de su equivalente en dinero actual a una entidad de cambio o que se obtiene por la venta de un servicio, este Bitcoin tiene chance de ser el resultado de alguna de estas tantas situaciones ilícitas.

Se decide investigar si dado un Bitcoin que se obtiene como forma de pago,

reconocer si forma parte en algún momento de una transacción ilícita, ya sea porque está implicada directamente en un ilícito o participa de una transacción donde hay una que sí lo es.

Para poder comprender mejor el problema, se marca como objetivo llevar a cabo un relevamiento de diferentes propuestas orientadas al rastreo de dinero en Bitcoin. Luego de observar que muchas de éstas realizaban clusterización de direcciones, se decide elegir una de éstas tácticas, el entintado FIFO. Se hace un análisis de la implementación del algoritmo realizada por un grupo de investigadores de la universidad de Cambridge [4] [5], con el objetivo de comprender la forma en que fue llevada a cabo y validar si con dicha implementación se puede determinar si una dirección formó parte de un ilícito (está entintada) o no (está limpia).

El resultado esperado de mayor importancia es el de entender la dificultad del problema y las diversas soluciones propuestas y documentarlas apropiadamente. Como consecuencia de este análisis previo, se espera poder comprender cómo hace el entintado FIFO en Bitcoin para manchar direcciones y finalmente poder ver que a partir de direcciones marcadas como participantes en algún ilícito, se marcan todas las direcciones que hubieran participado directa o indirectamente en alguna transacción con las anteriores utilizando el algoritmo de entintado FIFO. En cambio, una dirección que no haya participado debería aparecer limpia.

Se pudo sacar como conclusión que el método de entintado FIFO desarrollado por el equipo de Cambridge, puede responder la pregunta de si una Bitcoin recibida, fue en algún momento partícipe de un ilícito. Con respecto a los artículos analizados referentes al anonimato y clusterización de direcciones en Bitcoin, se aprendió que mayormente el clusterizado se realiza utilizando dos heurísticas, agrupación de entradas y dirección de cambio, mientras que para fortalecer anonimato se han propuesto diferentes metodologías donde cada una tiene sus pros y sus contras, además es una materia en constante evolución.

El presente documento se organiza de la siguiente manera. El capítulo 2 presenta los principales conceptos de importancia para comprender este trabajo. En el capítulo 3 se presentan detalles de las transacciones en Bitcoin importantes para entender la implementación estudiada. En el capítulo 4 se estudia el anonimato en Bitcoin. En el capítulo 5 se analiza el rastreo de direciones en Bitcoin. En el capítulo 6 se detalla el trabajo realizado analizando la implementación del algoritmo de entintado FIFO y las pruebas hechas. En el capítulo 7 se presentan las conclusiones obtenidas y las líneas de trabajo a futuro que se observan.

2 Marco Teórico

En esta sección se comenzará explicando los principales conceptos que son importantes para comprender este trabajo, se realiza un enfoque en la criptomoneda Bitcoin aunque algunos conceptos sean compartidos con otras criptomonedas.

Criptomoneda Una criptomoneda es una moneda virtual convertible y descentralizada [6], basada en matemática que está protegida por criptografía. Dependen de claves públicas y privadas para transferir valor de un individuo a otro y debe ser firmada criptográficamente cada vez que se transifere.

Bitcoin Bitcoin es la primer criptomoneda [6] [2], nació en el año 2009, en algunas jurisdicciones puede usarse como medio de pago para adquirir bienes y servicios. La diferencia con la moneda corriente es que es una divisa electrónica que presenta novedosas características y se destaca por su eficiencia, seguridad y facilidad de intercambio. Su mayor diferencia frente al resto de monedas, es que se trata de una moneda descentralizada, por lo que en teoría nadie la controla. Bitcoin no tiene un emisor central como los dólares o los euros, la criptomoneda es producida por las personas y empresas alrededor del mundo dedicando gran cantidad de recursos a la minería de la misma, o sea, a la creación de nuevas monedas y validación de nuevas transacciones. Muchas son las regulaciones que se le han intentado impartir a Bitcoin, de hecho, diferentes entes regulatorios le dan motes distintos, por ejemplo, la regulación en Japón la toman como una moneda virtual, las autoridades del Reino Unido la tratan como una moneda extranjera cuando a impuesto al valor agregado se refiere, mientras que la tratan como mercancía en el impuesto a los ingresos. En la jurisdicción de Estados Unidos es tratada como una mercancía, por otra parte en Uruguay se la ve como activos financieros según palabras de Mario Bergara [7].

Blockchain La *Blockchain* [1] es una base de datos compartida que recopila todas transacciones hechas en Bitcoin. La idea se remonta a Haber and Stornetta [8, año 1991], su propuesta inicial era un método de datado de documentos digitales, donde el datado es una idea de cuando el documento fue creado. Es la base tecnológica del funcionamiento de Bitcoin.

Minería La minería [1] es un sistema de consenso distribuido [9] que se utiliza para confirmar las transacciones pendientes a ser incluidas en la cadena de bloques *(Blockchain)*. Hace cumplir un orden cronológico en la *blockchain*, protege la neutralidad de la red y permite un acuerdo entre todos los equipos sobre el estado del sistema. Para confirmar las transacciones, deberán ser agregadas en un bloque que se ajuste a estrictas normas de cifrado y que será verificado por la red. Estas normas impiden que cualquier bloque anterior se modifique, ya

que hacerlo invalida todos los bloques siguientes. La minería también crea el equivalente a una lotería competitiva que impide que cualquier persona pueda fácilmente añadir nuevos bloques consecutivamente en la cadena de bloques. De esta manera, ninguna persona puede controlar lo que está incluido en la cadena de bloques o reemplazar partes de la cadena de bloques para revertir sus propios gastos. Un minero se une a la red de Bitcoin y se conecta con los demás nodos, luego participa de algunas tareas que lo consolidan como minero, éstas son: escucha por transacciones que sean enviadas por demás usuarios de la red y las valida, chequeando que las firmas son correctas y que las salidas no hayan sido gastadas antes. Debe mantener la *blockchain* al día y su trabajo es conseguir nuevas transacciones que sean emitidas y armar un nuevo bloque para expandir el último bloque que tiene en la *blockchain*. Tal vez el trabajo más importante que realiza un minero es agrupar las transacciones en un bloque y buscar un valor que sumado al cabezal del bloque haga que el hash del bloque sea válido. Como se puede ver en Nakamoto [2, Capítulo 4], en *Proof-of-work* como se llama este mecanismo, el minero busca un valor tal que el hash del bloque más este valor esté por debajo de cierto valor de referencia positivo. El trabajo promedio que se tiene que hacer crece exponencialmente a medida que el valor se aproxima a cero. Si la mayoría del poder computacional está en manos honestas, el sistema se mantiene. Un atacante que quisiera modificar un bloque dado debería encontrar el valor no sólo de este bloque, sino también de todos los sucesivos, haciendo practicamente imposible su éxito. Luego hay que esperar que el bloque minado sea aceptado por la comunidad, en cuyo momento el bloque ya es parte de la *blockchain* y el minero recibe finalmente su premio en Bitcoins tanto por haber minado el bloque, como además un extra cuando alguna transacción deja una tasa de premio por haber sido incluída a la *blockchain*.

Bitmain tiene casi el monopolio del minado, se estima que gana anualmente 4 billones de dólares [10], de ésto se puede ver que no es nada fácil entrar al mundo de mineros en Bitcoin.

Con respecto al minado, una de las recomendaciones para mejorar Bitcoin que se presentan en [4], implica que se regularicen tasas contra el minado *proof-of-work* por ser contraproducente para el medio ambiente.

Bloque Un bloque es un registro en la cadena de bloques que contiene confirmaciones de transacciones pendientes. Aproximadamente cada 10 minutos, un nuevo bloque que incluye nuevas transacciones se anexa a la *blockchain* a través de la minería. Cada bloque tiene un conjunto de transacciones, que son recolección de distintas transacciones emitidas por otros usuarios y no confirmadas anteriormente. El cabezal de cada uno de los bloques incluye un hash criptográfico del cabezal del bloque anterior de la *blockchain*. De esta manera cada bloque está encadenado al

anterior, generando una cadena desde el "bloque génesis" (o "bloque cero") hasta el bloque actual. Esto asegura la integridad de toda la *blockchain*. Aunque los bloques no son inalterables, modificar uno implicaría generar nuevamente todos los subsiguientes bloques de la cadena, lo cual es computacionalmente inviable.

Transacción Una transacción es una estructura de datos que codifica la transferencia de valores entre participantes de Bitcoin y que será incluida en la cadena de bloques. Para más información dirigirse al capítulo 3.

Direcciones Las direcciones en Bitcoin son hashes de la clave pública que es propia a cada usuario [1]. Una dirección es la identidad detrás de Bitcoin. Un usuario puede elegir trabajar siempre con la misma dirección o clave pública como se pueden referenciar indistintamente, pero puede generar tantas direcciones como quiera, ya que es una de las ventajas que proporciona Bitcoin para aumentar el anonimato de los usuarios y además es practicamente imposible que dos usuarios generen la misma clave de 256 bits. En la práctica por lo general un usuario hace uso de una billetera o *Wallet* para administrar todas sus direcciones.

Billetera *Wallet* o billetera de Bitcoin [1] es un programa que mantiene todas las monedas del usuario, maneja todos los detalles de las claves y presenta una interfaz para facilitar las cosas. Cuando uno quiere gastar sus Bitcoins, se ocupa de los detalles de que clave usar y cómo generar nuevas direcciones.

Rust Rust es un lenguaje de programación compilado, de propósito general y multiparadigma centrado en la seguridad, especialmente la concurrencia segura [11]. Es sintácticamente similar a C++, pero está diseñado para proporcionar una mejor seguridad de la memoria mientras se mantiene un alto rendimiento. Cuenta con un compilador estricto que se asegura de que se esté usando la memoria de manera segura. Este manejo rápido y eficiente en cuanto a memoria permite un análisis del uso de ésta en tiempo de compilación y la no utilización de *garbage collector* en tiempo de ejecución. Debido a ésto resulta útil para sitemas para los cuales es crítico su rendimiento y manejan grandes cantidades de memoria, como lo son los que trabajan sobre la *blockchain* y en particular el algoritmo analizado en la Sección 6. *Rust* se organiza por medio de los Crates. Los Crates son un conjunto de módulos *Rust* que se agrupan para formar paquetes o librerías, también llamados bibliotecas en otros lenguajes de programación. Finalmente *Rust* cuenta con una herramienta especial llamada Cargo, la cual se encarga de mantener las dependencias actualizadas, hacer uso del compilador como puede ser *rustc* y declarar convenciones para que trabajar con paquetes *Rust* sea más fácil.

Mixer Un *mixer* de criptomonedas [1], es un servicio que se ofrece para mezclar criptomonedas potencialmente identificables con otras, a fin de ocultar el rastro a

la fuente original. A diferencia que con las billeteras, los *mixers* prometen no mantener registros y ni siquiera pedir la identidad de quien deposita sus Bitcoins. Para interactuar con el *Mixer* no se necesita ni seudónimo ni identidad. Tan sólo se le envían los Bitcoins a una dirección publicada por el *Mixer* y se le da una dirección propia en la cual recibir los Bitcoins. Con fortuna, el *Mixer* enviará a la dirección que se le dio "otros" Bitcoins. Esencialmente se podría ver como un intercambio. Detalles que hay que tener en cuenta para el funcionamiento de los *mixers* es por ejemplo, que la cantidad de Bitcoins que un usuario envía en cada caso debería ser la misma que la que mandan los demás participantes del *mixer*, ésto es porque sería muy evidente si el participante 1 deposita X y el participante 2 deposita Y, luego en la salida del *mixer*, tenemos una por un monto de Y sumada a otra salida por X, por más que el orden de salida no sea igual al de entrada, se podría hacer una relación inmediata. En el capítulo 4 se extiende la información y se presentan distintas variantes para fortalecer el anonimato por éstos brindados. Los *mixers* nacieron como nuevos sistemas para fortalecer el anonimato, para aquellos buscando evitar ser atrapados por las legislaciones comportándose ilegalmente. Dentro de estos esquemas se pueden encontrar *mixers* como *Bitcoinfog [12], CoinJoin [13] y Tumblebit [14]*. Las modalidades de algunos de éstos utilizan *ring signature* o *smart contracts*.

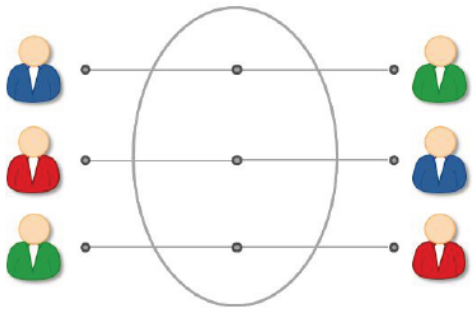

Fig. 1. Mixer [1]

En la figura 1 se pueden observar tres usuarios representados por los distintos colores que depositan un Bitcoin por ejemplo en el *mixer* y que luego cada usuario obtiene un Bitcoin distinto al que había depositado en un principio, o sea, mezcló sus Bitcoins con la de los demás usuarios.

Exchanges Los *exchanges* [4] [1], son los principales entes encargados tanto del intercambio entre distintas criptomonedas, entre criptomonedas y monedas corrientes, como también funcionan administrando las monedas de los usuarios. Hay tres modalidades en la que un usuario puede comprar y administrar sus Bitcoins mediante éstos. Comprársela a un *exchange* y hacerla dirigir a una billetera administrada por dicho usuario de cuya *private key* es dueño. Comprársela a un *exchange* pero permitir que éste mantenga tanto la suma como la *private key* para manejar dichas Bitcoins. Comprársela a un *exchange* y mantenerla en una cuenta de la que el usuario es propietario y el *exchange* guarda con sus propios Bitcoins para él (como un banco).

Según un estudio realizado sobre algunos *exchanges* por [4], su estatuto no es nada claro sobre cuál de las tres modalidades están constituidos. Posterior análisis sobre el *exchange* más importante en UK, *Coinbase*, mostró que los montos reportados no reflejan lo que debería ser el mercado.

Dentro de sus conclusiones en Anderson et al. [4], se dejan varias recomendaciones que se deberían agregar para mejorar Bitcoin. La primera recomendación dice que la ley debería regular los *exchanges*. La segunda recomendación es que la relación entre los *exchanges* y sus clientes debería ser cubierta por la segunda directiva de servicio de Pago *(Payment Service Directive)* [15], que es una directiva europea para regular los servicios de pago y sus proveedores. La tercera recomendación es que los *exchanges* regulados no deberían hacer transacciones con aquellos no regulados. Cuarta recomendación, que los *exchanges* dejen claro en el contrato si son custodios del dinero del cliente o si el cliente sólo tiene un poder sobre una parte de la que éstos son dueños. Quinta recomendación, los reguladores no deberían permitir que los *exchanges* compren y vendan criptomonedas que no permiten el rastreo y se utilizan para el financiado del terrorismo. Sexta recomendación, que los *exchanges* regularizados sean adecuadamente capitalizados, con claros estándares para su control.

