Anàlisi de la xarxa P2P de BitCoin

Alberto Pinilla Val

“Màster Interuniversitari de Seguretat de les Tecnologies de la Informació i Comunicació”

Cristina Pérez Solà

14/06/2013

Aquest treball està dedicat als meus pares i germà, així com a la meva dona.

El present treball de fi de Màster engloba un estudi del funcionament del sistema i en especial de la xarxa P2P utilitzada com a medi d'actuació del intercanvi de BTC. En la lectura del mateix, el lector podrà experimentar el coneixement del sistema de manera general, on hi trobarà referències als mètodes, funcionalitats i estructures, que fan possible que el sistema funcioni correctament. En una segona fase podrà trobar-ne demostracions pràctiques del funcionament de la xarxa així com petites conclusions extretes de les proves realitzades.

Veurem com les característiques del seu funcionament i estructura de la seva xarxa, basats en clau pública / privada, a més del seu sistema de transaccions a la vista de tothom i compartit per tothom, assenta les bases d'un sistema d'intercanvi monetari innovador, revolucionari i independent que proporciona a priori la confiança suficient per afirmar que les transaccions són fruit de la confidencialitat, privacitat, autenticitat i no repudi dels intercanvis entre els seus usuaris.

No és objectiu doncs d'aquest treball, definir la història, els esdeveniments ni els motius que han portat a terme el sistema BTC, sinó fer més èmfasi en la seva part tècnica, trobant la relació entre el funcionament teòric del sistema i els elements que formen part del mateix.

This "Master" work includes a study of the Bitcoin system and P2P network used as a medium of coins exchange. Along it, the reader can experience the knowledge of the system in general, where you will find references to methods, functions and structures that enable the system to work. On the second phase of this assignment you will find practical demonstrations on network as well as conclusions from tests.

We'll see how the characteristics of the structure and operation of the network, based on public / private key, in addition to its transaction system to public view shared by everyone, lays the groundwork for a system of monetary exchange innovative , revolutionary and independent that provides sufficient confidence to assert that transactions are the result of confidentiality, privacy, authenticity and non-repudiation of exchanges between users.

It is not therefore objective of this work to define history, motives or the events that make BTC system, but do a more emphasis on the technical side to find the relation between the system and his elements.

Taula de continguts

Índex de Figures

1. Introducció

1.1 Justificació del TFM

El sistema BITCOIN és un entramat de transaccions que fan possible l'existència d'una moneda virtual (BTC) equiparable al $ o al € però no de manera física.

BITCOIN presenta el seu funcionament com una xarxa d'intercanvi monetari virtual que no depèn de cap organització certificadora que auditi els processos que hi intervenen (Banc).

La responsabilitat d'aquesta tasca queda delegada al propi motor i funcionament del sistema. Això vol dir que recau sobre la pròpia xarxa P2P i la distribució dels seus processos el fet d'atorgar autenticitat, veracitat i seguretat als intercanvis monetaris que es realitzen.

Aquestes característiques són motius suficients per fer-ne un estudi del processos que hi intervenen des de l'inici de creació de l'intercanvi fins al final del mateix, i com la xarxa global de bitcoin es veu alterada en cadascuna d'aquestes transferències d'informació.

És per això que aquest projecte pretén tant de manera teòrica com pràctica reproduir l'escenari de la xarxa Bitcoin fent èmfasi en l'estudi de la seva xarxa P2P, els processos que se'n deriven de les transaccions i els components que formen part del sistema, de manera tal, que es clarifiqui el funcionament d'aquest sistema monetari virtual.

Com escenari inicial referent a aquesta temàtica trobem un conjunt de documentació oficial en Anglès, que ens ha estat de referència per la part teòrica d'aquest treball.

Com aportació principal doncs, destacaríem que el present TFM proporciona les nocions bàsiques del sistema de funcionament Bitcoin i la seva xarxa P2P a més de diverses proves pràctiques al voltant del seu concepte teòric.

1.2 Objectius del TFM

L'objectiu principal d'aquest TFM és estudiar el flux d'informació de Bitcoin a través de la xarxa P2P.

Dins d'aquest estudi de la xarxa P2P ens pot ser d'interès esbrinar els temps de transferència de les transaccions entre dos usuaris, la prioritat i velocitat amb la que es transmeten els blocs etc.

Això i tal i com veurem mes endavant, es farà mitjançant l'aglutinament de diferent informació teòrica i pràctica que permeti al lector conèixer de manera més profunda, el sistema bitcoin i tots els processos que l'envolten, així com de manera més pràctica observar l'intercanvi de transaccions.

A més a més d'aquests objectius, el TFM permetrà obtenir conclusions derivades de les proves pràctiques en relació a les variables que formen part del sistema i les seves transaccions.

Tots aquests objectius s'assoliran mitjançant:

- Una definició inicial del sistema Bitcoin i les seves estructures
- Una visió mes profunda dels missatges que es transfereixen als nodes que formen part del sistema bitcoin
- Estudi del graf de transaccions de manera teòrica i pràctica.
- Ús d'eines que permeten obtenir informació pràctica i en directe de les transaccions que succeeixen per tal de treure conclusions referents a la propagació del flux d'informació per tal d'avalar el funcionament del sistema i permetin provar autenticitat, privacitat, no repudi i confidencialitat.
- Observatori dels canvis que pateix la xarxa envers les transaccions i la creació de nous blocs.

També és part d'aquests objectius respondre a les preguntes que han anat sorgint al llarg de la realització del treball.

Aquestes preguntes pretenen ajudar a guiar el TFM al llarg de les seva part teòrica i pràctica.

El conjunt de preguntes correspondria a les següents:

Teòrica.

- Quants nodes hi ha connectats a la xarxa Bitcoin? És viable connectar-se a tots simultàniament?
- On s'estan originant les transaccions? I els blocs?
- Com és la xarxa de Bitcoin a nivell topològic?
- Quina quantitat de nodes es connecten a través de Tor?
-Quin és el cicle de compartició de les transaccions entre els nodes?
-Com es generen els blocs i les cadenes de blocs?
-Quin tipus de transaccions hi han i en que es diferencien
-Com es creen i s'identifiquen les bitlleteres i quin paper juguen en el sistema?
-Quina funció tenen els miners?
-Per que hi ha un número màxim de BTC que hi poden coexistir?
-Que passa si perds el client bitcoin del teu ordinador?
-Com funcionen les empreses que et guarden les bitlleteres?
-Com podem xifrar la nostra bitlletera?

Pràctica

- Cóm s'intercanvien els diferents missatges entre els nodes?
- Quines característiques de les transaccions afecten la velocitat amb la que es retransmeten per la xarxa ? (per exemple, es tenen en compte les fees, i la prioritat?)
- Què passa si indiquem una adreça errònia a l'hora de traspassar els bitcoins?(Cóm es veu reflectit aquest error dins la xarxa)

1.3 Enfocament i mètode seguit

Com podreu apreciar, s'ha seguit un esquema Teòric / Pràctic per la realització d'aquest treball. Això significa que en una primera part s'expliquen diversos apartats referents al funcionament teòric del sistema Bitcoin per posteriorment aplicar certes proves pràctiques que permetran verificar el correcte funcionament del propi sistema.

Per la recollida d'informació referent al funcionament s'ha pres nota de la informació publicada a la web oficial de bitcoin. Pel que fa a la part pràctica, està basada en l'ús del sniffer wireshark per la realització de diferents tasques d'investigació.

Durant aquesta part de proves amb wireshark, el lector podrà experimentar a un nivell baix, el funcionament dels mètodes descrits en la part teòrica, com ara la cerca de nodes, la connectivitat directa amb un dels nodes de la xarxa o la verificació d'una transacció entre dos adreces.

1.4 Planificació del projecte

Per tal d'assolir els objectius respectant al màxim les dates per presentar els resultats, es va optar per dividir en dos blocs la feina a realitzar.

Així doncs tindríem tasques d'un primer bloc dedicades a esbrinar el funcionament teòric del aplicatiu i sistema bitcoin mitjançant l'estudi dels components i procediments que intervenen.

D'altra banda i en un segon bloc, ens vam dedicar a investigar de manera pràctica amb transaccions bitcoin i mitjançant eines que escolten aquestes transaccions i en registren els moviments, vam poder interpretar la primera part del treball.

Al finalitzar el treball s'han extret unes conclusions com a resposta (al llarg dels dos blocs) d'algunes de les preguntes que hem definit als objectius.

Planificació inicial.

Bloc 1 (Pac2)

- Investigació prèvia general sobre Bitcoin

- Definició dels components/processos que intervenen en el funcionament de BicCoin

-Estudi del graf de transaccions de manera teòrica i pràctica.

Figura 1

Bloc 2

- Proves pràctiques de les funcionalitats de la xarxa per corroborar el funcionament intern dels processos

-Anàlisi del comportament de la xarxa envers els intercanvis de bitcoins i creació de nous blocs.

-Generar un informe de conclusions del sistema tot indicant els punts forts del sistema i els possibles punts febles

Figura 2

-Planificació temporal (Cronograma de GANTT)

Bloc 1.1 → Investigació BITCOIN

Bloc 1.2 → Estudi i definició components/processos BITCOIN

Bloc 1.3 → Primeres pràctiques d'anàlisi del graf i de la xarxa (Pac2)

Bloc 2.1 → Proves pràctiques d'anàlisi de propagació

Bloc 2.2 → Anàlisi afectació de creació de nous blocs i noves transaccions.

Bloc 2.3 → Conclusions (Pac3)

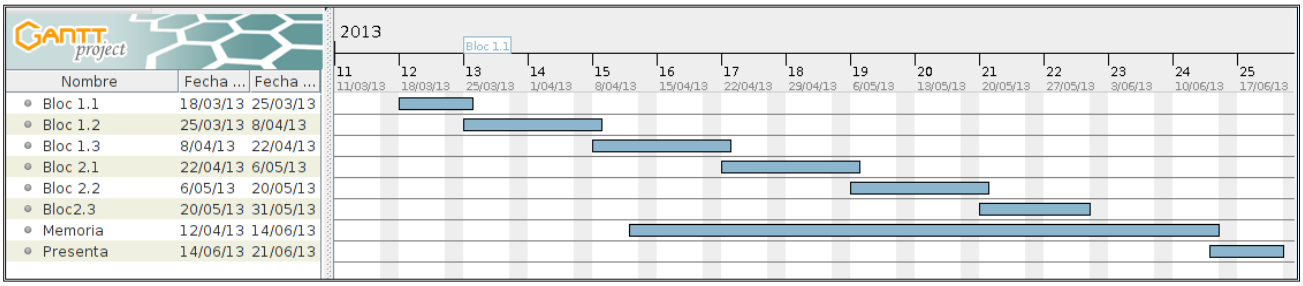

Figura 3

Planificació real

Com moltes vegades passa amb els projectes, van sorgir complicacions temporals que van fer que m'hi dediqués més temps de l'esperat a l'estudi de la part teòrica del sistema i de la part de P2P.

Això fa que mitjançant les conclusions sorgeixin noves línies de futur per omplir les fites que no s'han pogut completar amb aquest projecte..

Com podreu observar, hi han preguntes que aquest document no podrà respondre.

Veiem la realitat de la planificació temporal mitjançant el diagrama de Gantt anterior.

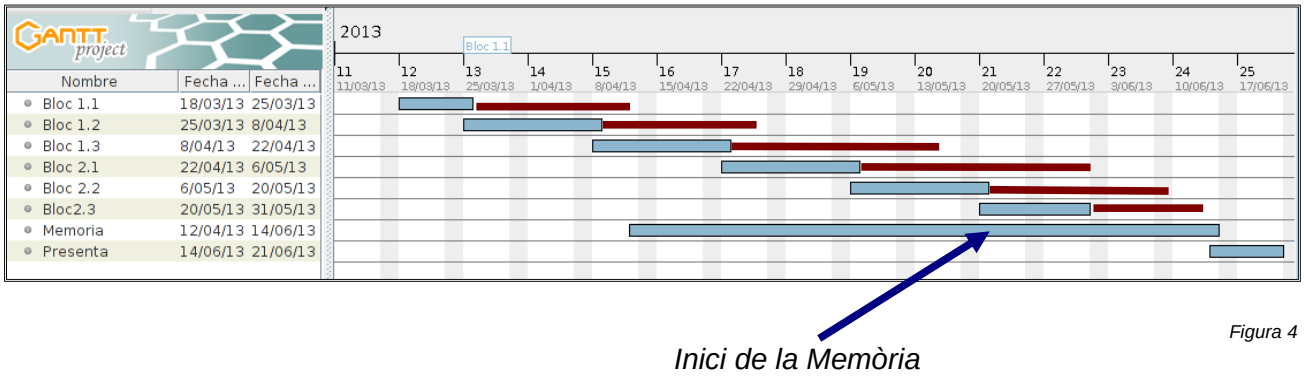

Figura 4

1.5 Breu descripció d'altres capítols

Al capítol 2 d'aquest treball s'explicaran els elements, processos i estructures que formen part d'aquest sistema d'intercanvi monetari.

A més a més, en el capítol posterior (Capítol 3) es realitzaran certes proves dins la xarxa, que permetran al lector d'aquest projecte verificar mitjançant imatges i captures dels elements investigats, que el sistema funciona tal i com indica la teoria.

D'altra banda, aquests experiments ens permetran treure conclusions referents a les variables que entren en joc en el sistema P2P i com aquestes, influeixen en el funcionament del propi sistema.

D'altra banda i per acabar el treball s'afegeix al capítol quatre una breu descripció de les conclusions obtingudes i possibles línies de futur mitjançant les fites que no s'han portat a terme en aquest projecte.

S'ha inclòs un capítol amb un annex que serà molt útil per al lector, ja que ens presenta la base del protocol de missatges que s'intercanvien els nodes. A més ens serà una bona referència a l'hora d'identificar les diferents transaccions i intercanvis mostrats a la part pràctica.

Finalment s'ha complementat amb un glossari dels termes més importants del sistema per ajudar a entendre el treball.

2. Coneixements bàsics del funcionament del sistema Bitcoin

Aquest treball pretén apropar el concepte actual de BITCOIN i el seu funcionament com a moneda electrònica, mostrant que una de les seves característiques principals és el seu funcionament independent de qualsevol tipus d'Autoritat Certificadora i Reguladora.

Aquesta premissa, que assenta les bases del sentit del Sistema Bitcoin, és feta realitat amb un conjunt de processos, on diferents components del sistema formen part d'una gran xarxa de P2P.

És finalitat d'aquest projecte doncs, ajudar al lector a concebre el funcionament del Sistema monetari Bitcoin així com la gran xarxa P2P que el fa funcionar, mostrant especial interès en els elements que formen part del procés de certificació dels intercanvis de moneda.

Per complir aquesta finalitat, primerament definirem els components que formen part de la xarxa per posteriorment i de manera més gràfica, relacionar els components amb els processos en els que hi formen part.

El bitcoin com a moneda i sistema, va néixer al 2009 a mans de Satoshi Nakamoto. Motivat per la característica principal del sistema, l'autor o autors van tenir que dissenyar una xarxa, que permetés, gràcies a la col·laboració de tots els implicats, que no fos necessària una entitat emissora de moneda i certificadora centralitzada que avalés l'intercanvi de moneda.

Aquesta idea la va plasmar en la xarxa P2P que sustenta el projecte del Bitcoin i en un sistema de repte criptogràfic que permetés el continu còmput de transaccions xifrades.

Però, quins components formen part d'aquest sistema o protocol de comunicació?

- Client Bitcoin
- Adreça
- Transaccions
- Blocs
- Cadena de blocs
- Miner
- Repte o problema criptogràfic
- Cartera o Wallet
- Clau Pública
- Clau Privada

Figura 5

2.1 Components

El Client Bitcoin

El Client Bitcoin és un software que ens permet connectar-nos a la xarxa P2P bitcoin i realitzar-ne o rebre transferències de bitcoins.

En la instal·lació del client Bitcoin ve implícitament, la creació d'una Cartera o Wallet que ens permetrà emmagatzemar les Claus Privades en el nostre computador personal. Més endavant definirem la Cartera i com s'emmagatzema aquesta clau.

L'adreça

L'adreça és un identificador que pot estar entre 27 i 34 caràcters alfanumèrics i representa la destinació de les transaccions de bitcoins. Tal i com si d'un e-mail es tractés, l'adreça permet ser distribuïda entre destinatari i emissor per poder realitzar-ne mitjançant les transaccions, transferències de bitcoins.

El client i el sistema bitcoin, ens permet generar tantes adreces com vulguem, el que ens ajudarà a gestionar els pagaments ja que podrem crear tantes adreces com transferències de bitcoins hi fem.

La creació d'aquestes adreces no influeix a la xarxa P2P, per que es poden crear de manera local i sense estar connectats. Això és possible perquè només hi entren en joc en el sistema si són objecte destí d'alguna transferència de bitcoins.

S'ha de dir però que en el procés de generació de l'adreça influeix la clau privada generada inicialment (amb la instal·lació del client de bitcoin) i que aquesta informació (Clau Privada) està emmagatzemada en un fitxer anomenat wallet.dat o moneder.

No cal doncs, saber que és molt important mantenir en un lloc segur les claus privades amb que hem generat les adreces, ja que, tot i que les adreces són xifres públiques, si no tenim la clau privada amb la que les hem generat, no podrem realitzar-ne moviments amb els bitcoins que hi tenen assignats.

Les transaccions

Quan des d'un client i més exactament des d'una adreça amb bitcoins realitzem un traspàs de moneda a una altre adreça, estem generant una transacció.

Aquesta transacció, que és un conjunt de dades firmades digitalment, es propaga per la xarxa d'un node a un altre, mitjançant un missatge de transacció signat digitalment per l'emissor. Això permet als nodes que reben el missatge verificar la certesa del mateix.

Com el seu nom indica, una transacció és això que diu, un canvi, un moviment d'alguna cosa d'un costat a un altre.

Important!!: El concepte del canvi

Quan es realitza una transferència de bitcoins d'una adreça a una altre es generarà una transacció amb una sortida per ingressar els bitcoins a l'adreça destinatari i una altre sortida per ingressar el "canvi" a l'adreça d'origen.

Aquest concepte s'entén només si entenem transacció com a transferència total de bitcoins d'un costat a un altre.

Si observem la definició de transacció en el sistema Bitcoin, hem de tenir en compte dos conceptes principals.

Entrada: Una entrada o input en una transacció Bitcoin, defineix l'origen dels bitcoins que es volen transferir.

-*Previous Tx*: Aquest valor fa referència al hash de l'anterior transacció,(Es a dir l'última transacció en la que el client que envia els bitcoins, va rebre els bitcoins que hi te actualment en la seva adreça).

-*Index:* Index de la sortida concreta d'origen.

-*ScriptSig:* Verifica la veracitat de la transacció (Primera part del script)

Sortida: Una sortida o output d'una transacció Bitcoin, defineix la destinació dels bitcoins que es volen transferir.

-*Value*: el valor de la transferència de bitcoins.
-*scriptPubkey:* Es l'adreça destinació dels bitcoins.

Exemple:

Si tenim a una adreça bitcoin 30 BTC i volem traspassar 25 a una altre adreça i ho indiquem al client BTC, automàticament generarem 1 transacció d'una entrada de 30 BTC i dos sortides a dos adreces diferents. Una d'aquestes sortides entrarà 25 BTC a l'adreça destí i els altres 5 bitcoins retornaran a l'adreça d'origen de la transferència.

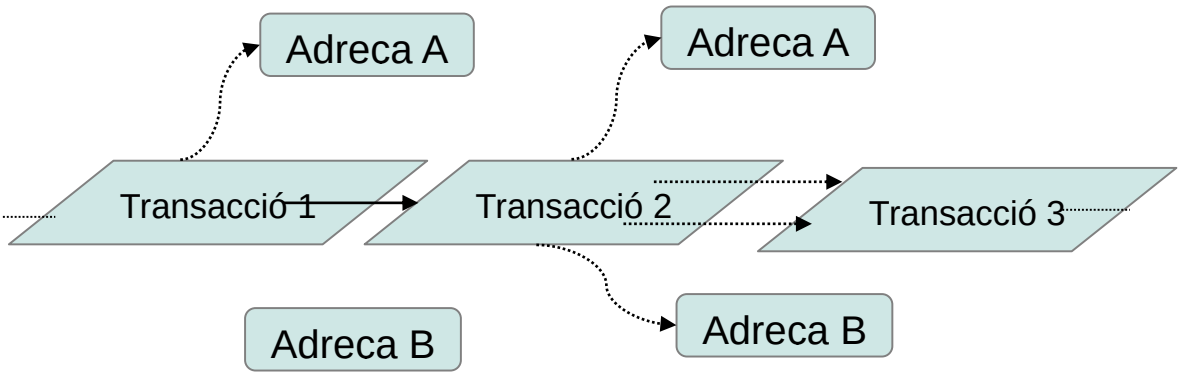

Figura 6

Existeixen dos tipus de transacció de bitcoins, però nosaltres ens centrarem en destinacions d'adreces de bitcoin generades mitjançant un client. Existeix un altre tipus de transacció a una adreça IP.

Transacció a una adreça BTC

Important!!! Transaccions de nous BTC

En la creació de nous BTC fruit de la generació gràcies als miners, es genera un nou tipus de transacció on en comptes de tenir la variable scriptSig i te coinbase, variable que no s'utilitza i queda anul·lada.

Si recordem, abans hem mencionat les variables transaccionals scriptPubKey i scriptSig de sortida i entrada de transaccions.

Aquestes variables li serveixen al sistema per corroborar la validesa de les transaccions.

L'enviament de transaccions es valida mitjançant la clau pública dels remitents. Els missatges són signats pels remitents de les transaccions i els clients que reben la transacció validen mitjançant la clau pública dels remitents que la transacció ha estat signada pel remitent de la transferència.

Quan el destinatari de la transacció vol realitzar-ne una transferència de BTC, scriptSig verifica que la signatura conté el mateix valor hash que la clau publica, així com el hash que apareix a scriptpubkey.

Els blocs

Els blocs, són estructures de dades que contenen entre altres valors, una quantitat determinada de transaccions i el seu valor total.
A més a més conté l'adreça del bloc anterior i el tamany del bloc en si.

També conté el repte o problema matemàtic que han de resoldre els miners per generar un altre bloc nou i finalitzar l'actual.

La recompensa per finalitzar un bloc i resoldre el problema matemàtic és una quantitat determinada de BTC i aquest és el motiu de l'existència de miners i del funcionament de la xarxa de creació de blocs.

Les cadenes de blocs

La cadena de blocs és el conjunt de blocs que conté totes les transaccions realitzades des de l'inici del sistema i que es va escrivint a mesura que hi van apareixent més transaccions.

En el procés de creació de la cadena de blocs, es generen uns altres blocs que s'anomenen orfes ja que són blocs que no seran resolts mai ja que la cadena de blocs ja ha finalitzat un altre bloc en la mateixa base.

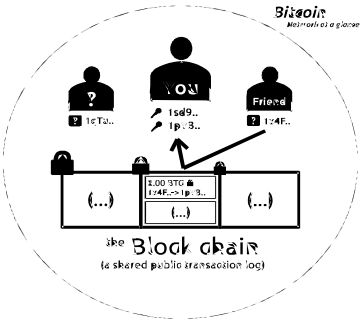

Figura 7

Els miners

Els miners són els encarregats de completar els blocs de transaccions amb la finalitat d'emportar-se els BTC de recompensa i regular i validar les transaccions mitjançant el procés de mineria.