Además en el artículo se observa que regular los *exchanges* es importante, estudios muestran que en 2013 de 40 *exchanges*, 18 ya habían cerrado y estudios posteriores en 2017 prueban que ésto no ha cambiado. Con respecto a regulación, en 2013 el gobierno estadounidense hizo registrar todos los *exchanges* del país, quedándose con cuentas estadounidenses de MT.Gox. Hablando de este *exchange* en febrero 2014 MT.Gox se declaró en bancarrota alegando que BTC 744000 le fueron robados, convirtiéndose éste en el robo más grande al momento en cantidad de Bitcoins.

Firma Digital Una firma digital es el análogo a un firma manual sobre papel [1]. Son deseadas dos propiedas en las firmas digitales. Primero, sólo uno puede hacer su firma,

pero todos los que la ven pueden verificar que es válida. Segundo, se quiere que la firma esté asociada a un documento en particular, de forma tal que la misma firma no se puede utilizar para otro documento. En las firmas manuales en papel, esto último se corresponde con asegurar que nadie pueda agarrar nuestra firma y pegarla en otro documento. Los algoritmos necesarios para definir una firma digital son los siguientes:

* **(sk, pk) = generarClaves(tamaño)** este método recibe un tamaño y genera un par de claves. La clave secreta sk es mantenida privada y utilizada para firmar mensajes. pk es la clave pública que cualquiera puede tener y con ésta verificar la firma.
* **firma = firmar(sk, mensaje)** este método recibe un mensaje y una clave secreta (sk) y produce como salida una firma para el mensaje bajo la clave privada sk.
* **esVálido = Verificar(pk, mensaje, firm)** este método recibe una clave pública pk, un mensaje y una firma. Retorna un booleano esVálido que será verdadero si firm es una firma válida para el mensaje bajo la clave pública pk y falso en caso contrario.

Hay dos propiedades que se deben mantener en las firmas:

* Verificar(pk, mensaje, firmar(sk, mensaje)) = verdadero
* Las firmas deben ser imposibles de falsificar.

3 Transacciones

El capítulo presenta la estructura de las transacciones y el mecanismo mediante el cual se asigna implícitamente en esta la recompensa de los mineros, lo cual tiene especial importancia en el entintado de direcciones que se estudia en el Capítulo 5. Como afirma Antonopoulos [16] las transacciones son una de las partes más importantes del sistema Bitcoin; todo lo demás en Bitcoin está diseñado para garantizar que las transacciones puedan crearse, propagarse en la red, validarse y, finalmente, agregarse a la *blockchain*. Las transacciones son estructuras de datos que codifican la transferencia de valor entre los participantes en el sistema de Bitcoin.

Cada transacción es una entrada pública en la *blockchain* de Bitcoin. Una vez que es creada una transacción ésta se firma con una o más firmas, que indican la autorización para gastar los fondos a los que hace referencia la transacción y luego se transmite en la red Bitcoin, donde cada nodo de la red valida y propaga la transacción hasta que llega a la mayoría de los nodos de la red. Finalmente, la transacción es verificada por un nodo de minería de datos y se incluye en un bloque de transacciones que se registra en la *blockchain*.

Una vez registrada en la cadena de bloques y confirmada por suficientes bloques subsiguientes la transacción es una parte permanente de la *blockchain* de Bitcoin y se acepta como válida por todos los participantes. Los fondos asignados a un nuevo propietario por la transacción se pueden gastar en una nueva transacción, extendiendo la cadena y comenzando nuevamente el ciclo de vida de una transacción.

3.1 Estructura de transacción

Una transacción es una estructura de datos que codifica una transferencia de valor de una fuente de fondos, denominada *input*, a un destino, llamado *output*. Las entradas y salidas de transacciones no están relacionadas con cuentas o identidades. En su lugar, se debe pensar en ellos como cantidades de Bitcoin (trozos de Bitcoin) que están bloqueados con un secreto específico que sólo el propietario, o la persona que conoce el secreto, puede desbloquear. El componente fundamental de una transacción de Bitcoin es un resultado de transacción no gastado o *UTXO (Unspent Transaction Output)*.

Los *UTXOs* son trozos indivisibles de moneda Bitcoin bloqueados a un propietario específico, grabados en la cadena de bloques y reconocidos como unidades de moneda por toda la red.

Cuando un usuario recibe Bitcoins, esa cantidad se registra dentro de la cadena de bloques como un *UTXO*. Por lo tanto, el Bitcoin de un usuario puede estar distribuido como *UTXOs* entre cientos de transacciones y cientos de bloques. En

efecto, no existe un saldo almacenado de una dirección o cuenta de Bitcoin; sólo hay *UTXOs* dispersos, bloqueados a propietarios específicos. El concepto de balance de Bitcoin de un usuario es una construcción derivada creada por la aplicación de billetera. La billetera calcula el saldo del usuario escaneando la cadena de bloques y agregando todos los *UTXO* que pertenecen a ese usuario.

Una *UTXO* puede tener cualquier valor arbitrario denominado como múltiplo de satoshis ($1x10^{-6}$ Bitcoin), pero una vez creado, es indivisible, si un *UTXO* es más grande que el valor deseado de una transacción todavía debe consumirse en su totalidad y el cambio debe generarse en la transacción. Es decir, si se tiene un *UTXO* de 5 y se desea pagar 1 Bitcoin, su transacción debe consumir el total de 5 Bitcoin *UTXOs* y producir dos salidas: una que paga 1 Bitcoin a su destinatario deseado y otra que paga 4 Bitcoins de cambio a la billetera. Como resultado, la mayoría de las transacciones de Bitcoin generarán cambios. Por otro lado también es posible que sea necesario combinar varias unidades más pequeñas para alcanzar un monto deseado. Toda esta compleja combinación de *UTXOs* lo realiza la billetera del usuario automáticamente y es invisible para los usuarios.

Los *UTXOs* consumidos por una transacción se denominan entradas de transacción, y los *UTXOs* creados por esta se denominan salidas de transacción. De esta manera, los fragmentos de valor de Bitcoin avanzan de un propietario a otro en una cadena de transacciones que consumen y crean *UTXOs*. Las transacciones consumen *UTXOs* al desbloquearlo con la firma del propietario actual y crean *UTXOs* al bloquearlo en la dirección de Bitcoin del nuevo propietario.

La excepción a las transacciones con entradas y salidas es un tipo especial de transacción llamada transacción de *Coinbase*, que es la primera transacción en cada bloque. Una transacción *Coinbase* está formada por una entrada, que no está asociada con ninguna otra dirección. Esta entrada es la que genera las nuevas monedas dentro de la *blockchain*. Para dar salida a las mismas, existe una salida que apunta a la dirección Bitcoin del minero. En esta transacción, se reúne la recompensa del bloque junto con las comisiones cobradas por todas las transacciones incluidas en el bloque.

Esta transacción es colocada allí por el minero "ganador" y crea Bitcoins completamente nuevos pagados a ese minero como recompensa por la minería. Este tipo de transacciones, forman parte del sistema de puesta en circulación de nuevas monedas que jamás se han gastado.

Cada cliente de Bitcoin de nodo completo realiza un seguimiento de *UTXOs* en una base de datos contenida en la memoria, llamada Conjunto *UTXO*. Las nuevas transacciones consumen una o más de estas salidas del Conjunto *UTXO*.

Las salidas de transacción asocian una cantidad específica y una secuencia de comandos de bloqueo que define la condición que se debe cumplir para gastar esa

cantidad. En la mayoría de los casos, la secuencia de comandos de bloqueo bloqueará la salida a una dirección de Bitcoin específica, transfiriendo así la propiedad de esa cantidad al nuevo propietario. Las entradas de transacciones son punteros a *UTXOs*. Señalan un *UTXO* específico por referencia al *hash* de la transacción y el número de secuencia dentro de la transacción, en el que se registra el *UTXO* en la cadena de bloques.

Para gastar *UTXOs*, una entrada de transacción también incluye secuencias de comandos de desbloqueo que satisfacen las condiciones de gasto establecidas por *UTXO*. El script de desbloqueo suele ser una firma que demuestra la propiedad de la dirección de Bitcoin que se encuentra en la secuencia de comandos de bloqueo.

Para validar una nueva transacción cada nodo de Bitcoin ejecuta script de desbloqueo en cada entrada junto con el script de bloqueo de la UTXO correspondiente para ver si cumple la condición de gasto.

El script de desbloqueo es una secuencia de comandos que resuelve o satisface las condiciones colocadas en una salida por una secuencia de comandos de bloqueo y permite que se gaste la salida. Los scripts de desbloqueo son parte de cada entrada de transacción, y la mayoría de las veces contienen una firma digital producida por la billetera del usuario a partir de su clave privada.

Cada nodo de Bitcoin validará las transacciones ejecutando los scripts de bloqueo y desbloqueo juntos. Para cada entrada en la transacción, el software de validación recuperará el UTXO que intenta gastar y lo ejecutará junto al script de desbloqueo contenido en ella.

Si el resultado de ejecutar el script de bloqueo con los datos del script de desbloqueo es correcto, el script de desbloqueo ha logrado resolver las condiciones impuestas por el script de bloqueo y, por lo tanto, la entrada es una autorización válida para gastar el UTXO . Si queda algún resultado que no sea correcto después de la ejecución de la secuencia de comandos combinada, la entrada no es válida porque no ha podido satisfacer las condiciones de gasto establecidas en el UTXO.

Solo una transacción válida que satisfaga correctamente las condiciones del UTXO da como resultado que el UTXO se marque como gastado y se elimine del conjunto de UTXO disponibles.

3.2 Tarifas de transacción

La mayoría de las transacciones incluyen tarifas de transacción o *transaction fee*, que compensan a los mineros de Bitcoin por proteger la red. La mayoría de las billeteras calculan e incluyen las tarifas de transacción automáticamente.

Las tarifas de transacción sirven como incentivo para incluir una transacción en el siguiente bloque y también como un desincentivo contra las transacciones de

"spam" o cualquier tipo de abuso del sistema, al imponer un pequeño costo en cada transacción. Las tarifas de transacción son cobradas por el minero que registra la transacción en la cadena de bloques.

Como se explica en Antonopoulos [16, Capítulo 5], las tarifas de transacción se calculan en función del tamaño de la transacción en *kilobytes*, no del valor de la transacción en Bitcoin. En general, las tarifas de transacción se establecen en función de las fuerzas del mercado dentro de la red de Bitcoin. Los mineros priorizan las transacciones en función de muchos criterios diferentes, incluidas las tarifas, e incluso pueden procesar las transacciones de forma gratuita en determinadas circunstancias.

Las tarifas de transacción afectan la prioridad de procesamiento, lo que significa que es probable que una transacción con tarifas suficientes se incluya en el siguiente bloque minado, mientras que una transacción con una cantidad insuficiente o sin tarifas puede demorarse o no ser procesadas en absoluto. Las tarifas de transacción no son obligatorias, y las transacciones sin tarifas podrían procesarse eventualmente; sin embargo, la inclusión de tarifas de transacción fomenta el procesamiento prioritario.

La estructura de datos de las transacciones no tiene un campo para las tarifas. En cambio, las tarifas están implícitas como la diferencia entre la suma de las entradas y la suma de las salidas. Cualquier cantidad en exceso que quede después de que todas las salidas se hayan deducido de todas las entradas es la tarifa que cobran los mineros.

Este es un elemento un tanto confuso de las transacciones y un punto importante que se debe comprender, porque si se están construyendo transacciones debe asegurarse de no incluir inadvertidamente una tarifa muy grande al reducir el gasto de los insumos. Esto significa que debe dar cuenta de todas las entradas, si es necesario mediante la creación de un cambio, sino se terminará dando una gran propina a los mineros.

Por ejemplo, si consume un *UTXO* de 5 Bitcoins para realizar un pago de 1 Bitcoin, debe incluir una salida de cambio de 4 Bitcoins en su billetera, de lo contrario, el "remanente" de 4 Bitcoins se contabilizará como una tarifa de transacción y será recogido por el minero que ingrese su transacción en un bloque.

3.3 Grafo de transacciones

Como se comenta anteriormente las transacciones están compuestas por un conjunto de entradas y salidas. A la vez, dichas salidas pueden pertenecer al conjunto de salidas no consumidas o ser entradas de otras transacciones. A partir de esto, se puede definir un grafo dirigido G(T, L) donde T es el conjunto de transacciones en la

blockchain y L es el conjunto de asignaciones directas (relaciones de entrada salida en transacciones) entre estas transacciones. Cada asignación $l \in L$ tiene asociado a un número de Bitcoins en Satoshis. De forma inherente las transacciones tienen un orden total definido por la *blockchain*, y no pueden existir ciclos en G.

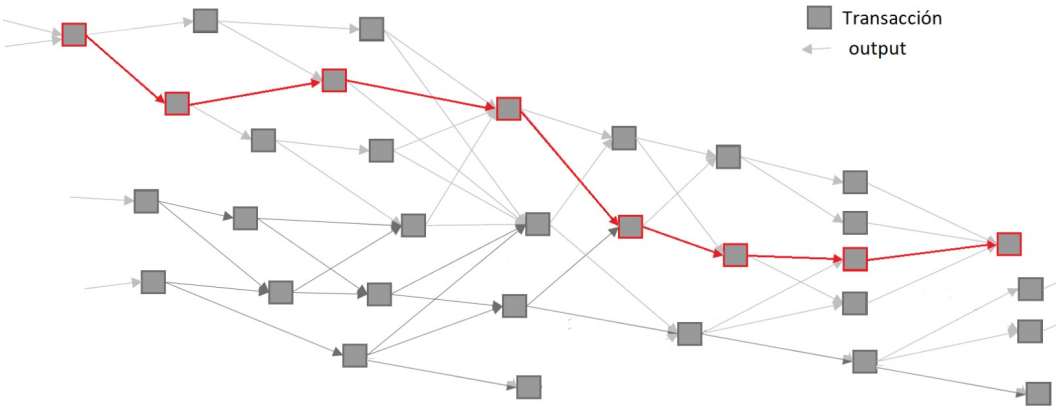

Fig. 2. Grafo rastreo de Bitcoins

En la figura 2 se observa una forma de poder rastrear un Bitcoin desde que fue recibida hacia atrás hasta que se llega a cuando fue creada.

Bitcoin tiene toda la información pública en la *blockchain*. A la hora del análisis de los datos allí guardados hay dos enfoques muy utilizados, uno es el de clusterización o *clustering* de direcciones y el otro es el de rastreo de direcciones. Ambos enfoques son abordados en el capítulo 5.