Actualment existeixen grups de miners que fan treballar les seves màquines conjuntament amb l'objectiu de trobar-ne la recompensa i compartir-la entre la resta de miners del grup.

És una lluita constant per analitzar un bloc per finalitzar i trobar-ne la solució al problema matemàtic.

El repte criptogràfic

En relació al component anterior hi ha el repte criptogràfic, que és el problema que se'ls presenta als miners per que el resolguin i rebin la recompensa. El sistema autoregula aquest procés de tal manera que la generació d'un nou bloc no demori més de 10 minuts.

Cartera

Com hem comentat anteriorment, la cartera és un fitxer anomenat wallet.dat, que conté un seguit de claus privades que permeten signar les transaccions associades a les seves adreces bitcoin.

Això, com veurem més endavant, és essencial en la xarxa P2P, ja que la resta de clients de la

xarxa verificarà que les adreces que envien les transferències de bitcoins, estan signades correctament.

Aquest punt és un dels detalls importants de la verificació de *l'autenticitat de la transferència*.

Clau Publica

Com hem comprovat en l'apartat de les transaccions la clau pública de cadascun dels clients permet a la resta de nodes validar l'autenticitat de les transaccions que inicia cadascun dels usuaris.

Gracies aquesta característica i mitjançant un procés anomenat confirmació, la xarxa P2P de BTC pot donar per vàlida o no una transferència de bitcoins.

Clau Privada

La clau privada pertany a la part de la cartera del client Bitcoin. Aquesta clau és la base per la creació d'adreces, signatures i claus públiques dins del sistema Bitcoin. És una clau única i la pèrdua o modificació de la mateixa fa que automàticament no puguis realitzar-ne transaccions amb les adreces que hi havies creat amb ella.

Aquesta clau està emmagatzemada en un arxiu anomenat wallet.dat i pot ser xifrada amb algorismes criptogràfics mitjançant el client Bitcoin.

El procés que xifra la bitlletera està basat en AES256 utilitzant una clau que deriva d'un hash SHA-512 d'una contrasenya que ens demana el client quan volem xifrar la bitlletera, a més d'un salt generat aleatòriament.

Tot i això s'ha de posar especial atenció a la part del procés en la que se'ns demana una password i introduir-ne una que en sigui difícil de trencar per força bruta (Es a dir Majúscules, minúscules números i caràcters que no signifiquin res).

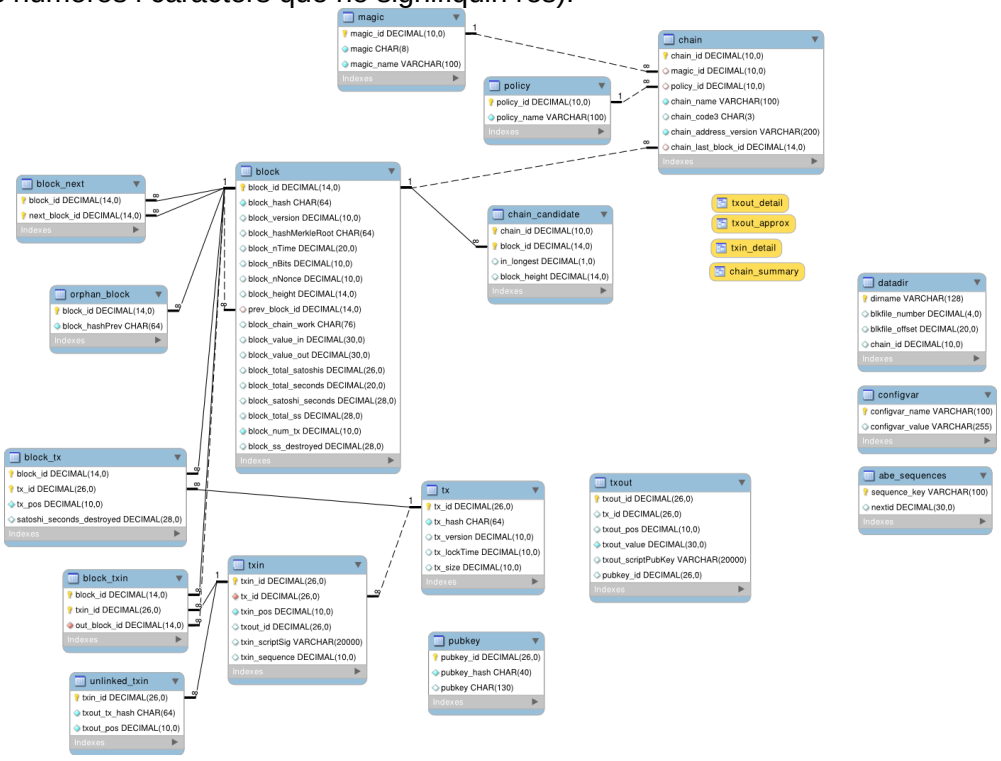

Figura 8

2.2 Xarxa Bitcoin

La capa de xarxa que utilitza bitcoin per realitzar-ne la difusió de les transferències és el TCP. A més a més Bitcoin és capaç de treballar en ports variables.

El mode de funcionament de la xarxa P2P bitcoin és molt semblant a qualsevol xarxa de transmissió de dades per compartició de nodes.

Aquest tipus de xarxes estan basades en intercanvis de missatges que realitzen diferents tipus de peticions entre els nodes que estan interconnectats. Al mateix temps, aquests nodes estan interconnectats a uns altres i aquets altres a uns altres, el que permet compartir dades entre nodes que no estan connectats directament.

A continuació es llisten i es defineixen els diferents missatges de comunicació dels nodes bitcoin dins la xarxa TCP.

- *version* - Mostra la informació referent a la versió del programa. I el número de blocs comptabilitzats pel client.
- *verack* – Resposta de ack al missatge de versió.
- *addr* – Llista de les ip's i ports.
- *inv* – Missatge amb informació rellevant als blocks i transaccions.
- *getdata* – Petició d'un bloc o transacció per un hash determinat.
- *getblocks* – Petició d'un determinat grup de blocs.
- *getheaders* – Petició de les capçaleres de blocs d'un determinat rang.
- *tx* – Envia una transacció com a resposta d'una petició getdata.
- *block* – Envia un bloc com a petició d'un missatge getdata.
- *headers* – Serveix per enviar capçaleres de blocs en comptes d'enviar el bloc sencer.
- *getaddr* – Missatge de petició de grups d'adreces amb els nodes actius.
- *submitorder*, *checkorder*, and *reply* – Aquests missatges s'utilitzen quan es realitza una transacció. (realitzar, verificar, replicar a la resta de nodes)
- *alert* – Missatge d'alerta a la xarxa.
- *ping* – Comprova que la connexió roman activa..

A l'annex podreu trobar una especificació més amplia dels missatges que s'intercanvien als nodes així com una definició més tècnica dels elements que formen part de cadascun d'ells.

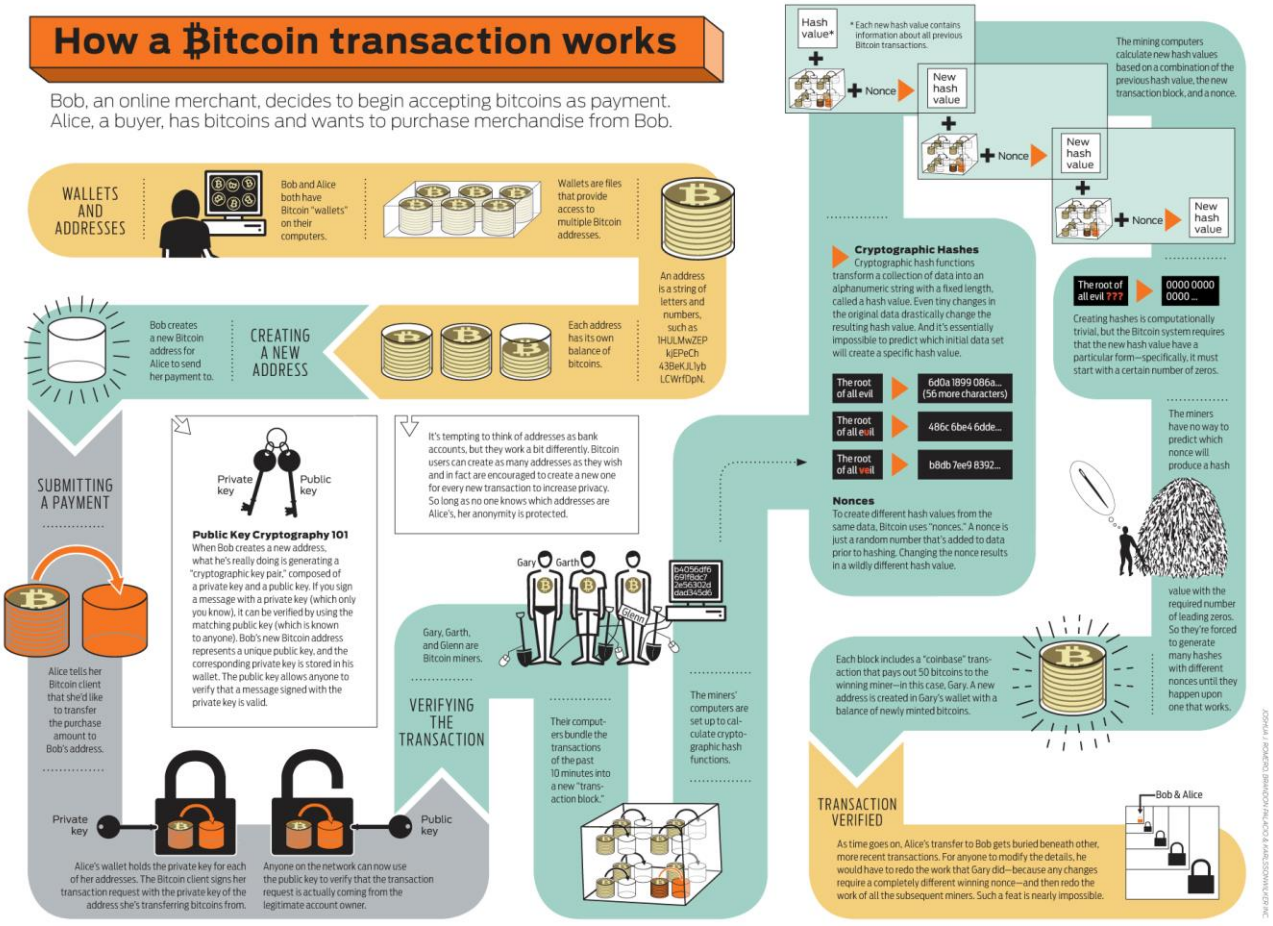

Figura 9

2.3 Processos

Podríem concretar, que bàsicament hi juguen un paper essencial, el procés de creació de blocs i generació de BTC associats, el de Propagació de les transaccions que van ocorrent, el de confirmació i validació d'aquestes mateixes i el de connexió.

A banda d'especificar subprocessos com ara la generació d'adreces o la comprovació de l'especificació correcte d'aquestes, passarem a explicar-ne una mica més detallat els processos bàsics de bitcoin.

2.3.1Càrrega inicial del sistema client

La informació d'aquest capítol s'ha basat en les nocions explicades a https://en.bitcoin.it/wiki/Satoshi_Client_Node_Discovery

- Metodologia de cerca de nodes

El problema del descobriment de nodes P2P a bitcoin ve resolt per un algorisme de cerca que conté mecanismes com el descobriment de la ip del propi node o l'actualització de la llista de nodes a connectar.

Si per exemple agafem el client GUI de bitcoin creat per Satoshi, observem que descobreix les

adreces ips i els ports dels nodes de diverses maneres diferents, basant-se en nou premisses.