4 Anonimato en Bitcoin

Como bien explica Narayanan et al. [1, Capítulo 6], literalmente anónimo significa "sin nombre". Cuando se le trata de aplicar esta definición a Bitcoin se pueden observar dos interpretaciones: interactuar sin usar un nombre real ó interactuar sin usar ningún nombre. Estas interpretaciones llevan a diferentes conclusiones a si Bitcoin es anónimo o no. Las direcciones de Bitcoin son hashes criptográficos de claves públicas. No se necesita usar el nombre real para interactuar con el sistema, pero sí se usa el hash de la clave pública como identidad. En conclusión, desde el punto de vista de la primera interpretación Bitcoin es anónimo ya que no se usa el nombre real para usar el sistema. En cambio, tomando la segunda interpretación no lo es, ya que la dirección usada es una pseudo identidad, ésto es lo que en la ciencia de la computación se llama pseudo anonimato, como se había anticipado anteriormente que era el caso de Bitcoin. Ni siquiera la posibilidad que brinda Bitcoin de generar tantas direcciones como se quiera llega a proveer el anonimato tal cual se espera. Bitcoin es pseudo anónimo, pero tampoco el pseudo anonimato le permite a un usuario estar completamente seguro que en algún momento no se violará su privacidad. Todas las transacciones realizadas por un usuario se encuentran públicas en la *blockchain*, si en un momento dado alguien logra asociar una de las direcciones Bitcoins con la identidad real del usuario, toda su privacidad sería comprometida. Son variadas las formas en las que esta asociación entre la dirección Bitcoin y la persona real puede ser hecha, por ejemplo, al pagar un café con Bitcoins se está presente por lo que el cajero podría identificarnos, o si un *exchange* pide la dirección de residencia o detalles de tarjeta de crédito o incluso si interactuando en un foro uno deja su dirección Bitcoin para recibir donaciones, etc.

Como se puede observar en el artículo Shanmugam et al. [17], no es fácil llegar a dar con la persona física que realizó una transacción, pero al menos el estudio brinda una forma de corroborar la participación de una billetera en una transacción en el caso de obtener el dispositivo físico desde donde se realizó dicha transacción. Además nos deja claro que el análisis se puede volver más difícil en el caso que de por medio se utiliza Tor [18], o algún otro sistema que refuerce el anonimato. Por fortuna, se conocen casos como el arresto del creador de *Silk Road* [19], que marcan precedentes en pos de la captura de criminales que hacen uso de Bitcoin para realizar sus fechorías.

Aún así, una persona puede elegir una criptomoneda para intentar pasar desapercibido en el mundo de las transacciones financieras, se dice también que criptomonedas como Bitcoin son muy usadas para el lavado de dinero y crimen cibernético, si bien ésto puede ser así, también es verdad que a la hora de transformar esos Bitcoins en moneda corriente, ahí sí normalmente los *exchanges*

pedirán algún tipo de documentación para pagar con digamos dólares los Bitcoins que se le están entregando, por lo que en ese caso las transferencias dejarían en cierto modo de ser anónimas.

De todas formas para profundizar en el estudio de Bitcoin, hay dos mecanismos de atacar el pseudo anonimato brindado por éste. La primera es analizando el grafo de transacciones, ya que el mismo comprende el historial de todas las transacciones hechas, con sus direcciones de entradas y salidas más información extra. El segundo ataque es analizando la red *peer-to-peer* dado que un usuario cuando crea una transacción, la propaga a sus nodos vecinos, en el caso que un atacante controle suficientes máquinas en la red, podrá observar cual es el usuario de la red que primero difundió dicha transacción, pudiendo así saber la dirección IP de dicho usuario. Ésta dirección es bastante cercana a la identidad real del usuario, de hecho se puede desenmascarar a la persona detrás de una dirección IP. Para hacer más difícil dicha práctica, se puede utilizar sistemas como Tor [18], que utiliza dos nodos intermedios de la red para realizar la conexión entre el emisor y un receptor de un mensaje. Este trabajo se centra más en la primera forma de ataque, el estudio del grafo de transacciones. También se encuentran prácticas para fortalecer el anonimato y frustrar este tipo de ataque, una de las más notorias son los *mixers* explicados en el capítulo 2, un sistema el cual se podría ver como una caja negra, donde uno deposita una cierta cantidad de Bitcoins, la mezcla y obtiene la misma cantidad pero en una dirección diferente, se podría incluso usar varios de estos *mixers* en cadena para obtener un incremento en la seguridad del método, similar a lo que se comentó pasaba con Tor.

Los *mixers* actuales tienen dos falencias que en los artículos Liu et al. [20] y Wang, Li, and Yu [21] presentan y buscan evitar mediante su algoritmo. Éstas son que el *mixer* se pueda quedar con la plata depositada y que el *mixer* pueda decodificar la información del usuario que hace uso del mismo.

El método desarrollado en Liu et al. [20], se basa en el algoritmo *ring signature* (Firmado anillo) utilizado para revelar secretos anónimamente. El modelo general de dicho algoritmo se elabora de la siguiente manera: consta de un algoritmo de tiempo polinómico con un parámetro k de entrada y dos parámetros de salida, la clave pública *pki* y la privada *ski* como par de claves para cada miembro firmante. Una firma s para un mensaje m es creada luego de presentar el mensaje m, las claves públicas de todos los miembros del anillo *pk1, pk2, .., pkn* y la clave privada del propietario *ski*. Además hace uso de un algoritmo determinístico que luego de ingresar una firma s, un mensaje m y las claves públicas de los miembros devuelve verdadero si se verifica la firma y falso en caso contrario. Las propiedades de la firma en anillo son: la correctitud (si fue bien firmada y no falsificada, satisfará la verificación), el anonimato incondicional (por más que un atacante tenga todas las

claves privadas la probabilidad que elija el firmante correcto es *1/n*) y la inquebrantabilidad. El protocolo de *mixing* por ellos propuesto utiliza un servidor centralizado que va en contra de la idea de Bitcoin pero no destruye la estructura descentralizada de Bitcoin. Consta de tres fases, en la fase de pedido el cliente manda su clave pública y transacción que quiere mixear. El servidor devuelve *n* claves públicas con la del cliente incluida. En la fase de generación el cliente recibe todas las claves públicas de otros miembro y genera *n* direcciones Bitcoin para luego obtener sus propias Bitcoin luego del mixing. Luego firma todas las direcciones con el algoritmo polinómico ring signature y las envía una a una al servidor. El servidor genera la transacción con todas las entradas y salidas. En la fase de confirmación el cliente chequea que estén todas las entradas y salidas de la transacción, el cliente firma la transacción para que pueda ser enviada a la red de Bitcoin.

En cambio en el artículo Wang, Li, and Yu [21], el protocolo presentado tiene tres pasos. Primero se crea una nueva dirección *"escrow address"* la cual es controlada por todos los participantes. En el segundo paso, cada participante usa las llaves de encriptación de los demás (que no han participado aún en el proceso, en un orden predefinido) para generar capas de encriptación de direcciones de salida. En el paso final, se crea la transacción que tiene como entrada la dirección creada y como salida la lista ordenada de las salidas. La dirección es creada usando un algoritmo llamado *Distributed Key Generation (DKG)* [22]. Cada participante comparte información con los demás utilizando el protocolo *Feldmans Verifiable Secret Sharing (VSS)*. Para agregar su dirección de salida, cada participante saca una capa y agrega encriptada su dirección de salida deseada y pasa el resultado al siguiente participante. Finalmente para que se haga efectiva la transacción final, todos los participantes deben estar de acuerdo, sino la transacción es invalidada. La mejora en anonimidad sube de *1/n* siendo *n* la cantidad de participantes en la transacción a *1/k* siendo *k* la cantidad de todos los usuarios Bitcoin, esto es porque el resultado es un conjunto de direcciones que no tienen ninguna conexión con la dirección desde la que el usuario transfirió sus Bitcoins. La desvinculación entre las entradas y salidas es lograda con el protocolo *DKG* y la forma como se agregan de forma randómica las salidas por cada participante. Con el método como se utiliza la dirección creada, se evita que alguien pueda robar lo depositado, además como todos ya depositaron sus Bitcoin en dicha dirección, es más difícil que alguien pueda comportarse deshonestamente, ya que perdería sus Bitcoins también.

Un sistema que busca vencer algunos de los problemas impuestos en los *mixers*, que deja de ser centralizado y que brinda la seguridad de que en caso de que llegue a su completitud, los Bitcoins de salida llegarán a su destino es un protocolo que se llama CoinJoin [1]. La diferencia principal con los casos presentados anteriormente,

es que Coinjoin es descentralizado, no hay una entidad como un *mixer* sino que el mixado se da entre usuarios. En este sistema distintos participantes conforman una transacción Bitcoin donde cada uno firma su entrada y propone una salida. El orden de las entradas y salidas es randomizado de forma tal que un atacante no puede ver la relación entre unas y otras. Finalmente cada participante controla que su salida esté presente en la transacción y con la cantidad correcta para luego firmar la transacción. Los detalles importantes para desarrollar CoinJoin son encontrar los demás participantes que quieren mezclar sus Bitcoin, lo cual suele resolverse con un servidor que permite a usuarios unirse y agruparse. El intercambio de las direcciones entre los participantes se debe dar de tal forma que ninguno de los demás participantes puedan relacionar dicha dirección con el participante. Luego que todas las direcciones son agregadas, uno de los usuarios crea la transacción y la pasa a los demás usuarios que corroboran que sus datos estén incluídos y firman la transacción. Cuando todos lo hayan hecho, uno o varios usuarios emitirán la transacción para ser agregada a la *blockchain*. Si bien este mecanismo sufre de denegación de servicio, por ejemplo en el caso que un usuario decide no firmar la segunda fase del proceso, se le aplica un costo a todos los participantes para intentar evitar este tipo de comportamiento, formas de aplicarlo son sustrayendo una mínima suma de Bitcoins o pidiéndole un esfuerzo computacional. Una alternativa posible es usar mecanismos criptográficos para encontrar el participante deshonesto y eliminarlo del grupo.

En el artículo [23], se estudia una forma de mercado de nombre *JoinMarket* que está desarrollado sobre *CoinJoin* y busca resolver el problema inherente a *CoinJoin*, que quienes quieren hacer uso del sistema, tienen en ese momento que encontrar otro usuario intentando hacer la misma operación, para juntarse. *JoinMarket* soluciona ésto separando el mercado en dos grupos, los *makers* son lo que proponen las transacciones, por lo general usuarios con gran cantidad de BTC que utilizan el sistema para inversión y esperan a los *takers*, que son los usuarios que están intentando en ese momento realizar la transacción. Los usuarios de ambos grupos se comunican mediante un canal de chat centralizado, claramente ésto es una de sus principales falencias ya que va contra la descentralidad de Bitcoin. Este sistema mejora el provisto por *CoinJoin* y no sólo mejora el anonimato sino que también puede ser usado por sus usuarios como un método para invertir dinero dadas las bajas tasas aplicadas. Como contras al método se puede citar su centralización, su baja defensa contra ataques y la posibilidad de que cambien las leyes y se marque esta actividad como ilícita, comprometiendo todo el mercado.

Para incrementar el anonimato se han presentado dos nuevos protocolos, Zerocoin [24] y su sucesor Zerocash [25], ingeniosos criptográficamente y por el anonimato que prometen. Las ventajas que traen estos mecanismo tiene un costo y

éste es que no son compatibles con Bitcoin tal cual está hoy. Si bien Zerocoin se podría anexar a Bitcoin con una pequeña bifurcación, las dificultades prácticas hacen que sea inviable, principalmente el hecho que necesita un número N suficientemente grande que sea resultado de multiplicar dos números primos también muy grandes. Luego el sistema sería muy difícil de quebrar siempre y cuando los números primos no sean revelados. Por otra parte Zerocash ni siquiera es posible anexarlo a Bitcoin, por lo que podrían tratarse como *altcoins* o monedas alternativas a Bitcoin. Ambos prometen anonimato sin necesidad de *mixers*, nodos o intermediarios, tan sólo basados en los límites del poder criptográfico del adversario. Ambos protocolos se basan en *zero-knowledge proof*, un mecanismo para probar una declaración matemática sin revelar mas información que lleve a que la declaración sea verdadera. Por ejempo, supongamos que queremos probar que solucionamos un *hash puzzle*, o sea, sabemos x tal que el hash de ese x es menor que un valor dado. Por supuesto se podría revelar este x, pero esto haría que otras personas lo sepan, lo que permite *zero-knowledge proof* es revelar este hecho de una manera que las demás personas sean incapaces de saber nada acerca de x. Una de las principales diferencias entre ambos, es que Zerocash es más poderoso criptográficamente, por esto necesita para ponerse en práctica, más de un gigabyte en parámetros públicos. Además con Zerocash se pueden hacer todo tipo de transacciones mientras que con Zerocoin no se hacen las transacciones directamente sino que se utiliza la moneda base, digamos Bitcoin para hacer las transacciones y Zerocoin brinda un mecanismo para cambiar dichas Bicoins por unas nuevas que no están relacionadas de ninguna manera con las obtenidas.

En el capítulo 5 se analizan distintas técnicas pensadas para comprometer el anonimato. Además se comenta como se comportan algunos de estos mecanismos contra análisis del estilo del entintado en Bitcoin.

5 Rastreo de dinero en Bitcoin

Como se explica en el capítulo anterior, Bitcoin es pseudo anónimo, esto permite que alguien haciendo uso de sus Bitcoins pueda en cierto modo sentirse tranquilo que su interacción no puede ser asociada a él. Por otra parte, como toda la información de Bitcoin queda registrada en la *blockchain*, es posible realizar un análisis de esta información para identificar distintos patrones de uso, incluso es posible realizar un rastreo del historial de transacciones de un usuario Bitcoin.

Al hablar de rastreo de dinero en Bitcoin, tal vez la técnica más usada y que posee un gran potencial es la de clusterizar direcciones Bitcoin. Por clusterizar, se entiende la acción de juntar direcciones Bitcoin de forma tal que puedan ser asociadas a una misma entidad.

En este capítulo se presentan las principales técnicas encontradas para clusterizar entidades en Bitcoin. Se comienza presentando ejemplos en los que el rastreo de Bitcoins fue exitoso para comprender la importancia de este tipo de análisis. Luego se presentan los mecanismos que hicieron posible estos rastreos.

Uno de los rastreos de Bitcoins más notables se vio en los esfuerzos de las autoridades estadounidenses en perseguir al creador de Silk Road. El mismo contaba con 144 mil BTC al momento de su captura [26], llevaba a cabo un mercado en el cual se podían obtener servicios tan bizarros como obtención de drogas o asesinos a sueldo. En cambio, el mayor robo de la historia en cantidad de Bitcoins fue el reportado por Mt. Gox (segundo mayor cambio al momento) por un monto que asciende a setecientos cuarenta y cuatro mil Bitcoins [27], dicho cambio debió declararse en bancarrota.

En el artículo Meiklejohn et al. [28] se hace un riguroso análisis de la información en la *blockchain*, sus autores buscan entender mejor la trazabilidad del flujo de Bitcoins y a partir de este entendimiento, explorar la evolución de cómo Bitcoin ha sido usado a través del tiempo. Más importantemente, dicen, su objetivo no es sacarle el anonimato a todos los usuarios de Bitcoin (lo cual afirman que según el protocolo de diseño de bitcoin sería imposible) sino el de identificar ciertos modos de operar presentes en la implementación de la red de Bitcoin, que corroen la anonimidad de los usuarios que los utilizan. Su aproximación, continúan, está basado en la disponibilidad de la *blockchain*, la estructura de datos replicada que mantiene toda la actividad de Bitcoin, pasada y presente, principalmente en las relaciones entre las claves públicas que se asocian en cada transacción. Sin embargo, dado que cada una de estas claves no lleva información explícita acerca de sus propietarios, comentan que su análisis depende mucho en imposiciones adicionales sobre el grafo de transacciones a partir de datos obtenidos fuera de la *blockchain*.

En un principio hacen análisis realizando una clusterización de direcciones por

uso, centrándose en como se gastaban los Bitcoin, con que montos, etc. Encontraron un detalle interesante, observan que en 2012 aproximadamente, el 64% del total de las Bitcoin están almacenadas en direcciones que en su principio acumularon Bitcoins pero luego nunca más las usaron. Ésto puede deberse a que las personas que obtuvieron dichas Bitcoins perdieron su clave privada, perdieron su acceso a la billetera, o similar. Podría ser también que dichas Bitcoins fueron robadas y quien las hurtó decidió no usarlas en una forma de "lavarlas", hacer pasar el tiempo para que se olvide su origen.

Un pequeño análisis complementario de la situación actual de las Bitcoins perdidas, lo encontramos en la referencia [29], en ésta se puede ver que el problema continúa, estiman que aproximadamente cada dos meses se pierde un Bitcoin completo. También hacen un recuento de algunos casos notorios. Por ejemplo, un CEO Canadiense que manejaba un *Exchange* y falleció, causando la pérdida de aproximadamente 100 millones de dólares en Bitcoins, además de otras monedas.

Otro caso distinto es el que se da con las monedas asociadas a Satoshi (Satoshi Coins), las cuales están en su misma dirección desde 2009 y no se han movido, se dice que ascienden a 1 millón de Bitcoins, una suma descabellada que no se sabe si murió junto a Satoshi Nakamoto, quien al momento no se conoce su identidad o si las mismas están deliberadamente guardadas ahí con un fin que no se ha dado a conocer.

Se puede ver en la siguiente figura de noviembre del 2017, que no es un tema menor; la cantidad de Bitcoins fuera de circulación asciende a 2,56 millones con un precio al momento de 20 billones de dólares, una cifra preocupante. A esto se le pueden sumar el total de las monedas originales, aquellas que pertenecen a las Satoshi Coins con un total de 1,04 millones de Bitcoins en pérdida, no se quisiera ni vaticinar qué pasaría con el mercado si de un momento a otro las Satoshi Coins se ponen en movimiento, probablemente el suceso sería devastador.

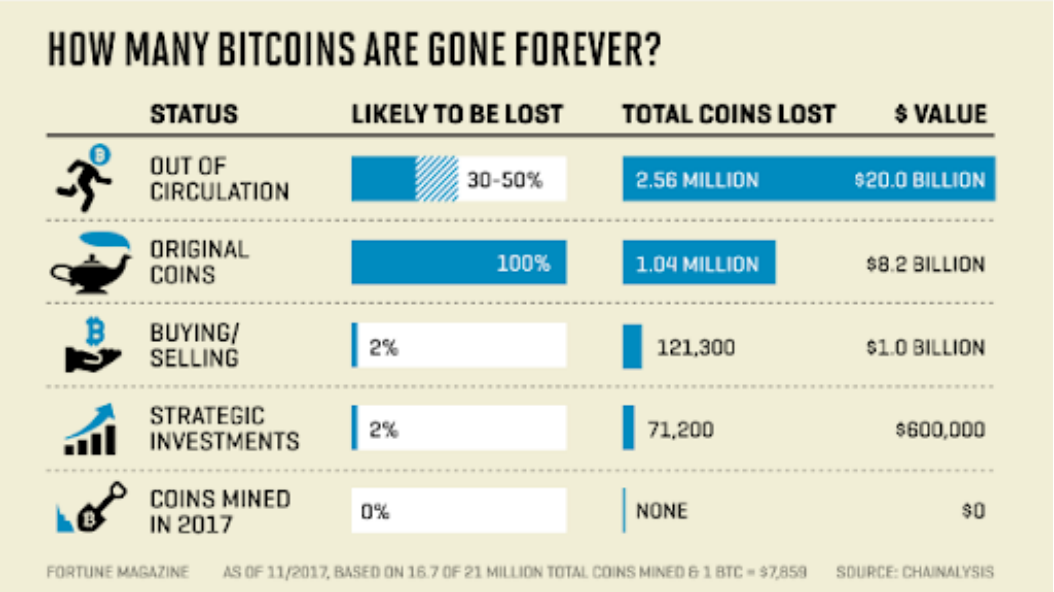

Fig. 3. Bitcoins Perdidas [30]

El estudio anterior, plantea la clusterización analizando sólo la participación o no de las direcciones en las transacciones, en cambio, para lograr asociar direcciones a entidades reales se necesita un método extra.

Para hacer la clusterización propiamente dicha de las direcciones tanto Meiklejohn et al. [28] como Ermilov, Panov, and Yanovich [31] y Maesa, Marino, and Ricci [32] hacen uso de heurísticas. Las heurísticas presentadas en estos trabajos son llamadas en [31] como *common spending* y *One Time Change (OTC)*. Cada artículo varía la forma en que la desarrolla y las distintas modificaciones que le hacen para ajustarla a lo que quieren analizar.

La heurística *common spending* hace uso de una particularidad de las transacciones de Bitcoin en las cuales todas las direcciones de entrada deben ser firmadas, ésto lleva a asumir que todas estas entradas pueden ser firmadas por la misma persona, ya que en caso contrario si cada clave pública estuviera controlada por diferentes entidades, una entidad tendría que revelar su clave privada a las demás. La heurística hace uso de esta característica para clusterizar todas estas direcciones de entrada en una misma entidad. Los efectos de la heurística son transitivos y se expanden mas allá de las entradas de una sola transacción. Si se

tiene una transacción con direcciones A y B como entradas, entonces si se tiene otra transacción con direcciones B y C como entradas, se puede concluir mediante esta heurística que se puede formar una entidad que engloba las direcciones A, B y C.

Por otra parte, la heurística *One Time Change (OTC)* hace uso de otra particularidad de las transacciones Bitcoin, la cual como se explica en el capítulo 3 la suma del monto que una persona tiene, por lo general difiere con lo que debe pagar, por lo que se crea una nueva dirección también asociada a la persona que sirve para depositar el cambio, resultado del pago. La heurística busca en el grafo de transacciones las direcciones que cumplen esta particularidad y lo que hace es asociarlas a la misma entidad que había hecho con las direcciones de entrada de la transacción. En [31] se refina la heurística pa evitar falsos positivos, por ejemplo, se pide que la transacción que se está analizando tenga exactamente dos salidas. La cantidad de entradas de la transacción no sea igual a dos para evitar los *mixers* compartidos. Que ambas salidas no sean de cambio. Que la dirección de cambio no haya sido usada anteriormente. La otra dirección que no es la de cambio, no puede haber sido de cambio anteriormente. Siguiendo algunos de estos consejos, evitan muchos falsos positivos que luego deberían ser depurados con pruebas.

Para complementar el estudio y poder relacionar los clusters obenidos con direcciones de usuarios reales, se emplean diferentes mecanismos, por ejemlo, el artículo Meiklejohn et al. [28] presenta dos mecanismos, por un lado realizar un estudio de direcciones interactuando con usuarios del mundo real y otro buscando en la web por direcciones publicadas por sus dueños. Con estos experimentos pudieron identificar ciertas direcciones Bitcoin y relacionarlas con entidades obtenidas clusterizando.

Con respecto a los resultados obtenidos con el estudio, por ejemplo, el artículo Meiklejohn et al. [28] muestra algunos datos interesantes de la aplicación de las heurísticas sobre la *blockchain* en ese momento (2013), dichos resultados se muestran en la siguiente figura 4, en la cual se pueden observar los clusters como círculos de diferentes colores y las aristas siendo las transacciones entre dos nodos siempre y cuando hayan hecho al menos 200 entre ellos.

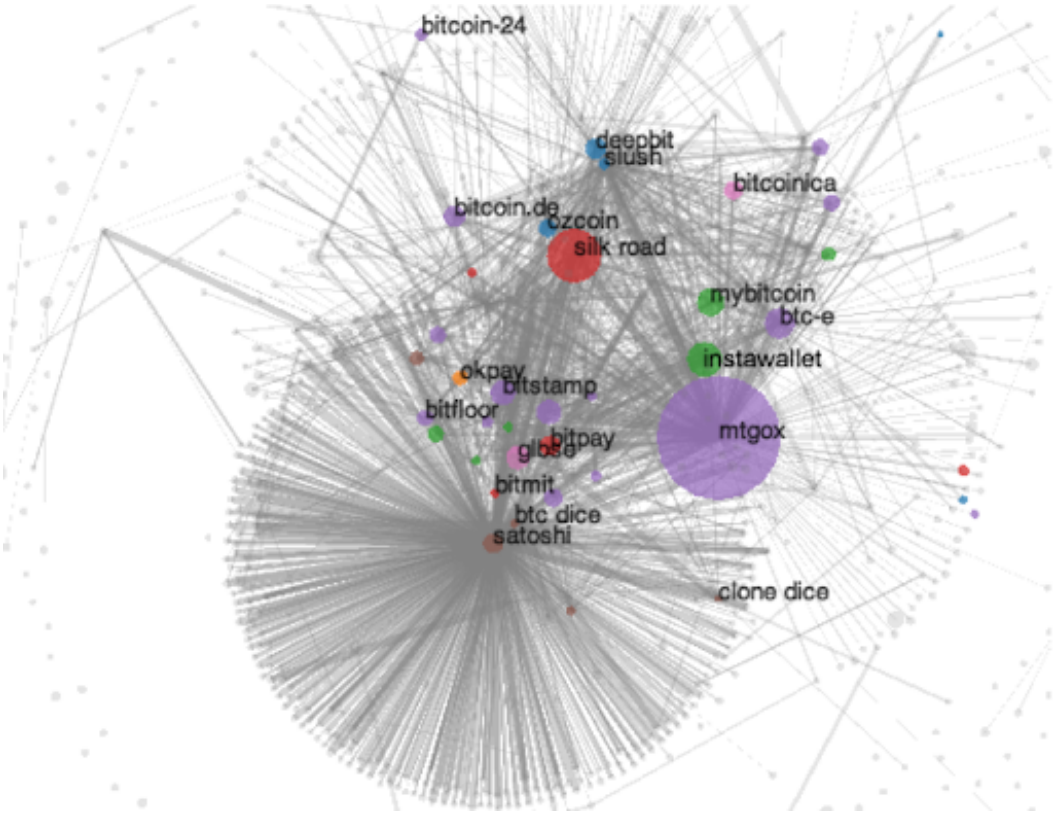

Fig. 4. Grafo de transacciones [28]

En el artículo Maesa, Marino, and Ricci [32], se realiza un estudio a partir de una cluterización de direcciones utilizando las heurísticas citadas anteriormente sobre la *blockchain* actualizada al 2015. Su enfoque se centra en un análisis estadístico de los clusters obtenidos. Algunos resultados se pueden observar en la siguiente figura 5 donde se detalla el tamaño de cada cluster y en los casos que fue posible encontrarle, el nombre real de la entidad.

THE 10 BIGGEST CLUSTERS

CLUSTER ID	IDENTITY	SIZE
66 482	Mt. Gox	10 216 380
2 899 325	LocalBitcoins.com	676 402
26 784 111	GoCoin.com	611 885
11 032 019	AgoraMarket	497 995
12 388 597	EvolutionMarket	420 632
2 477 299	N/A	392 589
2 547 597	SilkRoadMarketplace	372 753
10 072 646	SilkRoad2Market	349 874
1 175 285	BTC-e.com1	348 438
11 828 673	999Dice.com	301 990

Fig. 5. Clusters más grandes [32]

Ambas figuras demuestran que es posible representar la red de transacciones Bitcoin identificando entidades que participan, a tal punto de llegar a su nombre. Para esto último es necesario un estudio *off-chain*, o sea, por fuera de la *blockchain*. En este caso se logra haciendo un recorrido de la *web* para encontrar publicadas direcciones Bitcoin y asociarlas con cuentas *Twitter* por ejemplo, o encontrarlas publicadas en *blogs*, etc. Otra forma de encontrar relación entre direcciones Bitcoin con entidades, es por ejemplo interaccionando directamente con esa identidad, comprando y vendiendo productos.

Luego de este análisis nos podríamos preguntar las posibilidades de evitar estas heurísticas. La heurística *common spending* no puede ser evitada facilmente en futuras implementaciones según [33], al menos sin comprometer la mecánica actual de funcionamiento de Bitcoin. La heurística OTC es incluso más complicada de evitar ya que la solución de no utilizarla haría que el cambio vaya a la dirección de entrada, haciendo más fácil el estudio. Tal vez sería complicado la aplicación de esta heurística si Bitcoin permitiera transacciones *multi-input multi-output (MIMO)*, o sea con muchas entradas y muchas salidas, como también vaticinan en [33]

Otra forma de clusterización que se realiza en Bitcoin es el entintado de

direcciones. El entintado de direcciones Bitcoin se realiza con el objetivo de marcar tales direcciones para asociarles una particularidad, como por ejemplo, ver si dichas direcciones fueron partícipes de un ilícito.

En el artículo Anderson et al. [4] se realiza un estudio para probar si se pueden rastrear las direcciones ilícitas de forma tal de poder saber en cierto momento de la historia, si las bitcoins estudiadas podrían ser consideradas manchadas por ser el fruto de un ilícito (narcotráfico, robo, extorsión, etc) o si en algún momento fue partícipe de una transacción con direcciones que sí lo eran. Para llegar a ésto, se hace un estudio de la historia, no sólo de Bitcoin sino también de cómo las distintas monedas se han comportado en el pasado.

Como se presenta en Anderson et al. [4, Capítulo 2] "Nemo dat quod non habet", lo que traducido sería "nadie puede dar lo que no es suyo" marca un ejemplo claro de ilícito, si Alicia le roba el caballo a Bob y se lo vende a Charlie, Charlie no es el dueño del caballo, si más tarde Bob lo ve con el caballo, puede pedir que se lo devuelvan. Una posible solución a ésto, se aplicaba en la era medieval, donde Alice después de robar el caballo a Bob, lo vendía en el mercado del pueblo, si la venta era fructífera en el correr del día, entonces Charlie efectivamente era el dueño del caballo. Si bien Bob podía ir legalmente contra Alice, Charlie seguía siendo el dueño del caballo, para haberlo evitado, Bob tendría que haber asistido al mercado del pueblo para probar que el caballo era suyo. A los efectos del rastreo en Bitcoin, los Bitcoins robados por Alice y vendidos a Charlie, seguirán siendo de Charlie, aunque los mismos estarán manchados por ser el resultado de un robo, se corresponde a que Bob no asistió al mercado del pueblo.