Primera: Els nodes poden descobrir la seva ip externa mitjançant diferents mètodes que veurem més endavant.
Segona: Els nodes poden rebre l'adreça dels nodes que estan connectats a ells.
Tercera: Els nodes poden connectar-se a un servidor IRC per rebre les adreces
Quarta: Els nodes poden realitzar peticions DNS amb la finalitat de rebre les Adreces IP
Quinta: Els nodes poden utilitzar adreces que estan incloses de manera inicial en el codi del software client.
Sisena: Els nodes intercanvien adreces amb altres nodes.
Setena: Els nodes emmagatzemen adreces en la base de dades que llegeixen al inici de la càrrega del programa.
Vuitena: Els nodes poden rebre les adreces de manera manual mitjançant la línia de comandes.
Novena: Els nodes poden llegir les adreces d'un fitxer de text al inici del programa.

Lectura inicial de la BDD d'adreces

Del funcionament normal del bitcoin client, eventualment, s'executa la comanda *AddAddress* que va afegint les adreces que te present el node en aquell moment a la base de dades d'adreces.

A l'inici es pot intentar fer la connexió amb els nodes mitjançant una consulta a la base de dades d'Adreces fent la crida *LoadAddresses().*

Recepció d'adreces mitjançant IRC

Aquest mètode d'obtenció d'adreces es va deixa d'utilitzar a partir de la versió 0.6.x. El funcionament és molt semblant al descobriment d'ip externa del node local. En aquest cas però, mitjançant la comanda WHO, el node local connectat a un dels canals IRC, pot rebre una llista dels usuaris connectats(IP dels nodes remots).

Recepció d'adreces via DNS

El client bitcoin porta incorporada una llista de hosts DNS que serveixen en un inici per connectar a ells i descarregar-ne les adreces dels nodes més propers a ell.

La llista de hosts DNS que porta incorporat el client és la següent:

- bitseed.xf2.org
- dnsseed.bluematt.me
- seed.bitcoin.sipa.be
- dnsseed.bitcoin.dashjr.org

Lectura de les adreces hardCoded en el binari inicial

El codi font del client té incrustada una llista d'IP de nodes que permeten al node local realitzar-ne la connexió a la xarxa. S'ha de dir però que aquesta opció de connexió als nodes de la xarxa és l'últim recurs una vegada esgotats la resta de modes de connexió inicial.

Com que en un inici es connectarà el node local a les ip's d'aquests nodes, com hem explicat anteriorment, intercanviaran versions a més de la llista de nodes que els nodes hardcoded contenen en les seves llistes. D'aquesta manera i una vegada s'han intercanviat vàries llistes, el node local es desconnectarà dels nodes hardcoded per tal de alliberar-los.

Adreces introduïdes mitjançant la línia de comandes

Com a usuaris de la xarxa i del client bitcoin, podem interactuar mitjançant la línia de comandes. D'aquesta manera el client bitcoin pot emmagatzemar qualsevol adreça que li indiquem mitjançant el paràmetre *–addnode <ip>* on la ip es l'adreça ip del node. Amb això afegim a la llista d'adreces de nodes actius, l'adreça indicada.

D'altre banda existeix també la comanda *–connect <ip>,* que a diferencia de l'anterior comanda, permet connectar al node que li hem indicat. Això significa que mitjançant –connect podem connectar-nos únicament a un node actiu sense necessitat d'emmagatzemar-ne l'adreça. Això ens serà molt útil en el moment de realitzar-ne la part pràctica.

Carregar Adreces mitjançant fitxer de text al inici

A l'inici de la connexió, el client bitcoin pot llegir de manera automàtica el fitxer addr.txt disposat al directori de dades del client bitcoin. Aquest pot afegir adreces dins d'aquest fitxer si troba adreces noves per afegir.
Les adreces que hi ha dins d'aquesta llista no es retransmeten com a resposta a un *getaddr*.

- Descobriment de la IP externa del client bitcoin local

El client bitcoin pot utilitzar dos mètodes per esbrinar la seva ip externa local. Mitjançant IRC o mitjançant serveis web públics que retornen aquesta informació.

De manera normal, el client bitcoin prefereix realitzar la connexió mitjançant IRC però en el cas que no sigui possible aleshores intentarà obtenir la IP externa mitjançant serveis web destinats per tal efecte.

Mitjançant el descobriment via IRC, el client realitzarà una connexió IRC contra el servidor 92.243.23.21 o irc.lfnet.org en cas que la connexió contra la IP sigui fallida. El port utilitzat per aquesta tasca serà el 6667.

Si la connexió s'ha establert, el client envia una comanda demanant la seva IP i el servidor IRC li retorna. Cada dos minuts actualitza el valor de la ip mitjançant la consulta al servei IRC.

Si l'anterior mètode no funciona, el client intenta obtenir l'adreça IP connectant-se a diversos serveis de consulta de ip externa.
Primer de tot intenta mitjançant 91.198.22.70 pel port 80. Si això falla, fa una petició DNS a checkip.dyndns.org. Si això també falla connecta contra 74.208.43.192 pel port 80(aquesta adreça hauria de ser showmyip.com). Si la connexió falla fa una petició de DNS per resoldre la IP de showmyip.com i connecta. En els dos casos el client realitza una petició http per que el servei li retorni la IP externa.

2.3.2 Connexió

El procés de connexió comença quan un dels nodes s'hi vol connectar a un altre node de qui coneix l'adreça IP extreta de les seves llistes actualitzades.

Aquest procés s'esdevé d'un intercanvi de missatges entre els dos nodes on primerament, el node 1 envia un paquet de dades que estarà compost per la versió, l'últim número de bloc que hi va enregistrar i la data actual.

Des del node 2, s'hi retorna un *verack* , és a dir, un paquet amb la versió del node 2 i la seva compatibilitat o no amb el node1.

Si el node1 és compatible, retornarà un *verack* per confirmar-ho. A partir d'aquest moment, els dos nodes estan connectats entre si. El número màxim de nodes amb que un node hi pot connectar és 8 mitjançant el client oficial Bitcoin.

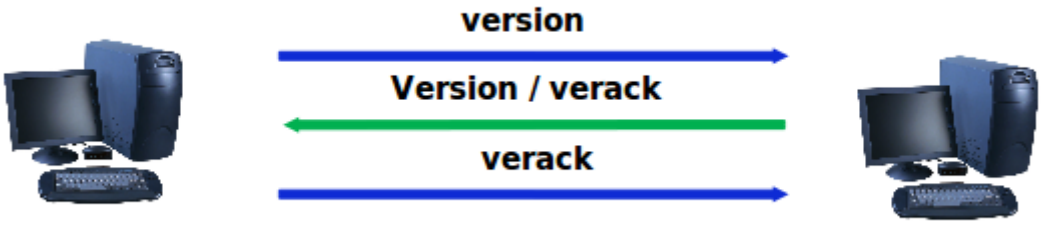

Figura 10

2.3.3 Descàrrega de blocs inicial

De manera inicial, el client bitcoin envia una petició *getblocks*, amb un missatge que conte el hash de l'últim bloc comptabilitzat pel client. Com a resposta es rep un missatge de tipus *inv* amb els 500 blocs que hi ha mes amunt del que hem llistat (a la cadena de blocs).

Amb aquest missatge i enviant un *getdata* cap a la xarxa, descarregarem tots els blocs que ens falten per arribar a completar la cadena de blocs actual.

Si hem de rebre més de 500 blocs, es realitzarien tants *getblocks* com sigui necessari.

2.3.4 Recepció de dades

- Tipus d'informació que s'intercanvia

El protocol de bitcoin contempla dos grans grups de dades que s'intercanvien els nodes dins de la xarxa. Aquestes són els estàndards, les estructures. S'ha de dir però que un tipus d'estructures són els missatges que regulen l'intercanvi de la resta d'estructures i normes que hi entren en joc en el sistema.

Per una banda hi entren en joc les dades normatives com els Hashes, els Merkle Trees , les

signatures, les verificacions de transaccions i les adreces. Aquestes dades són la base de la plataforma P2P que es veuen completades amb les diferents estructures i missatges.

Com a estructures, el sistema contempla els missatges, variables de longitud (cadena i enter), les adreces de xarxa, els vectors d'inventari i les capçaleres de bloc.

- Metodologia d'intercanvi d'informació

La base de l'intercanvi d'informació està en els missatges que els nodes s'envien entre ells, ja sigui a l'hora de connectar per primer cop, ja sigui en el moment d'intercanviar adreces, al realitzar-ne una transferència de bitcoins o simplement a l'actualitzar la cadena de blocs. Els nodes estan constantment enviant-se i retransmeten les dades que processa la xarxa, és a dir, les estructures i normes abans comentades.

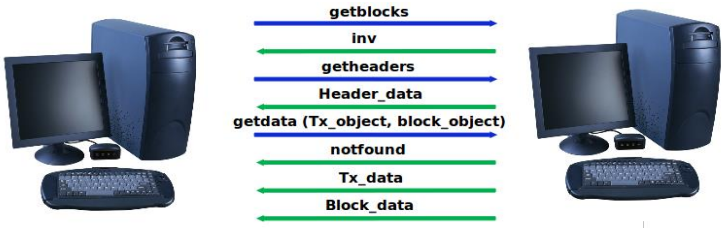

Figura11

2.3.5 Generació de BTC i creació de blocs (Mineria)

La generació de nova moneda Bitcoin, ve determinada per la creació d'un nou bloc. Quan un node que està intentant resoldre el problema criptogràfic troba la solució, es genera automàticament un bloc on la primera transacció és aquella que abona el premi o recompensa BTC a l'adreça del client que ha trobat la solució. Això propicia que molts nodes lluitin per intentar resoldre el problema i generar-ne un bloc amb la seva recompensa associada obtenint un òptim funcionament del sistema.

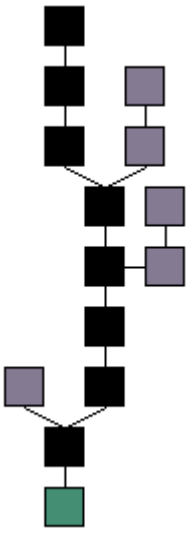

Figura 12

Tot just d'haver creat el bloc, aquest node propaga el nou bloc per tota la xarxa fins que arriba a tots els nodes del sistema.

El procés de mineria dons realitza tasques molt importants a nivell intern del sistema bitcoin

- Manté el correcte funcionament de la xarxa
- Genera nous BTC
- Permet gestionar un sistema de validacions de transaccions.

S'ha de dir però que el número màxim de bitcoins que poden existir a la xarxa és de 21 milions i que hi haurà un moment en el temps que resoldre el repte criptogràfic no aportarà cap recompensa. Això pot plantejar en un futur el cobrament de comissions per la transmissió de les transferències i les creacions de blocs.

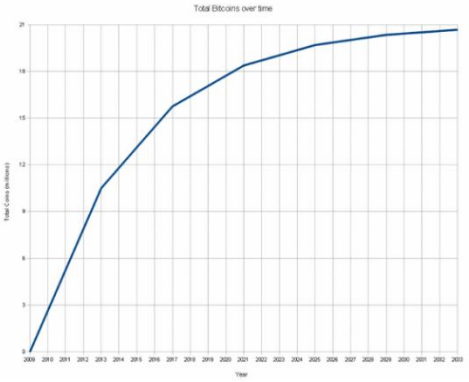

Figura 13

2.3.6 Validació de blocs

Aquest procés ve precedit pel procés de creació de blocs en el que hi juguen un paper molt important els Miners.

2.3.7 Propagació i Confirmació de transaccions

El procés de propagació ve donat de manera intrínseca en la gestió pròpia de la transacció. En el moment que es crea una transacció, es veu necessari un procés que transporti aquesta informació cap a la resta de nodes de la xarxa i aquest procés és el de propagació. Com si d'un missatge es tractés, la propagació de la transacció no cessa fins que no s'ha informat a tots els nodes de la xarxa de tal efecte.

Hem realitzat la unió del procés de confirmació i del procés de propagació perquè un ve acompanyat del altre. La confirmació és aquell procés que permet que el client que gestiona btcoins doni per vàlida una transacció. En aquest procés intervenen un nombre determinat de nodes i ve definit en cada cas per cadascun dels clients de xarxa bitcoin.

En el procés de confirmació, els nodes verifiquen que una transacció existent és vàlida, verifiquen la mateixa mitjançant la signatura i la clau pública de les adreces que hi intervenen.

Per més seguretat, tot i que els nodes afirmen que les transaccions són susceptibles de ser vàlides, caldrà que es generin 6 blocs mitjançant la mineria per que el client oficial bitcoin doni per vàlida i confirmada una transferència de coins.

A la part pràctica podrem observar com minuts després que la transferència de coins s'hagi realitzat, els node destinatari rep confirmació de la transacció. Uns pocs minuts després rebrà la validació al haver rebut 6 blocs validats posteriors a la transacció .

2.3.8 Alive

Aquest procés verifica que els nodes de la xarxa estan aixecats. Si no ha rebut cap missatge en 30 minuts o més, envia un missatge de confirmació per verificar-ne si el node esta up.

D'altre banda si en 90 minuts no ha rebut cap missatge, assumeix que la connexió està tancada.

2.3.9 Intercanvi d'adreces entre nodes

L'intercanvi d'adreces és constant en la xarxa bitcoin, ja sigui mitjançant una resposta *addr* dels nodes cap al node local derivat d'un *getaddr*, ja sigui mitjançant la recepció de missatges *addr* fruit dels nodes que anuncien les seves adreces de manera gratuïta o simplement quan es crea una nova connexió.

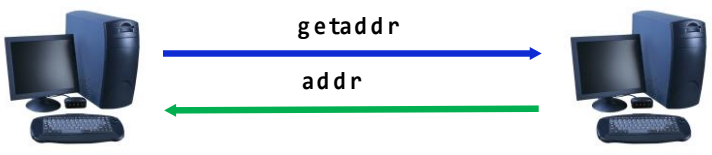

Figura14

Retransmissió d'adreces

Una vegada el node local rep un missatge *addr* i afegeix les adreces a la seva base de dades, aquest retransmet la llista cap a la resta de nodes tot seguint un criteri.

> ***Primer:*** Les adreces tindran un timestamp de 60 minuts respecte el time actual
> ***Segon*****:** El missatge addr contindrà com a màxim 10 adreces. (Si es necessita traspassar més adreces s'enviaran diferents missatges)
> ***Tercer*****:** La variable fGetAddr ha d'estar setejada a fals. És a dir aquesta variable estarà a true quan el node necessiti fer les peticions a la resta de nodes amb els que està connectat, una vegada el node ha rebut com a mínim 1000 adreces, la variable és seteja a fals. (és a dir com a mínim ha de tenir 1000 adreces per poder retransmetre)
> ***Quart*****:**Les adreces han de ser enrutables

Broadcast d'adreces

Cada 24 hores el node local informa de la seva llista d'adreces a la resta de nodes connectats.

Neteja d'adreces

Cada deu minuts s'esborren les adreces antigues de la llista, sempre i quan existeixin com a mínim 3 connexions actives en aquell moment.

A més a més els nodes esborren els missatges que no s'han utilitzat en els últims 14 dies sempre i quan hi hagin com a mínim 1000 adreces conegudes i el procés d'esborrat no duri més de 20 segons.

3. Anàlisi pràctic de connectivitat i transaccions de la xarxa BITCOIN (Proves node a node)

Com a escenari inicial d'aquesta part pràctica, ens trobarem amb dos clients bitcoin separats per la xarxa pública d'internet.

Com tindrem accés als dos clients podrem realitzar les proves i veure les modificacions que hi pertoquin en temps real.

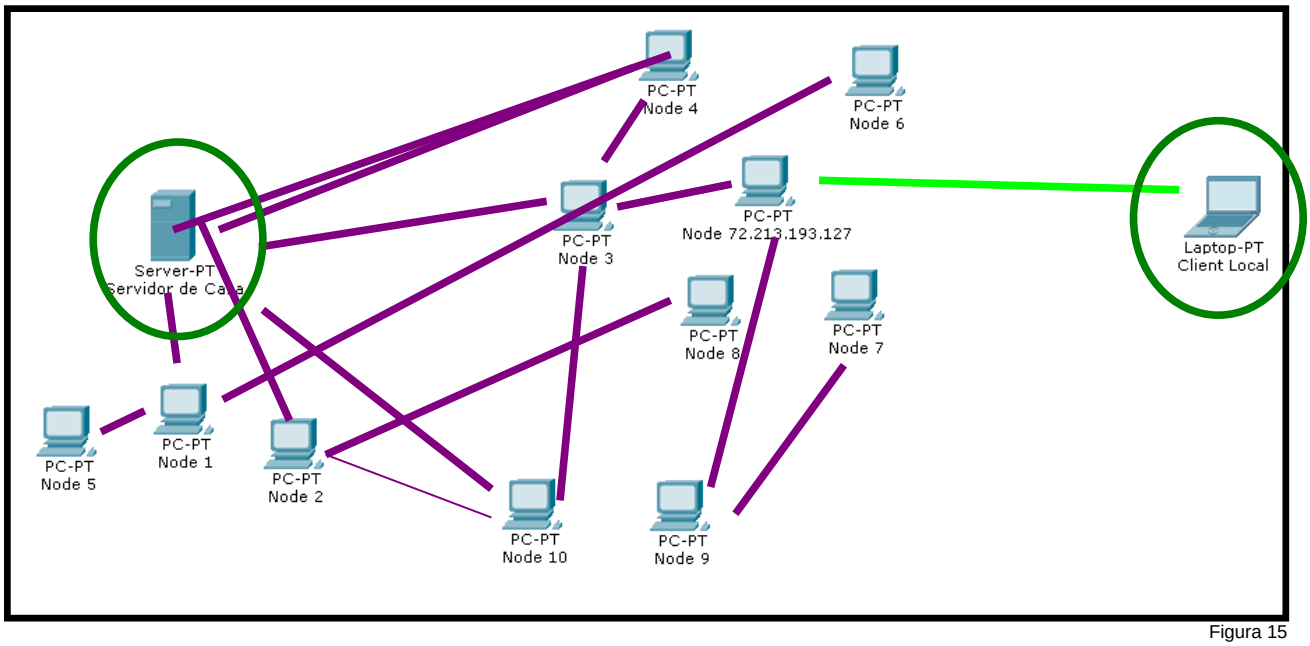

Figura 15

Com a node intermedi per compartir les transaccions, el node client local que serà el propietari dels coins que es volen traspassar, es connectarà únicament a un node escollit dins la xarxa P2P i aquest serà l'encarregat de retransmetre la informació a la resta de nodes.

S'ha realitzat connexió única contra un node perquè ens resultarà més fàcil identificar-ne la informació que es retransmet.

3.1 Cerca de nodes al començament de la carrega inicial

Primer de tot carreguem el bitcoin-qt perquè s'iniciï. Una vegada està iniciat podrem observar que s'ha connectat a la xarxa per algun dels mètodes abans comentats.

Això ho podem veure reflectit al debug log de **Ayuda-->Ventana de depuración-->Abrir Registro de depuración**

Però observem amb més detall el log de depuració:

```
init message: Cargando monedero...
nFileVersion = 80000
 wallet                2489ms
init message: Rescaneando...
Rescanning last 59 blocks (from block 236135)...
 rescan                 487ms
init message: Importando bloques de la base de datos de bloques...
init message: Cargando direcciones...
Loaded 10512 addresses from peers.dat  38ms
mapBlockIndex.size() = 236196
nBestHeight = 236194
setKeyPool.size() = 101
mapWallet.size() = 1
mapAddressBook.size() = 2
send version message: version 70001, blocks=236194, us=0.0.0.0:0, them=0.0.0.0:0, peer=127.0.0.1:0
init message: Generado pero no aceptado
refreshWallet
ThreadOpenConnections started
ThreadMessageHandler started
ThreadDumpAddress started
ThreadSocketHandler started
ThreadOpenAddedConnections started
ThreadIRCSeed started
ThreadIRCSeed exited
ThreadMapPort started
ThreadDNSAddressSeed started
Loading addresses from DNS seeds (could take a while)
ipcThread started
Flushed 10512 addresses to peers.dat  140ms
GetMyExternalIP() received [81.34.240.201] 81.34.240.201:0
GetMyExternalIP() returned 81.34.240.201
```

Figura 16

En aquest tros de log podem veure que carrega la base de dades de blocs tal i on ho va deixar anteriorment, a més carrega les adreces que tenia guardades.

Podem veure també que la nostre versió és la 8.0.0.

Observem que després de carregar-ho tot fa una petició a un servidor DNS per tal de demanar per la seva IP externa.

Recordem des de la versió 0.6.x ja no s'utilitza el IRC per saber l'adreça IP externa que hi té associada el client Bitcoin.

```
Loading addresses from DNS seeds (could take a while)

ipcThread started

Flushed 10512 addresses to peers.dat  140ms

GetMyExternalIP() received [81.34.240.201] 81.34.240.201:0

GetMyExternalIP() returned 81.34.240.201
```

Figura 17

La nostra IP externa en aquest cas és 81.34.240.201. Però busquem la transacció inicial contra el servidor DNS que li retorna la IP.

Després de revisar amb el wireshark la connexió, veiem que ha fet una petició a un servidor de dyndns demanant per la ip externa.

Aquí tenim la captura de dades que retorna el servidor de dyndns cap a la nostra ip interna pel port 80 http.

```
0000  00 16 ea 56 e1 7c 00 01  38 fd 3a fb 08 00 45 00   ...V.|.. 8.:...E.

0010  01 38 89 fb 40 00 35 06  86 ee 5b c6 16 46 c0 a8   .8..@.5. ..[..F..

0020  01 22 00 50 8e 57 9c c5  a3 d7 f9 5d 67 c5 80 18   ."·P·W·· ···]g···

0030  20 58 98 de 00 00 01 01  08 0a 0e 69 56 f9 00 11    X...... ...iV...

0040  4e 7a 48 54 54 50 2f 31  2e 31 20 32 30 30 20 4f   NzHTTP/1 .1 200 O

0050  4b 0d 0a 43 6f 6e 74 65  6e 74 2d 54 79 70 65 3a   K··Conte nt-Type:

0060  20 74 65 78 74 2f 68 74  6d 6c 0d 0a 53 65 72 76    text/ht ml··Serv

0070  65 72 3a 20 44 79 6e 44  4e 53 2d 43 68 65 63 6b   er: DynD NS-Check

0080  49 50 2f 31 2e 30 0d 0a  43 6f 6e 6e 65 63 74 69   IP/1.0·· Connecti

0090  6f 6e 3a 20 63 6c 6f 73  65 0d 0a 43 61 63 68 65   on: clos e··Cache

00a0  2d 43 6f 6e 74 72 6f 6c  3a 20 6e 6f 2d 63 61 63   -Control : no-cac

00b0  68 65 0d 0a 50 72 61 67  6d 61 3a 20 6e 6f 2d 63   he··Prag ma: no-c

00c0  61 63 68 65 0d 0a 43 6f  6e 74 65 6e 74 2d 4c 65   ache··Co ntent-Le

00d0  6e 67 74 68 3a 20 31 30  35 0d 0a 0d 0a 3c 68 74   ngth: 10 5····<ht

00e0  6d 6c 3e 3c 68 65 61 64  3e 3c 74 69 74 6c 65 3e   ml><head ><title>

00f0  43 75 72 72 65 6e 74 20  49 50 20 43 68 65 63 6b   Current  IP Check

0100  3c 2f 74 69 74 6c 65 3e  3c 2f 68 65 61 64 3e 3c   </title> </head><

0110  62 6f 64 79 3e 43 75 72  72 65 6e 74 20 49 50 20   body>Cur rent IP

0120  41 64 64 72 65 73 73 3a  20 38 31 2e 33 34 2e 32   Address:  81.34.2

0130  34 30 2e 32 30 31 3c 2f  62 6f 64 79 3e 3c 2f 68   40.201</ body></h

0140  74 6d 6c 3e 0d 0a                                 tml>··
```

Figura 18

Una vegada sap la IP externa que té, continua el procés de descobriment d'adreces que hem descrit a l'inici . Comença a connectar amb nodes i a balancejar la seva llista d'adreces de nodes actius.