Uno de los mecanismos actuales para intentar exponer actividades ilícitas y que favorece el rastreo y entintado de direcciones Bitcoin son las listas negras. Listas en las que se agrupan un conjunto de direcciones que formaron parte en algún momento de actividades ilícitas. Quizás una de las listas negras más emblemáticas es la publicada en el foro de Bitcoin [34], aunque la misma se discontinuó en 2013 debido a la falta de disponibilidad de quien la había realizado.

Una forma de análisis del grafo de transacciones que cobra relevancia ya que parte de información externa a la *blockchain* como las listas negras, es el entintado de direcciones presentado en Anderson et al. [4]. El entintado es una forma de rastreo de Bitcoins en la que una marca se le pone a un Bitcoin de forma tal que ésta quede identificada con cierta particularidad, en este caso, un ilícito que puede ser lavado de activos, narcotráfico, etc. Luego, se utiliza algún algoritmo para que esta marca se propague a otras Bitcoin que participaron en transacciones con una marcada. Los principales algoritmos utilizados para el entintado de direcciones en Bitcoin son llamados Poison, Haircut y FIFO. Todos éstos tienen sus particularidades y difieren entre sí en algunos detalles, su diferencia más notoria se da en la forma

y cantidad de direcciones Bitcoin que terminan “pintando”, siendo Poison el que más direcciones termina “pintando” o envenenando como lo dice su nombre en inglés, seguido por Haircut y siendo FIFO el algoritmo que concentra mejor el entintado. A continuación se provee una breve explicación de cada uno. Para ayudar la comprensión se utilizan imágenes para cada algoritmo. Cada imagen corresponde a una transacción. En barras verticales se encuentran las entradas (inputs) y salidas (outputs) de la transacción. Las barras horizontales punteadas delimitan cada entrada o salida, las cuales son identificadas con una letra A, B, C, D, E, J. Con distintos colores se presentan los tipos de ilícitos, por ejemplo, en rojo podría ser robo, azul narcotráfico, etc.

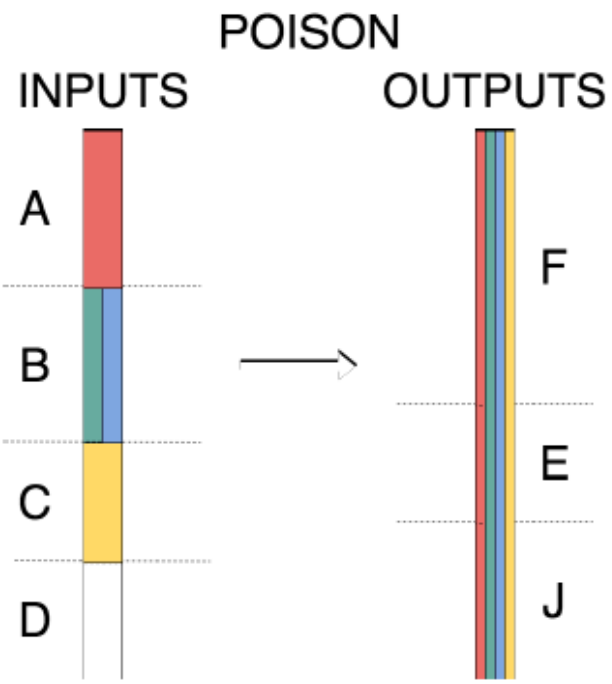

Fig. 6. Algoritmo Poison [4]

Poison es un algoritmo de entintado en el cual la salida total será marcada por todas las marcas que contenga la entrada. Como muestra la figura 6, las cuatro marcas que tienen las entradas, pasan a marcar toda la salida.

Dada una transacción en Bitcoin, la cual tiene *n* entradas y *m* salidas, Poison hace que todas las salidas sean agregadas a la lista negra siempre y cuando al menos una de las entradas esté en dicha lista. O sea, una entrada manchada logra entintar todas las salidas de la transacción a la que pertenecen. Como se puede observar y como el nombre lo dice, la forma que Poison envenena la red es bastante masiva. La figura 6 muestra las entradas A, B y C manchadas con los colores rojo, verde, azul y amarillo, la aplicación del algoritmo resulta en todas las salidas (incluso J que estaba a la altura de D que estaba "limpia") manchadas con todos los colores.

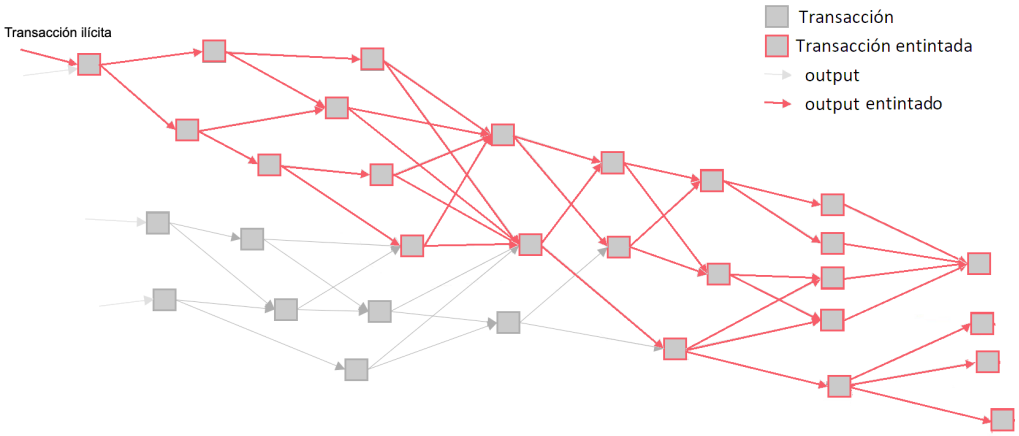

Fig. 7. Ejemplo de Grafo resultado de aplicar Poison

En la figura 7 se presenta el entintado utilizando el algoritmo Poison a partir de una dirección manchada o entintada con un único color (ilícito). Se puede observar como el número de direcciones entintadas crece significativamente a medida que se procesan las transacciones como así también crece el número de Bitcoins entintados. Por ejemplo, si se tiene una transacción cuyas entradas equivalen a cinco Bitcoins y ésta posee una entrada de 0,5 Bitcoins entintada, se obtiene como resultado los 5 Bitcoins de salida entintados.

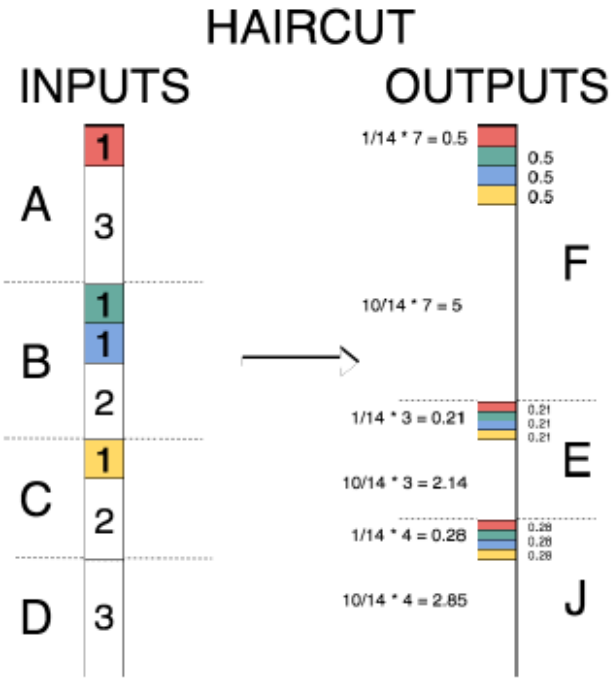

Fig. 8. Algoritmo Haircut [4]

Haircut es un algoritmo de entintado que propaga a la salida un porcentaje en el total de cada entrada. Como se observa en la figura 8, se tienen cuatro marcas, con un peso de 1/14, como la salida F tiene un tamaño de 7, cada marca en dicha salida tiene un peso de 1/14*7.

Haircut nace como una aproximación menos drástica al entintado en Bitcoin que Poison, lo que se propone con este algoritmo es que la salida esté manchada con un porcentaje del total entintado en las entradas. Por ejemplo, si se tiene una transacción con tres entradas y una salida, de las cuales una de dichas entradas está entintada por ser denunciada como robada, entonces la salida será entintada en un tercio como robada.

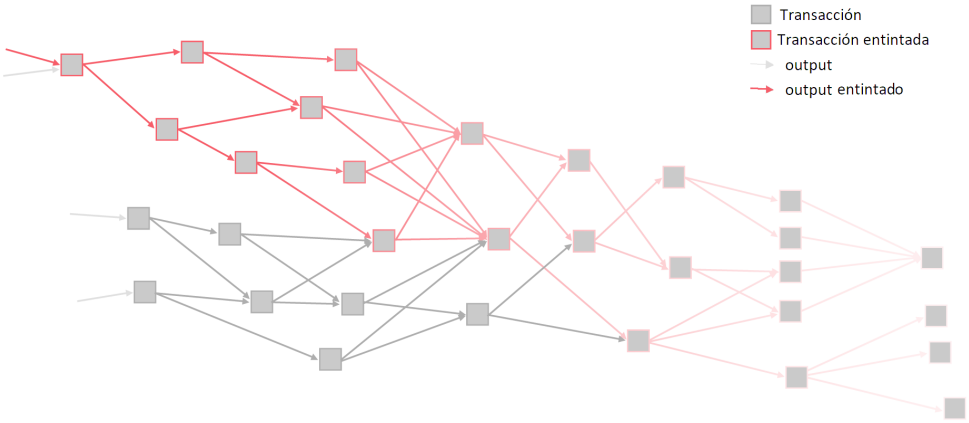

Fig. 9. Ejemplo de Grafo resultado de aplicar Haircut

Como se observa en la figura 9 el entintado del grafo de transacciones resultado de aplicar el algoritmo Haircut, se va difuminando a medida que se van procesando transacciones. Esto se corresponde con que el porcentaje asociado al entintado es cada vez menor, a medida que las entradas manchadas se juntan con entradas limpias. En la figura se parte de una transacción con una entrada manchada y una limpia y el color rojo se va transformando en gris como una forma de representar la disminución del procentaje asociado al entintado.

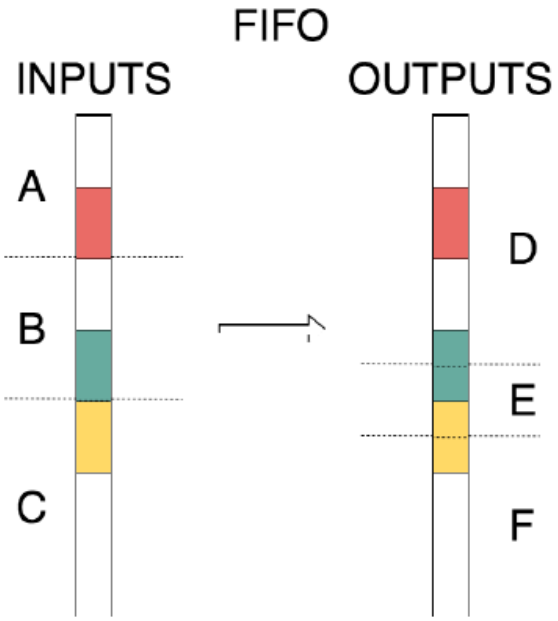

Fig. 10. Algoritmo FIFO [4]

FIFO es un algoritmo de entintado donde la marca se propaga en componentes individuales, en Satoshis individuales de cada salida. De hecho, como se observa en la figura 10, si una parte de la entrada está marcada, la misma parte en la salida lo estará. En la figura se puede ver que la entrada A está entintada en parte con el color rojo por lo que la salida D debe ser entintada no solamente con el mismo monto que en A sino que también en la posición correspondiente. Es decir suponiendo que la entrada A tiene un monto de 1 Bitcoin asociado, de los cuales los últimos 0.5 están marcados con rojo y que la salida D es de 2 Bitcoins, el procesamiento del entintado de color rojo tendrá como resultado los primeros 0.5 Bitcoins de D limpios peros los 0.5 siguientes marcados con rojo y el último Bitcoin limpio.

Luego de haber analizado cómo se comportan tanto Poison como Haircut según el artículo [4], es que decidieron recurrir a un antiguo caso para presentar un enfoque diferente. Dicho caso que tuvo lugar en 1816 en Inglaterra, marca el precedente del rastreo de fondos robados. En éste se trata el tema de las obligaciones que tenía un banco que quebró con su cliente dependiendo de las sumas que se habían hecho y lo que se había extraído. En el caso Clayton como fue llamado, se debió abordar

el problema de la mezcla de fondos buenos y malos, para esto se estableciío una simple regla de *First-In First-Out (FIFO)*: Las extracciones de una cuenta serán comparadas con los depósitos que se le hicieron a la misma. La regla FIFO hace que el rastreo en Bitcoin sea determinístico y en principio sencillo. Aplicando esta regla se pueden rastrear en la blockchain todas y cada una de las 744,000 BTC robadas a Mt. Gox, ésto es lo que la hace tan interesante de investigar y desarrollar.

Los resultados que obtuvieron luego de una primera prueba del entintado FIFO a partir de robos conocidos, mostraron que concentra mucho más lo entintado que el algoritmo *Haircut*, por ejemplo para el caso de robo de 46,653 Bitcoin de *Linode* si se usara *Haircut* se verían entintadas 16,855,619 (93% del total) direcciones mientras que FIFO entinta sólo 245,120 (1,93% del total)

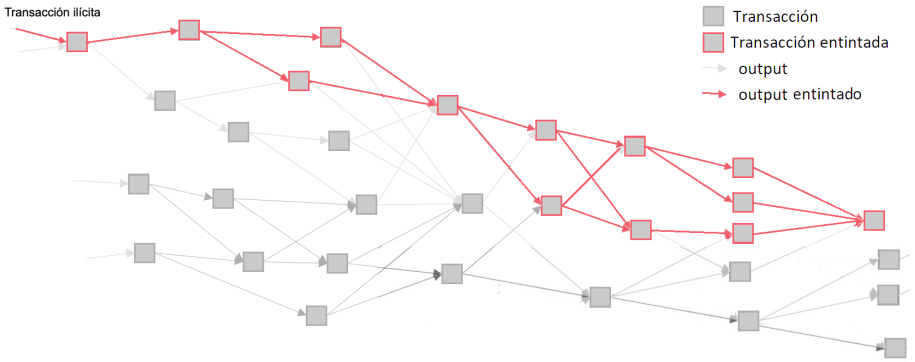

Fig. 11. Ejemplo de Grafo resultado de aplicar FIFO

En la figura 11 se puede observar cómo se transmite el entintado aplicando el algoritmo FIFO. La cantidad de nodos pintados por este algoritmo es significativamente menor que los otros dos algoritmos. También se puede ver que este posee la característica que a medida que se procesan transacciones el número

de Bitcoins entintados permanece constante.

Con respecto a los mecanismos existentes hasta el momento para identificar entintado, en [4] realizan un estudio empírico comprando Bitcoins sospechosos y recibiendo distintos niveles de alerta de algunos reportes de sistemas anti-lavado de dinero estandard. En otros casos el chequeo daba negativo, cuando se sabía que la dirección era participante de *Cryptolocker* (el primer ransomware) [35]. Además de ésto, las principales firmas anti-crimen se focalizan principalmente cuando grandes sumas de dinero están de por medio, o sea, no ayudan en nada a los usuarios normales, por lo que éstos son vulnerables a recibir Bitcoins ilícitas, lo que les llevó a publicar la *blockchain* de direcciones ilícitas a la cual le llamaron "*Taintchain*" [5] así como el programa en el cual se basa este trabajo [36].

6 Implementación

En este capítulo se detalla el análisis realizado de la implementación del algoritmo de entintado FIFO llevado a cabo en el artículo [4].

6.1 Indexado de la blockchain

En el artículo Anderson et al. [5] presentado anteriormente, donde utilizan el término *"The taint chain"* para denotar la *blockchain* resultado de correr su algoritmo de entintado, se presenta *Rusty Block Parser*, un parser de la *blockchain* de Bitcoin escrito en lenguaje *Rust*, que permite la extracción de varios tipos de datos (bloques, transacciones, *scripts, public keys/hashes*, balances, etc) a partir de un copia local de la *blockchain*, normalmente descargada de *Bitcoin Core*. El parseo de la *blockchain* se realiza en dos etapas; primero se itera sobre todos los archivos de tipo blk*.dat realizando un proceso de Indexado analizando los encabezados de cada bloque. Este proceso no evalúa el bloque completo sino que sólo calcula los hashes del bloque para determinar la cadena principal. Luego, la cadena principal se guarda como un archivo de tipo JSON llamado *ChainStorage*. Una vez que se determina la cadena principal, el analizador inicia una exploración "*Full Data*". Éste comienza cargando el archivo *ChainStorage* y el *Parser* delega cada archivo blk.dat a un *worker* en el grupo de subprocesos. Cada uno de estos *workers* procesa y valida todos los tipos de datos (*hash* del bloque *txid, script, public key/hashes*, etc). Una vez procesados los datos, se envían de vuelta al *parser* y se pasan a una *callback*. Este *parser* soporta múltiples hilos para optimizar recursos pero garantiza que siempre los bloques son procesados en orden. Las *callbacks* se ejecutan al iniciar el proceso, al finalizar y cada vez que se procesa un bloque. El *parser* implementa dos *callbacks*, una llamada "*simple-stats*" que imprime estadísticas de *blockchain* como recuento de bloques, recuento de transacciones, transacciones promedio por bloque, transacciones más grandes, tipos de transacciones. La otra de nombre "*csvdump*" que permite almacenar la información de los bloques, transacciones, entradas y salidas en una base datos en archivos csv.

6.2 Entintado de la blockchain

El algoritmo FIFO que se analiza e implementa, se basa en una implementación de una *callback* que se debe registrar en código del parser de la *blockchain*. Este procedimiento espera como parámetro de entrada un archivo con la información de las transacciones asociadas a eventos ilícitos en el formato *tx, [taint type, taint value]* donde *tx* corresponde al hash identificador de la transacción y *[taint type, taint value]* corresponde a un *array* donde se especifica la razón del entintado y el valor entintado.

A partir de este documento se crea un índice con las diferentes razones de entintado donde 0 corresponde a *Clean* (limpio) y los demás se asignan según el orden en el csv. Estos índices se utilizan para identificar el entintado en el archivo resultado y se devuelve dicha asignación en un archivo llamado *taint_mapper.csv*, explicado en sección 6.5.

La función principal del código es *on_block* y ésta se ejecuta cada vez que se procesa un bloque de la *blockchain*. El resultado se almacena en un diccionario llamado "*address_mapping*" cuya clave está compuesta por un id de transacción y un índice que corresponde a la ubicación del *output* o salida de la transacción. El valor del diccionario es un *AddressInfo* el cual está compuesto por un *timestamp* del bloque, un balance donde se almacena en satoshis el valor del *output* y un *tainted balance* que corresponde a un *array* que contiene en orden el entintado de dicha salida.

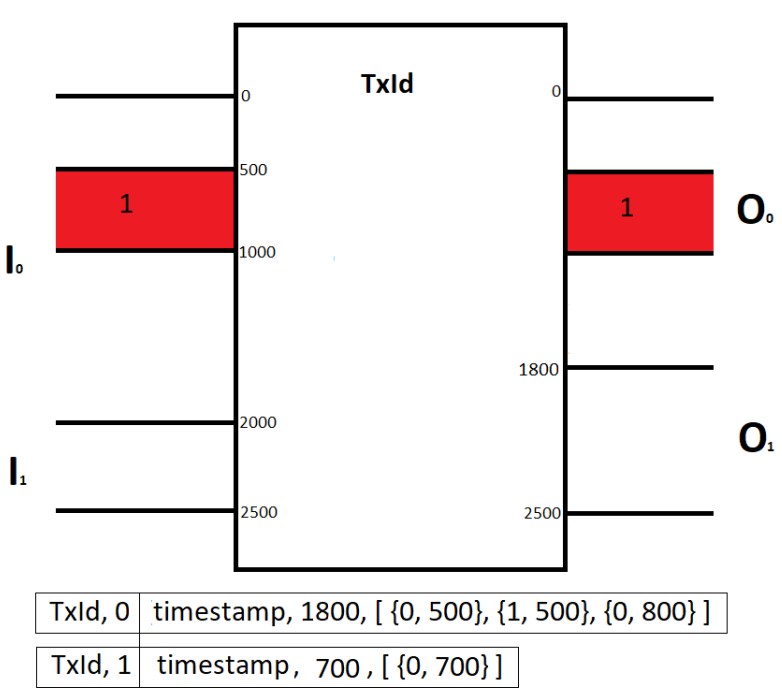

Fig. 12. Entintado en Transacción

En *tainted balance* se almacenan objetos de tipo *TaintPart* conformados por *[taint type, taint value]* donde no sólo tiene importancia el valor y la razón del entintado sino el orden en el cual se encuentran dentro del array. En la Figura 12 se

puede ver un ejemplo de una transacción donde el primer output posee 500 satoshis limpios, 500 satoshis entintados con tinta de tipo 1 y luego 800 satoshis limpios y en el segundo output todos los satoshis están limplios.

Una vez finalizado el procesamiento la información de este *hash* se devuelve en el archivo *address_info.csv*, para más información consultar sección 6.5. Cuando se procesa un bloque en una primera instancia se identifica la transacción *Coinbase* la cual contiene la recompensa y el *fee* o tasa del minero, lo que en una primera instancia es ignorada en el entintado. Para cada transacción primero se analiza si se encuentra entre las transacciones ingresadas en la lista y en caso positivo se recorren sus salidas agregando al "*address_mapping*" el entintado correspondiente.

Luego, como se vió en el capítulo 3 para obtener el valor del *fee* de los mineros se realiza la resta entre la suma de las entradas y la suma de las salidas. Como se considera que se debe entintar también el *fee* de una transacción ilícita éste es utilizado en el entintado posterior.

En una segunda instancia se procede a realizar el entintado a partir del estado ya almacenado en el "*address_mapping*". Se recorren todos los *inputs* de la transacción obteniendo el *tainted balance* y si este contiene algún entintado se agrega el estado que luego se utilizarán para procesar las salidas.

Antes de realizar el entintado correspondiente debido a que el *fee* del minero no aparece definido en las salidas se agrega un output con la dirección del minero con el valor correspondiente y un *offset* calculado correspondiente dentro de las salidas de la *Coinbase* para el caso en que corresponda pintar este *output* agregar la *TaintPart* en la posición correspondiente en el *output* de la *Coinbase*.

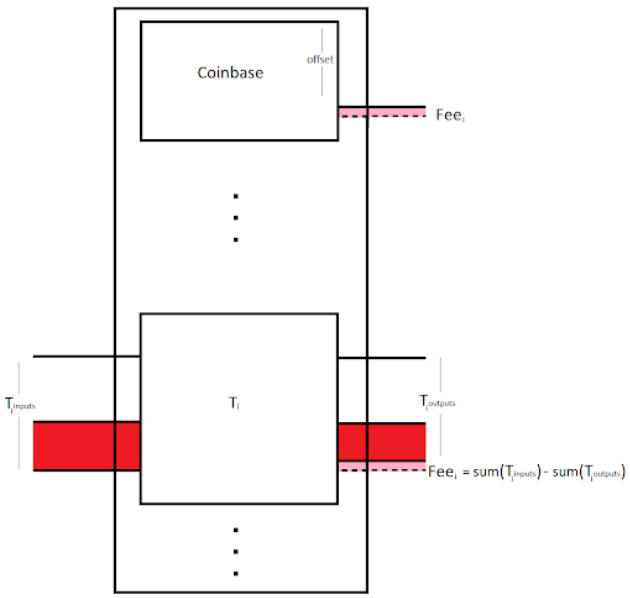

Fig. 13. Entintado en Transacción y Coinbase

Como se puede ver en la figura 13 existe un *fee* del minero implícito en cada transacción como la diferencia entre la suma de las entradas y la suma de las salidas. Caso corresponda entintar dicho *fee* es necesario calcular el *offset* correspondiente dentro de la salida de la *Coinbase* que corresponde a dicha transacción, para así realizar el entintado correctamente. Este offset es utilizado para entintar la sección de la *Coinbase* que corresponde al *fee* de dicha transacción.

Por último se mapea el entintado de las entradas de la transacción a sus salidas respetando el orden correspondiente y se agrega el entintado correspondiente al "*address_mapping*". Este poceso se repite para todas las transacciones del bloque porcesado en la *callback*

6.3 Trabajo sobre Código

Se trabaja sobre la implementación descargada desde el repositorio en *GitHub* [36] y se realiza un extenso análisis para comprender mejor cómo funciona y cuáles fueron las decisiones tomadas a la hora de desarrollar el algoritmo FIFO comentado en 2; luego de realizado ésto y dado que el desarrollo no compilaba en el entorno en el que se trabajaba, se toma la decisión de modificar el código a medida que se compilaba para lograr una versión ejecutable de la aplicación. Las principales falencias encontradas se centraron en el manejador de paquetes, llamado cargo en *Rust*, muchas de las dependencias que se encontraban en éste tenían versiones desactualizadas por lo que tuvo que hacerse el trabajo de ir buscando las que más se adecuaban para interactuar entre ellas y conseguir aquellas que no estaban presentes. Otras diferencias encontradas eran *Crates* faltantes en las definiciones. Después de varios de éstos cambios se logra obtener una versión ejecutable del código, cuya última versión se encuentra subida a *GitHub* [37].

Luego de realizada la primer ejecución ya se detectaron varios comportamientos que no se entendieron, principalmente porque la comprensión del algoritmo de forma teórica no permitía reconocer algunas de las salidas obtenidas, por ésto luego de realizar un exhaustivo análisis y comprensión de los algoritmos, se logra identificar las causas y finalmente se llega a una primer versión de la implementación donde se podían observar salidas acordes a lo esperado.

6.4 Análisis de las salidas

Luego de obtener una versión ejecutable se pasó a analizar tanto los archivos csv de entrada que se necesitaban como los archivos csv de salida. Dado que dichos archivos de salida no tenían especificación de los datos que cada columna del archivo representaba, se tuvo que estudiarlos para darle un sentido.

Llegar a reconocer cada una de las columnas que se devuelven en los archivos comprendió una mezcla de razonamiento de lo que debía devolver el algoritmo e intuición de lo que podía especificar cada columna. Con este procedimiento, se identificaron columnas que representaban direcciones de Bitcoin, marcas de tiempo del estilo *timestamp*, identificadores de transacciones, fechas, entre otros. Algo que fue de mucha ayuda para hacer este análisis fue la página *blockchain.info* [38], en ésta se guarda la información de todas las transacciones de Bitcoin, donde para cada transacción se pueden observar todas sus entradas, salidas, totales y alguna información general.

Fig. 14. Transacción en blockchain.info parte izquierda imágen [39]

Fig. 15. Transacción en blockchain.info parte derecha imágen [39]

En las figuras 14 y 15 se presenta una transacción elegida de *Blockchain.info* que posee pocas entradas y pocas salidas para facilitar la comprensión. En la figura 14 se puede observar el *id* de la transacción siendo éste el texto en celeste más largo, mientras que las líneas representadas en celeste debajo de ésta son las direcciones de entrada, también en esta figura se observan algunos datos resumidos de la transacción en la parte de abajo. Por otra parte en la figura 15 se observan en celeste las direcciones de salida con sus respectivos montos y el total de salidas de la transacción en la caja verde. También en la parte de abajo se pueden observar detalles resumidos de entradas y salidas.

6.5 Experimentación

En primera instancia se apuntaba a hacer el análisis de un mes particular de la *blockchain* pero ésto tuvo que ser cambiado ya que el *parser* sobre el que ejecutaba la *callback* "*taintchain*" debía sí o sí comenzar por el bloque génesis, ya que durante el proceso mantiene en memoria el histórico de UTXOs disponibles. A continuación se trabaja experimentando con la *blockchain* completa, acá se pudo observar que la duración era mayor a 24 horas lo que motivó a comenzar a hacer pruebas con tan sólo algunos bloques iniciales de la *blockchain*.

Luego de comenzar a probar con bloques iniciales, se pudo observar que éstos mayormente están compuestos por transacciones del estilo *CoinBase* en los que aplicar el entintado no es tan clarificante. Por ésto, se debieron agregar más bloques para poder analizar el comportamiento y verificar la correctitud de la implementación. A medida que se iban agregando cada vez más bloques, la performance del proceso decae, esto se atribuye a que se mantienen en memoria las UTXOs como se comentó previamente, además de la información de entintado y se procesan más datos.