En la següent figura podem observar un tros de log del debug on realitza una connexió contra un node que està actiu i es veu un intercanvi de versions.

```
trying connection 24.21.5.237:8333 lastseen=5,5hrs

stored orphan tx 2042405c3a (mapsz 213)

connection timeout

Added 1 addresses from 95.79.185.198: 21 tried, 12173 new

trying connection 72.213.193.127:8333 lastseen=5,5hrs

connected 72.213.193.127:8333

send version message: version 70001, blocks=236198, us=81.34.240.201:8333, them=72.213.193.127:8333, peer=72.213.193.127:8333

Added time data, samples 10, offset +0 (+0 minutes)

Moving 72.213.193.127:8333 to tried

receive version message: version 70001, blocks=238383, us=81.34.240.201:52142, them=72.213.193.127:8333,
peer=72.213.193.127:8333

stored orphan tx 92b26539d9 (mapsz 214)

Added 1 addresses from 95.79.185.198: 22 tried, 12172 new
```

Figura 19

Veiem la transacció en una captura de wireshark

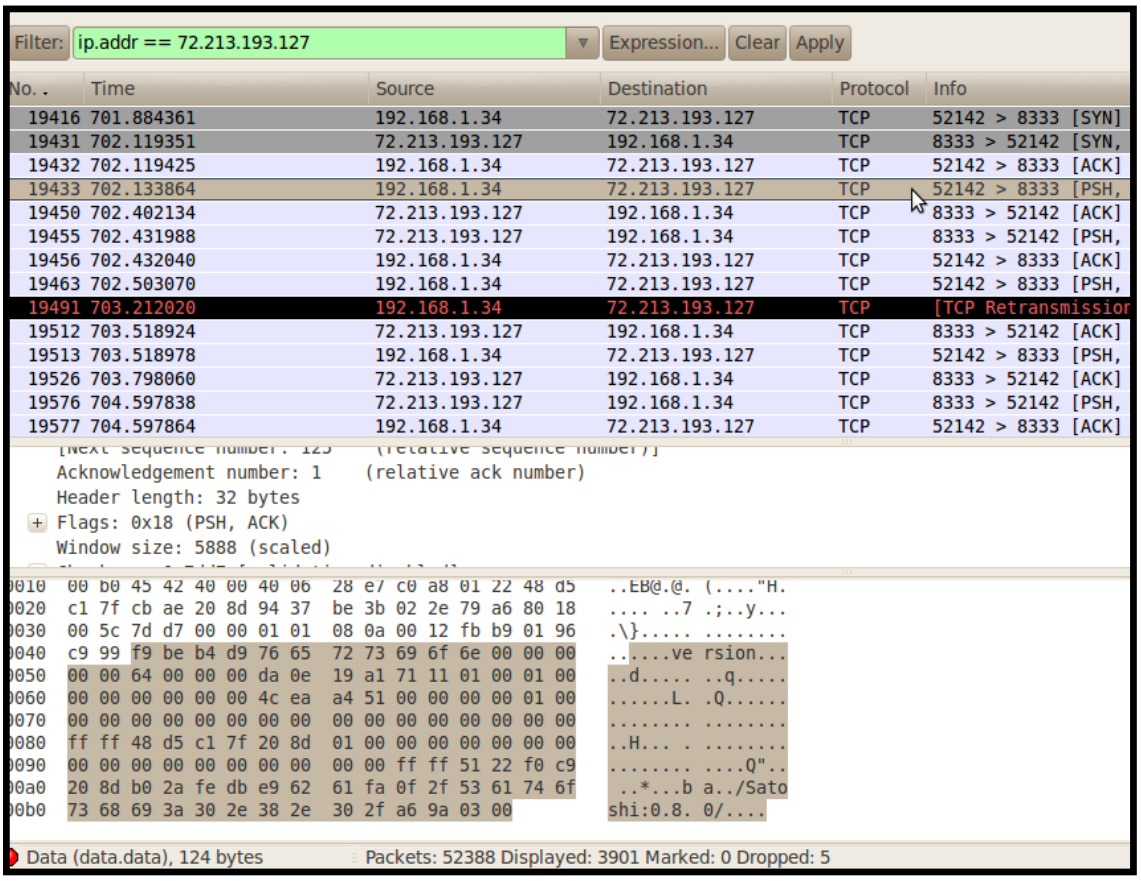

Figura 20

Si ens fixem hem filtrat per la IP del node amb el que s'ha intercanviat la versió.

El meu client li envia la versió 0.8.0 i en la següent línia (19455) el node remot ens retorna la seva versió 0.8.1 i un *verack* confirmant la compatibilitat amb la versió del nostre client local.

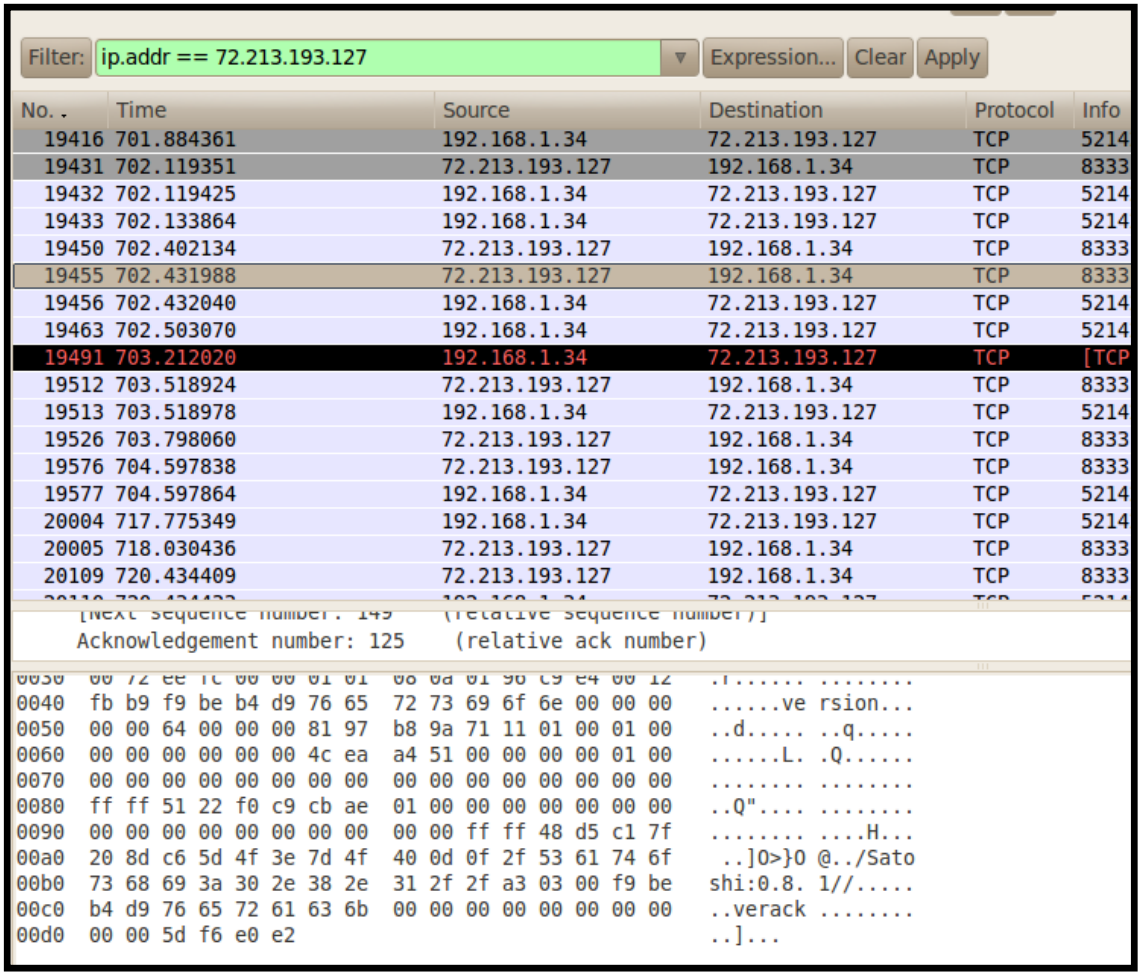

Figura 21

A continuació, connectarem el client bitcoin únicament a un dels nodes, més exactament al node 72.213.193.127 i mitjançant el wireshark farem un anàlisi exhaustiu de les transaccions d'informació entre el node local i el node remot.

Mitjançant el fitxer de debug i l'anàlisi amb wireshark anirem corroborant els missatges que anteriorment hem definit i el seu procés d'intercanvi.

3.2 Connectivitat amb un node escollit (Node 72.213.193.127)

Després d'indicar-li al client que es connecti directament a un dels nodes mitjançant la comanda –connect, obtenim diferents frames referents a l'intercanvi de versions i el *ack* final del node local.

```
No.     Time            Source                Destination           Protocol Info

     60 19:57:08.483742 192.168.1.34          72.213.193.127        TCP      39609 > 8333 [PSH, ACK] Seq=1 Ack=1 Win=5888
Len=124 TSV=10176538 TSER=70221440


Frame 60 (190 bytes on wire, 190 bytes captured)

Ethernet II, Src: Intel_56:e1:7c (00:16:ea:56:e1:7c), Dst: XaviTech_fd:3a:fb (00:01:38:fd:3a:fb)

Internet Protocol, Src: 192.168.1.34 (192.168.1.34), Dst: 72.213.193.127 (72.213.193.127)

Transmission Control Protocol, Src Port: 39609 (39609), Dst Port: 8333 (8333), Seq: 1, Ack: 1, Len: 124

Data (124 bytes)


0000  f9 be b4 d9 76 65 72 73 69 6f 6e 00 00 00 00 00   ....version.....

0010  64 00 00 00 cc b9 64 0d 71 11 01 00 01 00 00 00   d.....d.q.......

0020  00 00 00 00 f4 92 a7 51 00 00 00 00 01 00 00 00   .......Q........

0030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 ff ff   ................

0040  48 d5 c1 7f 20 8d 00 00 00 00 00 00 00 00 00 00   H... ............

0050  00 00 00 00 00 00 00 00 ff ff 00 00 00 00 00 00   ................

0060  71 f9 ac f9 be ae 8e a6 0f 2f 53 61 74 6f 73 68   q......../Satosh

0070  69 3a 30 2e 38 2e 30 2f 84 a4 03 00               i:0.8.0/....

    Data: F9BEB4D976657273696F6E0000000000064000000CCB9640D...

    [Length: 124]
```

Figura 22

Observem que la versió del nostre client és la 0.8.0.
Ara veiem la resposta del node remot just 2 frames més a sota.

```
Frame 62 (214 bytes on wire, 214 bytes captured)

Ethernet II, Src: XaviTech_fd:3a:fb (00:01:38:fd:3a:fb), Dst: Intel_56:e1:7c (00:16:ea:56:e1:7c)

Internet Protocol, Src: 72.213.193.127 (72.213.193.127), Dst: 192.168.1.34 (192.168.1.34)

Transmission Control Protocol, Src Port: 8333 (8333), Dst Port: 39609 (39609), Seq: 1, Ack: 125, Len: 148

Data (148 bytes)


0000  f9 be b4 d9 76 65 72 73 69 6f 6e 00 00 00 00 00   ....version.....

0010  64 00 00 00 35 79 cc b3 71 11 01 00 01 00 00 00   d...5y..q.......

0020  00 00 00 00 f3 92 a7 51 00 00 00 00 01 00 00 00   .......Q........

0030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 ff ff   ................

0040  51 22 f0 c9 9a b9 01 00 00 00 00 00 00 00 00 00   Q"..............

0050  00 00 00 00 00 00 00 00 ff ff 48 d5 c1 7f 20 8d   ..........H... .

0060  5d d1 eb d0 20 e0 a0 40 0f 2f 53 61 74 6f 73 68   ]... ..@./Satosh

0070  69 3a 30 2e 38 2e 31 2f 84 a4 03 00 f9 be b4 d9   i:0.8.1/........

0080  76 65 72 61 63 6b 00 00 00 00 00 00 00 00 00 00   verack..........

0090  5d f6 e0 e2                                       ]...

    Data: F9BEB4D976657273696F6E000000000006400000003579CCB3...

    [Length: 148]
```

Figura 23

Observem que la versió del node remot és la 0.8.1.