Las pruebas realizadas sobre la *blockchain* completa se hicieron con una computadora con las siguientes características:

Equipo 1
CPU: Intel Core i7-7820 (2,9GHz-3,9GHz)
RAM: 16GB DDR4
Video: NVIDIA GeForce GTX 960

Equipo 2
CPU: Intel Core i5-7200 (2,50GHz 2,70GHz)
RAM: 16GB DDR4
Video: NVIDIA GeForce GTX 960

Equipo 3
CPU: Intel Pentium CPU 997 1,60GHz
RAM: 4GB
Video: Intel Chipset

Equipo 4
CPU: Intel Core i7-8550U (1,8GHz 1,9GHz)
RAM: 16GB DDR4
Video: NVIDIA GeForce MX150

Éstas últimas tienen la particularidad de que no soportan las ejecuciones con la *blockchain* completa. Esto se debe a que se mantiene mucha información en memoria y la capacidad de las máquinas no permite almacenarla en la RAM ni realizar el *swap* a disco. Finalmente por disponibilidad y buen rendimiento se utiliza el equipo 2. La salida de una ejecución se presenta en el listado 1:

```
[16:36:43] INFO - FIFO [on_block]: Progress: block
   406000, 36104807} UTXOs, 5832210 fragments on 680375
    accounts, collisions: 3159, tx left: 0
[17:13:55] INFO - FIFO [on_block]: Progress: block
   407000, 36511768} UTXOs, 5878404 fragments on 686567
    accounts, collisions: 3159, tx left: 0
[17:40:27] INFO - FIFO [on_block]: Progress: block
   408000, 36890056} UTXOs, 5923590 fragments on 692609
    accounts, collisions: 3159, tx left: 0
[17:52:38] INFO - dispatch: All threads finished}.
[17:52:38] INFO - dispatch: Done. Processed 408607
   blocks in 1356.65 minutes}. (avg: 5 blocks/sec)
[18:02:13] INFO - FIFO [export_utxo_set_to_csv]:
   Exporting 37108496} UTXOs\ to CSV...
[18:22:40] INFO - FIFO [export_utxo_set_to_csv]:
   Exported 37108496} UTXOs to\ CSV.
[18:22:41] INFO - FIFO [on_complete]: Done}. Processed
    all 408608 blocks
[18:22:41] INFO - dispatch: Saving block headers as
   chain.json} ...
[18:24:08] INFO - main}: Fin.
```

Listado 1: ejecución de código

De ésta información se observa que la ejecución contiene 408608 bloques y 37108496 UTXOs. Con respecto al tiempo que tomó la ejecución fueron aproximadamente 23 horas, teniendo en consideración que se trabajó sobre la *blockchain* descargada por Torrent [40] en julio 2018, actualizada hasta la fecha de abril 2016 y que actualmente la *blockchain* contiene cerca de 583527 bloques.

El siguiente paso fue darle sentido a los archivos de salida con los que el proceso terminaba. Los mismos eran archivos csv y también se tuvieron que tomar recaudos cuando se corría con la *blockchain* entera ya que si se partía de direcciones que pintaran mucho, algunos de estos archivos tenían tantas líneas que pesaban varios GB lo que dificultaba su procesamiento.

Primeramente, para que el proceso devuelva alguna salida interesante desde el punto de vista del entintado, primero hay que brindarle un archivo con las direcciones entintadas en un principio, dicho archivo se llama *example_dataset.csv* y contiene tres columnas, las cuales representan la dirección entintada, por qué tipo de ilícito está entintada y por cuánto en orden de izquierda a derecha.

Por otra parte, los nombres de los archivos de salida que se pasarán a explicar sus columnas son *address_info.csv*, *taint_overlap.csv*, *taint_utxo.csv* y *taint_mapper.csv*

Taint_mapper.csv

El archivo reúne los distintos tipos de ilícito que se van a tener en cuenta en la corrida del proceso y con qué índice se lo asociará. Para observar un ejemplo gráfico ir a la figura 16 del anexo donde se puede ver el mapeo de 0 con *Clean* o limpio, 1 con *Allinvain Theft* y 2 con *Linode Hacks*.

Address_info.csv

Este archivo contiene todas las direcciones entintadas resultado de correr el proceso. La primera columna reporta la transacción y el *id* como un byte, la columna 2 es la fecha mientras que la última el balance que en caso de haber más columnas es la suma de éstas. En el anexo, en la figura 17 se puede observar una salida del mismo. Se puede ver una transacción que presenta los primeros 3 Bitcoins (expresados en satoshis) limpios, luego 12 bitcoins marcados con ”2” y los ultimos 5 bitcoins limpios.

Taint_overlap.csv

En este archivo recupera solapamientos de entintado, si una dirección estaba entintada por un ilícito y luego se encuentra entintada por otro, aparece en éste archivo. La columna 1 identifica por qué estaba pintado, la columna 2 cuál es el

nuevo entintado, la columna 3 es la cantidad en Satoshis que se está entintando, la columna 4 el id de la transacción, por último la columna 5 es la posición en la salida, por ejemplo primera salida, salida número 2, etc.

Tain_utxo.csv
Este archivo contiene todas las *utxo* que quedaron sin utilizar luego de la corrida del proceso. La primera columna es el *id* de la transacción, la columna dos es el índice en las salidas de la transacción y por último la tercera columna es la dirección. Para observarlo graficamente dirigirse a la figura 18 del anexo.

Se realizaron modificaciones al *address_info.csv* para mejorar la información provista, se agregó el *id* de la transacción ya que como estaba representado como un byte no aportaba nada visualmente.

Las principales pruebas realizadas fueron poniendo en el archivo de entrada direcciones que eran partícipes en la *blockchain*, al declarar éstas como entintadas, por un tipo particular, en el *address_info.csv* aparecían líneas representando las direcciones finales que habían quedado entintadas. Por otra parte, cuando en el archivo de entrada se ponían direcciones que no aparecían en ese tramo de la *blockchain* en la salida no se reportaba ninguna salida entintada.

En la sección Pruebas del capítulo Anexo se recopilan imágenes de una de las pruebas realizadas para comprobar si el algoritmo funcionaba. En la figura 19 se presenta la transacción inicial manchada por el ilícito *Allinvain Theft* una forma de robo. La figura 20 muestra la transacción llamémosle 1 en *BlockchainInfo*, donde se puede observar la dirección de entrada y las dos salidas. Por otra parte, la figura 21 muestra la transacción 2, en la cual es gastada la primera de las salidas de la transacción 1. Lo mismo sucede con la figura 22 que es la transacción que llamamos 3 en la que se utiliza la primera salidad de la transacción 2, de este forma se presenta un camino en el que las direcciones deben estar manchadas. En las figuras 23, 24, 25 se observan las salidas del algoritmo con marcada en amarillo la transacciones que se corresponden a las del camino antes mencionado y que las identifica como manchadas como debía ser.

7 Conclusiones y trabajo futuro

7.1 Conclusiones

El estudio y relevamiento de las distintas técnicas de clusterización y rastreo de direcciones de Bitcoin utilizados para reforzar o quebrar el anonimato provisto por dicha criptomoneda, arrojó varias conclusiones que se presentan a continuación.

Para el estudio de la información guardada en la *blockchain* se vio que el enfoque era clusterizar direcciones para tratarlas como distintas entidades. Estas entidades luego serían relacionadas con una persona real en los casos que ésto es posible. Las principales técnicas de clusterización se logran utilizando mayormente dos heurísticas que analizan cada transacción agrupando sus direcciones, asumiendo que cumplen cierta propiedad. Una de estas heurísticas agrupa las entradas, haciendo uso de una particularidad de las transacciones de Bitcoin, que ya que cada dirección de entrada debe ser firmada, es posible asumir que todas estas entradas corresponden a la misma entidad o persona. La segunda heurística es la llamada "de cambio" y se enfoca en analizar las salidas de la transacción para encontrar una de éstas que fuera usada para depositar el cambio de la transacción, en caso que exista, quiere decir que dicha dirección también puede asociarse con la entidad que se formó con las entradas. Siguiendo este proceso combinado, se recorren las distintas transacciones y se van obteniendo entidades cada vez más grandes. El siguiente paso para llegar a asociar las entidades obtenidas es estudiar la web, foros y hacer transacciones directamente con distintos servicios de forma tal de etiquetar con un nombre dicha entidad.

Todas estas técnicas observadas atacan el anonimato de Bitcoin, aunque se observa que éste puede ser quebrado aplicándolas, también se vio que llegar a relacionarlo con una persona es muy difícil en los casos que la persona opera de forma muy inteligente. Esto se da porque hay mecanismos que refuerzan el anonimato brindado por Bitcoin, algunos de éstos son Tor, los *mixers* encadenados y sistemas de mixado como *Coinjoin*. Se observó que si bien aplicando estos métodos se refuerza mucho la anonimidad en Bitcoin, no llegan a asegurar que una dirección no pueda ser asociada a una persona. Si se aplica el estudio y se llega a un posible sospechoso y se recupera el dispositivo con el que manejaba sus direcciones se podría finalmente hacer la relación. Un estudio parecido fue el realizado por el FBI para atrapar al creador de *Silk Road*. Como se comenta anteriormente, para llevar el anonimato al extremo surgieron nuevas propuestas como los son *Zerocoin*, aunque ésta necesita un *soft fork* del actual Bitcoin lo que lo vuelve impráctico, por otra parte, *Zerocash* dice tener el anonimato resuelto, pero como se explicó, no es compatible con Bitcoin.

Finalmente se analiza el desarrollo del grupo de investigadores de Cambridge presentado en el artículo "*Bitcoin Redux*", se llega a comprender cómo lograron realizarlo y se prueba que el método empleado para entintar direcciones de Bitcoin mediante el algoritmo FIFO funciona. En el proceso de trabajo con su implementación se observa que con los requisitos de las máquinas usadas era imposible trabajar con la *blockchain* completa, ya que en los primeros intentos se logró pero luego de más de 24hs de ejecución. Ésto llevó a que se hicieran todas las pruebas con reducida cantidad de bloques de la *blockchain*, bajando ahí sí la ejecución a algunos segundos o minutos dependiendo de la cantidad de bloques.

En conclusión, Bitcoin es una criptomoneda innovadora y cada vez más son los servicios que son ofrecidos con Bitcoin como medio de pago. Si bien sus ventajas son el ser descentralizada y pseudo anónima, los algoritmos analizados proveen de mecanismos transparentes y reproducibles de entintado de dinero, que permiten la detección del uso de monedas provenientes de ilícitos en las transacciones. En particular el algoritmo FIFO produce resultados razonables en cuanto a la concentración del entintado comparado con Haircut o Poison. El comportamiento de estos algoritmos en presencia de mixers es variado. En los tres casos alguien que puso Bitcoins limpias al *mixer* puede terminar con Bitcoins manchadas, en Poison y Haircut alcanza con que uno de los participantes del *mixer* deposite una entrada manchada para que todos los participantes terminen con sus Bitcoins manchadas. En cambio para el algoritmo FIFO, el participante que termina con sus Bitcoins manchados depende del ordenamiento de las salidas realizado por el *mixer*. Estos algoritmos podrían ser tenidos en cuenta por el regulador para reducir significativamente el riesgo asociado al uso de Bitcoin en la economía.

7.2 Trabajo a futuro

A continuación se presentan algunos trabajos a futuro:

Presentar el trabajo gráficamente. Tener el algoritmo desarrollado en formato gráfico permitiría observar los resultados de forma más práctica, además se podría observar cuáles son las transacciones que participaron en el entintado de una dirección y cuál fue la entrada y el ilícito.

Desarrollar los algoritmos *Poison* y *Haircut* en *Rust* a partir del desarrollo actual de *FIFO* en *Rust*. Ambos algoritmos se podrían desarrollar con los conocimientos adquiridos y tenerlos permitiría poder comparar gráficamente las diferencias de entintado que cada uno aplica.

Modificar el sistema para que no mantenga todo en memoria. Actualmente el algoritmo desarrollado va guardando todo en memoria, lo que hace imposible trabajar con grandes volúmenes a no ser que se tengan recursos altos. Se estimó que hacer una corrida inicial con toda la *blockchain* y guardar los datos obtenidos que antes se mantenían en memoria en una base de datos permitiría hacer las siguientes corridas más rápidas.

Bibliografía

[1] Arvind Narayanan et al. *Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction*. Princeton, NJ, USA: Princeton University Press, 2016. ISBN: 0691171696, 9780691171692.

[2] Satoshi Nakamoto. *Bitcoin: A Peer-to-Peer Electronic Cash System*. https://bitcoin.org/bitcoin.pdf (Accessed: 2019-07-11). Dec. 2008. URL: https://bitcoin.org/bitcoin.pdf.

[3] P. Ranakoti et al. "Deep web online anonymity". In: *2017 International Conference on Computing and Communication Technologies for Smart Nation (IC3TSN)*. Oct. 2017, pp. 215–219. DOI: 10.1109/IC3TSN.2017.8284479.

[4] Ross Anderson et al. "Bitcoin Redux". In: *17th Annual Workshop on the Economics of Information Security*. Innsbruck, Austria, 2018. URL: https://weis2018.econinfosec.org/wp-content/uploads/sites/5/2018/05/WEIS_2018_paper_38.pdf.

[5] SRoss Anderson et al. *The Taint Chain*. http://www.taintchain.org (Accessed: 2019-07-11). May 2018. URL: http://www.taintchain.org.

[6] Grupo de Acción Financiera Internacional (GAFI). *Directrices para un enfoque basado en riesgo para Monedas Virtuales*. (Accessed: 2019-07-11). 2015. URL: http://www.fatf-gafi.org/media/fatf/documents/Directrices-para-enfoque-basada-en-riesgo-Monedas-virtuales.pdf.

[7] Infobae. *El banco central del uruguay se amiga con blockchain*. (Accessed: 2019-20-10). 2018. URL: https://www.infobae.com/cripto247/mercados/2018/09/07/el-banco-central-de-uruguay-se-amiga-con-blockchain-mientras-que-toma-distancia-de-las-criptomonedas/.

[8] Stuart Haber and W. Scott Stornetta. "How to time-stamp a digital document". In: *Journal of Cryptology* 3.2 (Jan. 1991), pp. 99–111. ISSN: 1432-1378. DOI: 10.1007/BF00196791. URL: https://doi.org/10.1007/BF00196791.

[9] Leslie Lamport. "Paxos Made Simple". In: (Dec. 2001), pp. 51–58. URL: https://www.microsoft.com/en-us/research/publication/paxos-made-simple/.

[10] The Economist. *How a few companies are bitcoining it*. https://www.economist.com/business/2018/05/19/how-a-few-companies-

are-bitcoining-it (Accessed: 2019-07-11). 2019. URL: https://www.economist.com/business/2018/05/19/how-a-few-companies-are-bitcoining-it.

[11] Rust Team Mozilla. *Rust*. https://www.rust-lang.org/ (Accessed: 2019-07-11). Dec. 2010. URL: https://www.rust-lang.org/.

[12] Bitcoin Fog. *Accesing Bitcoin Fog*. (Accessed: 2019-08-26). 2009. URL: https://bitcoinfog.info/.

[13] Coinjoin. *CoinJoin*. (Accessed: 2019-08-26). 2019. URL: https://www.investopedia.com/terms/c/coinjoin.asp.

[14] Ethan Heilman et al. "TumbleBit: An Untrusted Bitcoin-Compatible Anonymous Payment Hub". In: *24th Annual Network and Distributed System Security Symposium, NDSS 2017, San Diego, California, USA, February 26 - March 1, 2017*. 2017. URL: https://www.ndss-symposium.org/ndss2017/ndss-2017-programme/tumblebit-untrusted-bitcoin-compatible-anonymous-payment-hub/.