A continuació el node local ha de respondre al node client si és o no compatible i a partir de

l'acceptació per part del node local, començarà la compartició d'informació de xarxa (adreces, transaccions, blocs..etc).

```
No.     Time            Source                Destination           Protocol Info

     64 19:57:08.806167 192.168.1.34          72.213.193.127        TCP      39609 > 8333 [PSH, ACK] Seq=125 Ack=149 Win=6912
Len=1037 TSV=10176619 TSER=70221524


Frame 64 (1103 bytes on wire, 1103 bytes captured)

Ethernet II, Src: Intel_56:e1:7c (00:16:ea:56:e1:7c), Dst: XaviTech_fd:3a:fb (00:01:38:fd:3a:fb)

Internet Protocol, Src: 192.168.1.34 (192.168.1.34), Dst: 72.213.193.127 (72.213.193.127)

Transmission Control Protocol, Src Port: 39609 (39609), Dst Port: 8333 (8333), Seq: 125, Ack: 149, Len: 1037

Data (1037 bytes)


0000  f9 be b4 d9 76 65 72 61 63 6b 00 00 00 00 00 00   ....verack......

0010  00 00 00 00 5d f6 e0 e2 f9 be b4 d9 67 65 74 61   ....]........geta

0020  64 64 72 00 00 00 00 00 00 00 00 00 5d f6 e0 e2   ddr.........]...

0030  f9 be b4 d9 67 65 74 62 6c 6f 63 6b 73 00 00 00   ....getblocks...

0040  c5 03 00 00 e1 6f ea 9a 71 11 01 00 1d 40 65 25   .....o..q....@e%

0050  45 83 f1 51 23 75 1d 06 27 92 11 03 e8 e0 7e 5d   E..Q#u..'.....~]

0060  7a 6e fe fc 90 57 01 00 00 00 00 00 00 c3 53 55   zn...W........SU

0070  24 6f d4 27 bc 0d 54 1a cc 23 e8 e2 26 57 7d 85   $o.'..T..#..&W}.

0080  9a 51 8a 79 7a 17 01 00 00 00 00 00 00 f0 c2 82   .Q.yz...........

0090  4e 71 a2 0e 04 ba 03 43 a7 d9 94 82 ad 71 4b 8f   Nq.....C.....qK.

00a0  97 b2 fc 87 82 23 00 00 00 00 00 00 00 56 c4 5a   .....#.......V.Z

00b0  8d 5d 0e 8d a6 e9 a6 0d c1 05 7a 7a 2c ea 1f 25   .]........zz,..%...................
```

Figura 24

Com veiem, no només ha respost amb acceptació de connexió al node remot sinó que li demana mitjançant un *getaddr* una actualització de les adreces que conté i amb el *getblocks* un paquet dels últims blocs que coneix el node remot. D'aquesta manera podrà esbrinar quants li resten i començarà d'aquesta manera a demanar *tx* i *headers* mitjançant les comandes corresponents.

Com hem vist a la teoria el resultat d'una petició *getblocks* és un inventari d'objectes de la xarxa. <inv>.
Efectivament, després del frame anterior ve el següent indicant un inventari d'objectes.

```
No.     Time            Source                Destination           Protocol Info

     66 19:57:10.093810 72.213.193.127        192.168.1.34          TCP      8333 > 39609 [PSH, ACK] Seq=149 Ack=1162 Win=16640
Len=61 TSV=70221854 TSER=10176619



Frame 66 (127 bytes on wire, 127 bytes captured)

Ethernet II, Src: XaviTech_fd:3a:fb (00:01:38:fd:3a:fb), Dst: Intel_56:e1:7c (00:16:ea:56:e1:7c)

Internet Protocol, Src: 72.213.193.127 (72.213.193.127), Dst: 192.168.1.34 (192.168.1.34)

Transmission Control Protocol, Src Port: 8333 (8333), Dst Port: 39609 (39609), Seq: 149, Ack: 1162, Len: 61

Data (61 bytes)


0000  f9 be b4 d9 69 6e 76 00 00 00 00 00 00 00 00 00   ....inv.........

0010  25 00 00 00 d4 1f 94 67 01 01 00 00 00 32 d0 8e   %......g.....2..

0020  7a 95 e1 2d 3d 67 fd 6f 28 49 ec 99 6f 75 94 59   z..-=g.o(I..ou.Y

0030  3d 88 98 00 8f 63 8d 0b 1b 75 23 68 55            =....c...u#hU

    Data: F9BEB4D9696E76000000000000000000025000000D41F9467...

    [Length: 61]
```

Figura 25

Una vegada el nostre node local rep la informació de l'inventari realitza un *getdata* per començar a rebre les dades transaccionals.

```
No.     Time            Source                Destination           Protocol Info

     68 19:57:10.164331 192.168.1.34          72.213.193.127        TCP      39609 > 8333 [PSH, ACK] Seq=1162 Ack=210 Win=6912
Len=61 TSV=10176958 TSER=70221854


Frame 68 (127 bytes on wire, 127 bytes captured)

Ethernet II, Src: Intel_56:e1:7c (00:16:ea:56:e1:7c), Dst: XaviTech_fd:3a:fb (00:01:38:fd:3a:fb)

Internet Protocol, Src: 192.168.1.34 (192.168.1.34), Dst: 72.213.193.127 (72.213.193.127)

Transmission Control Protocol, Src Port: 39609 (39609), Dst Port: 8333 (8333), Seq: 1162, Ack: 210, Len: 61

Data (61 bytes)


0000  f9 be b4 d9 67 65 74 64 61 74 61 00 00 00 00 00   ....getdata.....

0010  25 00 00 00 d4 1f 94 67 01 01 00 00 00 32 d0 8e   %......g.....2..

0020  7a 95 e1 2d 3d 67 fd 6f 28 49 ec 99 6f 75 94 59   z..-=g.o(I..ou.Y

0030  3d 88 98 00 8f 63 8d 0b 1b 75 23 68 55            =....c...u#hU

    Data: F9BEB4D96765746461746100000000025000000D41F9467...

    [Length: 61]
```

Figura 26

Per entendre bé el funcionament hem de fixar-nos sempre en l'origen i la destinació.
En aquest moment el nostre node local comença a rebre dades referents a les transaccions i els blocs i entre els diferents missatges també i trobem missatges *addr*.

(Recordem que en captures anteriors hem observat que el node local enviava un *getaddr* al node remot)

Observem les dades de transacció

```
Frame 130 (527 bytes on wire, 527 bytes captured)

Ethernet II, Src: XaviTech_fd:3a:fb (00:01:38:fd:3a:fb), Dst: Intel_56:e1:7c (00:16:ea:56:e1:7c)

Internet Protocol, Src: 72.213.193.127 (72.213.193.127), Dst: 192.168.1.34 (192.168.1.34)

Transmission Control Protocol, Src Port: 8333 (8333), Dst Port: 39609 (39609), Seq: 520, Ack: 1284, Len: 461

Data (461 bytes)


0000  f9 be b4 d9 74 78 00 00 00 00 00 00 00 00 00 00   ....tx..........

0010  b5 01 00 00 06 10 ac 93 01 00 00 00 02 5e d8 ae   .............^..

0020  3a b4 77 fb f8 61 57 8a 18 97 36 59 ad 30 e4 16   :.w..aW...6Y.0..

0030  c7 d0 1d 0f 93 64 51 bc e6 d9 62 7a 83 00 00 00   .....dQ...bz....

0040  00 8a 47 30 44 02 20 3f 61 97 01 43 4a 07 2e f3   ..G0D. ?a..CJ...

0050  82 dc 75 22 be fa c6 5a 9b 43 81 73 bf d6 75 ab   ..u"...Z.C.s..u.

0060  52 ee 7b 55 5e 07 e7 02 20 18 a3 11 53 b6 b6 76   R.{U^... ...S..v

0070  03 38 c6 fc 0a 6f b3 b3 98 dc a6 56 7f 51 5d 9e   .8...o.....V.Q].

0080  2b c9 8e 1a cf 48 c2 7c fb 01 41 04 7c fc 6e 80   +....H.|..A.|.n.

0090  1a 25 c1 c5 2b a2 e0 08 59 14 2e 9f f0 3b ea 35   .%..+...Y....;.5

00a0  67 0e 9f 94 be 22 b8 db e4 ed ae 2a 40 46 fa 9c   g...."......*@F..

00b0  86 d2 d4 c2 af 64 02 ae 3a 98 d7 25 ec 1a 30 a4   .....d..:..%..0.
```

```
00c0  61 01 3a 96 82 94 04 3e a3 3d 3b 79 ff ff ff ff   a.:....>.=;y....
00d0  b8 b3 cf 89 1a 63 c0 bd 79 bf 56 15 b3 4e d9 ae   .....c..y.V..N..
00e0  c7 11 09 8d 06 6e 7e 82 25 16 1e a5 6d f7 43 b5   .....n~.%...m.C.
00f0  01 00 00 00 8b 48 30 45 02 21 00 cc 71 8c 07 d6   .....H0E.!..q...
0100  39 6e f2 a2 1e fc a7 02 7f 30 98 d9 b5 92 d8 8e   9n.......0......
0110  1e 3d ea ab 34 cf 93 61 a4 fa eb 02 20 59 cd 58   .=..4..a.... Y.X
0120  46 6e 52 84 f2 b3 2e 1f 2f c5 6b 08 17 37 c4 b5   FnR...../.k..7..
0130  9b 29 54 a8 c1 e8 f4 a0 9d d7 50 f4 b6 01 41 04   .)T.......P...A.
0140  63 68 60 3c bd b7 b8 6d cf 72 17 39 1c b3 e8 c6   ch`<...m.r.9....
0150  d2 d0 15 42 d9 87 2a f5 f3 f1 c7 d7 50 d6 07 4f   ...B..*.....P..O
0160  ab a4 36 0a 6e cb 21 b9 d4 67 bf cd 85 01 3a 7a   ..6.n.!..g....:z
0170  db 01 68 52 5d 33 f0 57 bb c0 25 8c db 11 08 5f   ..hR]3.W..%...._
0180  ff ff ff ff 02 cc 57 10 00 00 00 00 00 19 76 a9   ......W.......v.
0190  14 c6 92 e4 e1 8b 80 29 b6 7d f8 c7 9f dc 2e e8   .......).}......
01a0  d1 a9 00 e3 46 88 ac 81 57 de 04 00 00 00 00 19   ....F...W.......
01b0  76 a9 14 f8 79 24 03 d2 1c a8 ed 44 20 13 61 e2   v...y$.....D .a.
01c0  99 6e 30 55 b9 62 2c 88 ac 00 00 00 00            .n0U.b,......
    Data: F9BEB4D9747800000000000000000000B50100000610AC93...
    [Length: 461]
```