[15] Alan Brener. "Payment Service Directive II and Its Implications". In: *Disrupting Finance: FinTech and Strategy in the 21st Century*. Ed. by Theo Lynn et al. Cham: Springer International Publishing, 2019, pp. 103–119. ISBN: 978-3-030-02330-0. DOI: 10.1007/978-3-030-02330-0_7. URL: https://doi.org/10.1007/978-3-030-02330-0_7.

[16] Andreas M. Antonopoulos. *Mastering Bitcoin: Unlocking Digital Crypto-Currencies*. 1st. O'Reilly Media, Inc., 2014. ISBN: 1449374042, 9781449374044.

[17] B. Shanmugam et al. "A critical review of Bitcoins usage by cybercriminals". In: *2017 International Conference on Computer Communication and Informatics (ICCCI)*. Jan. 2017, pp. 1–7. DOI: 10.1109/ICCCI.2017.8117693.

[18] Inc The Tor Project. *Browse Privately, Explore Freely*. https://www.torproject.org/ (Accessed: 2019-07-11). Dec. 2006. URL: https://www.torproject.org/.

[19] Kim Zetter. *How the Feds Took Down the Silk Road Drug Wonderland*. (Accessed: 2019-07-11). 2013. URL: https://www.wired.com/2013/11/silk-road/.

[20] Y. Liu et al. "Enhancing Anonymity of Bitcoin Based on Ring Signature Algorithm". In: *2017 13th International Conference on Computational*

Intelligence and Security (CIS). Dec. 2017, pp. 317–321. DOI: 10.1109/CIS.2017.00075.

[21] Q. Wang, X. Li, and Y. Yu. "Anonymity for Bitcoin From Secure Escrow Address". In: *IEEE Access* 6 (2018), pp. 12336–12341. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2017.2787563.

[22] Rosario Gennaro et al. "Secure Distributed Key Generation for Discrete-Log Based Cryptosystems". In: *Journal of Cryptology* 20.1 (Jan. 2007), pp. 51–83. ISSN: 1432-1378. DOI: 10.1007/s00145-006-0347-3. URL: https://doi.org/10.1007/s00145-006-0347-3.

[23] Malte Möser and Rainer Böhme. "The price of anonymity: empirical evidence from a market for Bitcoin anonymization". In: *Journal of Cybersecurity* 3.2 (Aug. 2017), pp. 127–135. ISSN: 2057-2085. DOI: 10.1093/cybsec/tyx007. eprint: http://oup.prod.sis.lan/cybersecurity/article-pdf/3/2/127/23721609/tyx007.pdf. URL: https://doi.org/10.1093/cybsec/tyx007.

[24] I. Miers et al. "Zerocoin: Anonymous Distributed E-Cash from Bitcoin". In: *2013 IEEE Symposium on Security and Privacy*. May 2013, pp. 397–411. DOI: 10.1109/SP.2013.34.

[25] E. B. Sasson et al. "Zerocash: Decentralized Anonymous Payments from Bitcoin". In: *2014 IEEE Symposium on Security and Privacy*. May 2014, pp. 459–474. DOI: 10.1109/SP.2014.36.

[26] Dorit Ron and Adi Shamir. "How Did Dread Pirate Roberts Acquire and Protect his Bitcoin Wealth?" In: *Financial Cryptography and Data Security*. Ed. by Rainer Böhme et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 3–15.

[27] Kyle Chayka. "Everything You Need to Know About the Mt. Gox Bitcoin Debacle." In: *Time.com* (2014), p. 1. ISSN: 0040781X. URL: https://proxy.timbo.org.uy:443/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=edb&AN=94781971&lang=es&site=eds-live.

[28] Sarah Meiklejohn et al. "A Fistful of Bitcoins: Characterizing Payments Among Men with No Names". In: *Proceedings of the 2013 Conference on Internet Measurement Conference*. IMC '13. Barcelona, Spain: ACM, 2013, pp. 127–140. ISBN: 978-1-4503-1953-9. DOI: 10.1145/2504730.2504747. URL: http://doi.acm.org/10.1145/2504730.2504747.

[29] CriptoLine News. *Millions in Lost Bitcoins are gone forever. What happens next?* https://www.cryptolinenews.com/bitcoin-101/lost-bitcoins-gone-forever/ (Accessed: 2019-07-11). 2019. URL: https://www.cryptolinenews.com/bitcoin-101/lost-bitcoins-gone-forever/.

[30] Jeff John Roberts and Nicolas Rapp. *Exclusive: Nearly 4 Million Bitcoins Lost Forever, New Study Says*. http://fortune.com/2017/11/25/lost-bitcoins/ (Accessed: 2019-07-11). 2017. URL: http://fortune.com/2017/11/25/lost-bitcoins/.

[31] D. Ermilov, M. Panov, and Y. Yanovich. "Automatic Bitcoin Address Clustering". In: *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*. Dec. 2017, pp. 461–466. DOI: 10.1109/ICMLA.2017.0-118.

[32] D. D. F. Maesa, A. Marino, and L. Ricci. "Uncovering the Bitcoin Blockchain: An Analysis of the Full Users Graph". In: *2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*. Oct. 2016, pp. 537–546. DOI: 10.1109/DSAA.2016.52.

[33] Ahmad-Reza Sadeghi. *Financial Cryptography and Data Security: 17th International Conference, FC 2013, Okinawa, Japan, April 1-5, 2013, Revised Selected Papers*. Springer Publishing Company, Incorporated, 2013. ISBN: 3642398855, 9783642398858.

[34] Bitcoin Forum Dree12. *List of Major Bitcoin Heists, Thefts, Hacks, Scams, and Losses*. (Accessed: 2019-08-25). 2012. URL: https://bitcointalk.org/index.php?topic=83794.0;all.

[35] Trend Micro. *The history of ransomware: From CryptoLocker to Onion*. (Accessed: 2019-08-25). 2014. URL: https://blog.trendmicro.com/the-history-of-ransomware-from-cryptolocker-to-onion/.

[36] RustyTaintChain. *Rusty Taint Chain*. (Accessed: 2019-08-26). 2019. URL: https://github.com/TaintChain/RustyTaintChain.

[37] Grupo de proyecto. *Código ejecutable*. (Accessed: 2019-29-11). 2019. URL: https://github.com/fabriciog09/rusty-blockparser.

[38] Blockchain Luxemburgo SA. *Conectamos las criptomonedas con el mundo*. https://www.blockchain.com/es (Accessed: 2019-07-11). 2019. URL: https://www.blockchain.com.

[39] Blockchain Explorer. *BTC/Tx*. (Accessed: 2019-20-10). 2019. URL: https://www.blockchain.com/es/btc/tx/a1fd26c4cc3df0e4832dd5bc5aa004d2ca5050c659c6fcf302e41db3c2fc99f0

[40] getbitcoinblockchain.com. *getbitcoinblockchain.com*. (Accessed: 2019-07-11). 2019. URL: https://getbitcoinblockchain.com/.

[41] BlockchainInfo. *BTC/Transacción*. (Accessed: 2019-29-11). 2019. URL: https://www.blockchain.com/btc/tx/20fcd18e3d86513618fb058e3152806dd5e82954b2e304303aabbd7e909db51e

[42] BlockchainInfo. *BTC/Transacción*. (Accessed: 2019-29-11). 2019. URL: https://www.blockchain.com/btc/tx/08ebb06fdf4f91458d6d04fed79ee612e400b02ea46254594ccb3a5b634403d6

[43] BlockchainInfo. *BTC/Transacción*. (Accessed: 2019-29-11). 2019. URL: https://www.blockchain.com/btc/tx/4e818414811616dde41fe082bf9617d8bd3c90a624a9cc4457ef6c5619dd85af

Anexo

Salidas

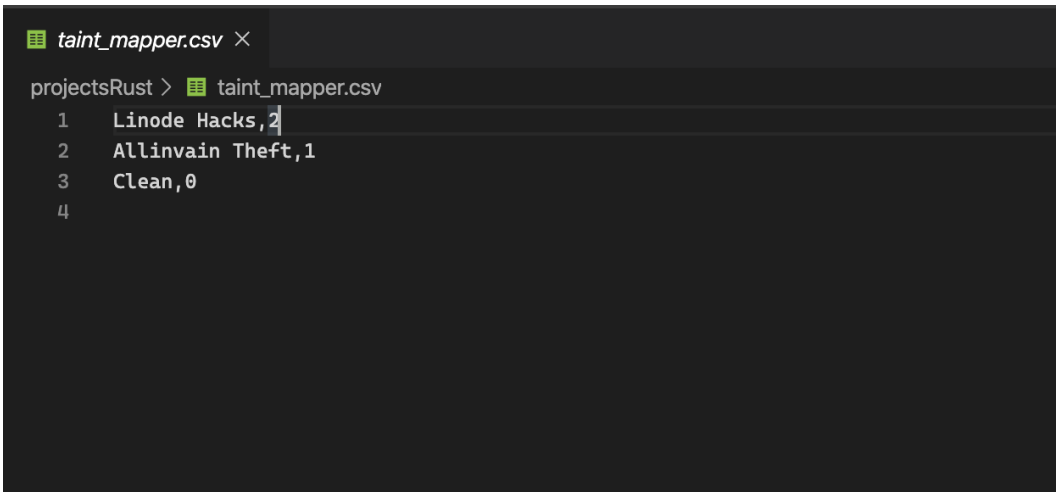

Fig. 16. Ejemplo taint_mapper.csv

Fig. 17. Ejemplo address_info.csv

Fig. 18. Ejemplo taint_utxo.csv

Pruebas

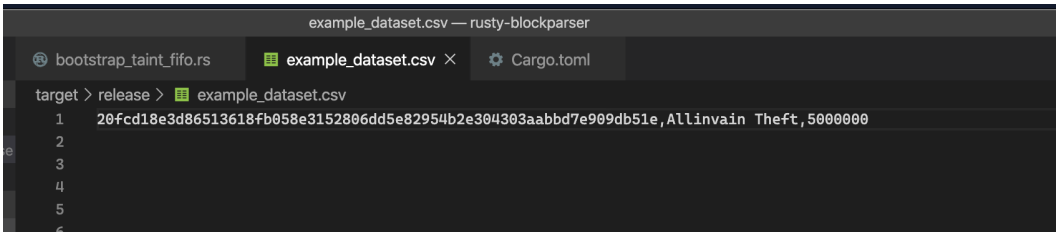

Fig. 19. CSV Input

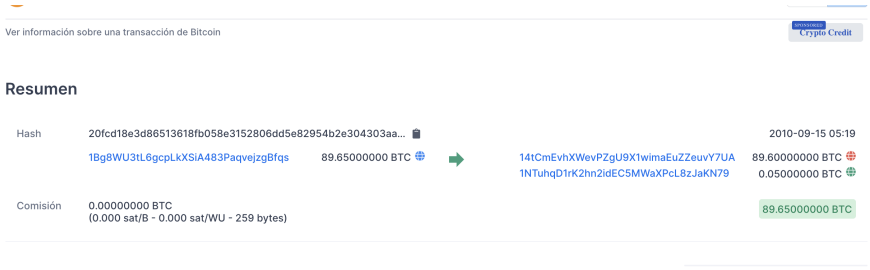

Fig. 20. Transacción 1 [41].

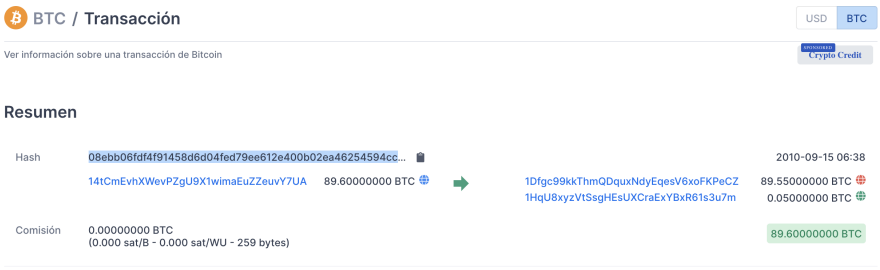

Fig. 21. Transacción 2 [42].

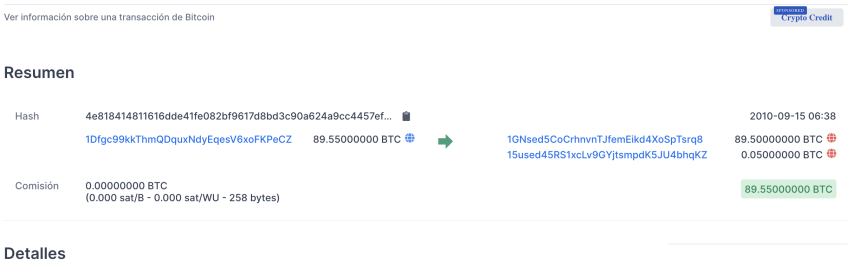

Fig. 22. Transacción 3 [43].

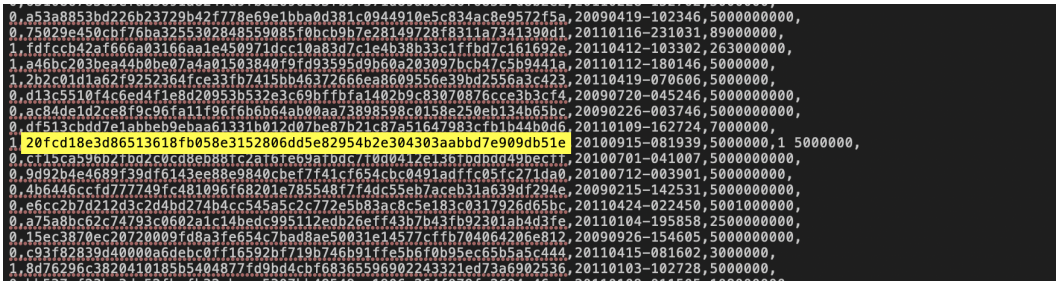

Fig. 23. Transacción 1 en salida

Fig. 24. Transacción 2 en salida

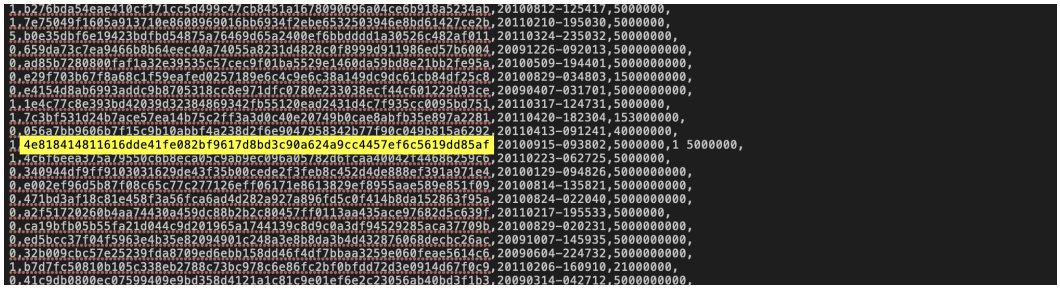

Fig. 25. Transacción 3 en salida