Figura 27

I el frame *addr*

```
Frame 136 (1506 bytes on wire, 1506 bytes captured)
Ethernet II, Src: XaviTech_fd:3a:fb (00:01:38:fd:3a:fb), Dst: Intel_56:e1:7c (00:16:ea:56:e1:7c)
Internet Protocol, Src: 72.213.193.127 (72.213.193.127), Dst: 192.168.1.34 (192.168.1.34)
Transmission Control Protocol, Src Port: 8333 (8333), Dst Port: 39609 (39609), Seq: 981, Ack: 1284, Len: 1440
Data (1440 bytes)

0000  f9 be b4 d9 61 64 64 72 00 00 00 00 00 00 00 00   ....addr........
0010  33 75 00 00 14 6c 05 b2 fd e8 03 c8 53 a7 51 01   3u...l......S.Q.
0020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0030  00 ff ff ca 70 80 86 20 8d b9 34 9e 51 01 00 00   ....p.. ..4.Q...
0040  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ff   ................
0050  ff 45 8f 0e d4 20 8d da f3 a6 51 01 00 00 00 00   .E... ....Q.....
0060  00 00 00 00 00 00 00 00 00 00 00 00 00 ff ff de   ................
0070  62 f0 4c 20 8d 9b e3 9c 51 01 00 00 00 00 00 00   b.L ....Q.......
0080  00 00 00 00 00 00 00 00 00 00 00 00 ff ff ad 16 21   ...............!
0090  f4 20 8d fb 6f a7 51 01 00 00 00 00 00 00 00 00   . ..o.Q.........
00a0  00 00 00 00 00 00 00 00 00 ff ff 57 98 1c 45 20   ...........W..E
00b0  8d bd 5f a7 51 01 00 00 00 00 00 00 00 00 00 00   .._.Q...........
00c0  00 00 00 00 00 00 00 ff ff 53 4d 86 f7 20 8d 9e   .........SM.. ..
```

Figura 28

3.3 Realització d'una transacció de 1000 microcoins cap a una adreça d'un altre client.

Per la realització d'aquesta prova s'ha utilitzat l'escenari anterior en el que disposem de dos ordinadors separats per la xarxa d'internet.

Un ordinador porta la bitlletera amb els coins de prova que s'han establert en aquest projecte. A l'altre no hi ha cap coin a la seva bitlletera.

El procediment de la prova contemplarà un intercanvi de 1000 microcoins entre una i l'altre adreça. Així mateix i del fet de realitzar la transacció, se'n derivaran unes despeses de 500 microcoins com a donació de transport de les transaccions.

Escenari: transferència de 1000 microbtc

Adreça origen: 124VWCdbaDCF1kQogj6BgRaHpr8joW9TGQ
Adreça destí: 17zgix942ZDEiqxT4ZGjhjyRExmChgxmY3

Transacció:

Figura 29

En el registre de transaccions se'ns indica l'hora en que vam realitzar la transferència que ens pot guiar per localitzar la mateixa dins del dump de wireshark.

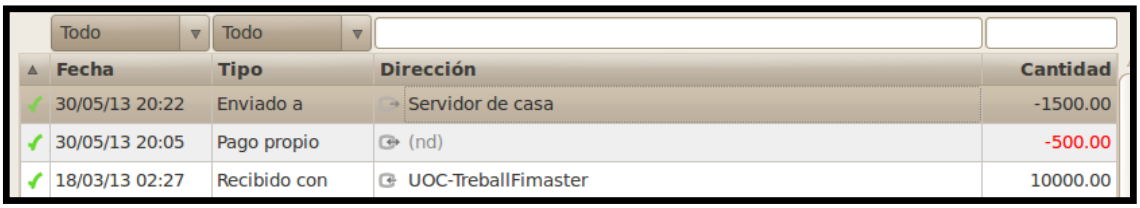

Figura 30

Ara cercarem la transacció en el dump de wireshark i en el log de debug del client bitcoin. Amb això podrem analitzar la transacció més profundament.

Tot i això, com només hem realitzat una transacció de 1000 microcoins cap a fora, bastarà en cercar una "tx" en que l'adreça d'origen sigui la nostra adreça local (192.168.1.34) i l'adreça destí sigui (72.213.193.127) .

És al número de frame 11250 on es troba la transacció.
Anem a analitzar-la.

```
No.     Time            Source                Destination           Protocol Info

  11250 20:22:41.821957 192.168.1.34          72.213.193.127        TCP      39609 > 8333 [PSH, ACK] Seq=46767 Ack=1238299
Win=92608 Len=251 TSV=10559873 TSER=70604693

Frame 11250 (317 bytes on wire, 317 bytes captured)

Ethernet II, Src: Intel_56:e1:7c (00:16:ea:56:e1:7c), Dst: XaviTech_fd:3a:fb (00:01:38:fd:3a:fb)

Internet Protocol, Src: 192.168.1.34 (192.168.1.34), Dst: 72.213.193.127 (72.213.193.127)

Transmission Control Protocol, Src Port: 39609 (39609), Dst Port: 8333 (8333), Seq: 46767, Ack: 1238299, Len: 251

Data (251 bytes)

0000  f9 be b4 d9 74 78 00 00 00 00 00 00 00 00 00   ....tx.........

0010  e3 00 00 00 61 b9 6f a4 01 00 00 00 01 b1 d9 28   ....a.o........(

0020  c8 f5 f2 d8 77 89 58 82 38 d8 7e ee 95 b1 9a f7   ....w.X.8.~.....

0030  a6 ef bb 00 92 72 22 5c 0f a3 5a 06 8a 00 00 00   .....r"\..Z.....

0040  00 6c 49 30 46 02 21 00 88 7c 51 82 82 16 c8 c4   .lI0F.!..|Q.....

0050  0c 98 b0 01 2e 06 b0 63 36 43 00 94 23 27 74 07   .......c6C..#'t.

0060  75 fd 56 25 48 37 5d 6a 02 21 00 a0 f2 1d d4 e8   u.V%H7]j.!......

0070  bf 98 77 f2 15 eb 00 1f 4b 1e 18 ce b1 df 2e 30   ..w.....K......0

0080  62 e7 db 38 6c 3b 95 cf 5f bc 6e 01 21 02 79 c6   b..8l;.._.n.!.y.

0090  83 b2 96 15 c1 7c 5b 43 29 57 21 41 19 70 4e 0f   .....|[C)W!A.pN.

00a0  b4 03 9f a4 cf 61 ee c7 39 89 6c 54 3f 6d ff ff   .....a..9.lT?m..

00b0  ff ff 02 2c 04 0c 00 00 00 00 19 76 a9 14 18   ...,.......v...

00c0  a9 13 94 f8 53 cd 38 ac 47 0a 22 07 f1 29 2c 7d   ....S.8.G."..),}

00d0  f2 2c b0 88 ac a0 86 01 00 00 00 00 19 76 a9   .,............v.

00e0  14 4c b9 9f b4 f7 8f 0e 47 d2 00 aa d5 fe 00 ff   .L......G.......

00f0  cc 8e ec 14 cc 88 ac 00 00 00 00                 ...........

    Data: F9BEB4D9747800000000000000000000E300000061B96FA4...

    [Length: 251]
```

Figura 31

Capçalera del missatge:

```
f9 be b4 d9      --> Xarxa Principal
74 78 00 00 00 00 00 00 00 00 00 00   --> Comanda TX
e3 00 00 00      -->  Payload de 227 bytes
61 b9 6f a4      -->  Checksum del payload
01 00 00 00      --> Versió de la transacció
Entrades:
01       --> Número d'entrades

b1 d9 28 c8 f5 f2 d8 77   89 58 82 38 d8 7e ee 95
b1 9a f7 a6 ef bb 00 92      72 22 5c 0f a3 5a 06 8a   -->Punt de Sortida o sortida anterior.
00 00 00 00
6c       -->El script és de 108 bytes de longitud
49 30 46 02 21 00 88 7c   51 82 82 16 c8 c4 0c 98
b0 01 2e 06 b0 63 36 43   00 94 23 27 74 07 75 fd
56 25 48 37 5d 6a 02 21   00 a0 f2 1d d4 e8 bf 98   -->      ScriptSig (Signatura)
77 f2 15 eb 00 1f 4b 1e     18 ce b1 df 2e 30 62 e7
db 38 6c 3b 95 cf 5f bc       6e 01 21 02 79 c6 83 b2
96 15 c1 7c 5b 43 29 57    21 41 19 70 4e 0f b4 03
9f a4 cf 61 ee c7 39 89       6c 54 3f 6d

ff ff  ff ff     --> Seqüencia
Sortides:
02       --> 2 transaccions de sortida

2c 04 0c 00 00 00 00 00     --> Sortida1: Coins totals després de la transacció (7875 micro)
19
76 a9 14 18 a9 13 94 f8 53 cd 38 ac 47 0a 22 07 f1 29 2c 7d
 f2 2c b0 88 ac
a0 86 01 00 00 00 00 00   -->Sortida 2: coins transferits a l'altre adreça(1000 micro)
19
76 a9 14 4c b9 9f b4 f7 8f 0e 47 d2 00 aa d5 fe 00 ff
 cc 8e ec 14 cc 88 ac
00 00 00 00
```

Figura 32

3.4 Verificació temporal de la transacció

Per verificar el temps de transacció, vaig agafar dos captures, una en el moment de la transacció i una altra en el moment que hi va arribar-hi al servidor de casa la confirmació i validació de la transferència.

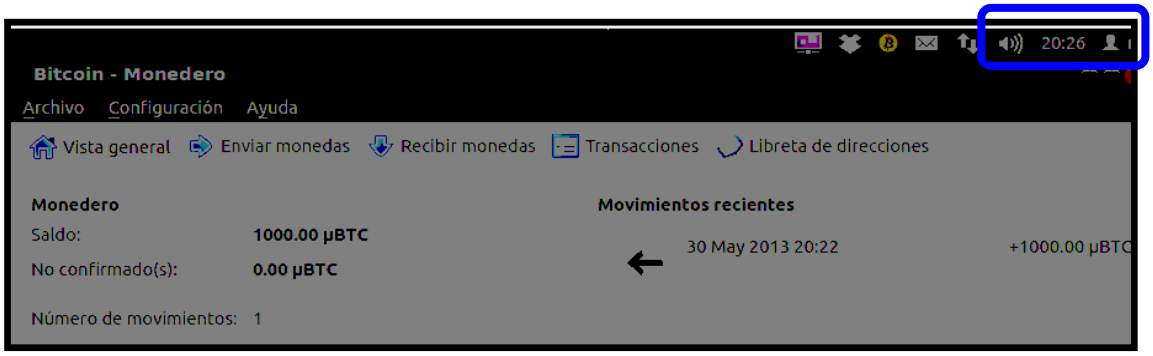

Figura 33

El temps total de transacció (fins que es va rebre el saldo confirmat al servidor de casa) van ser 5 minuts. S'ha de dir però que això depèn en certa mesura de la quantitat de nodes als que estàs connectat. En el client local no va arribar la confirmació fins a 40 minuts després.(Recordem que el node local estava connectat només a un node remot).

D'això podem extreure que quan més nodes estiguis connectat (fins a 8) més ràpidament rebràs les actualitzacions i confirmacions dels blocs que es van inserint a la cadena de blocs.

3.5 Verificació de la transacció a la cadena de blocs

Per verificar la transacció visitarem la plana web blockchain.info/es que mostra informació online de la cadena de blocs actualitzada.

Veurem que podrem fer cerques amb l'adreça com a patró.

Cerquem per l'adreça del node destinatari dels coins.

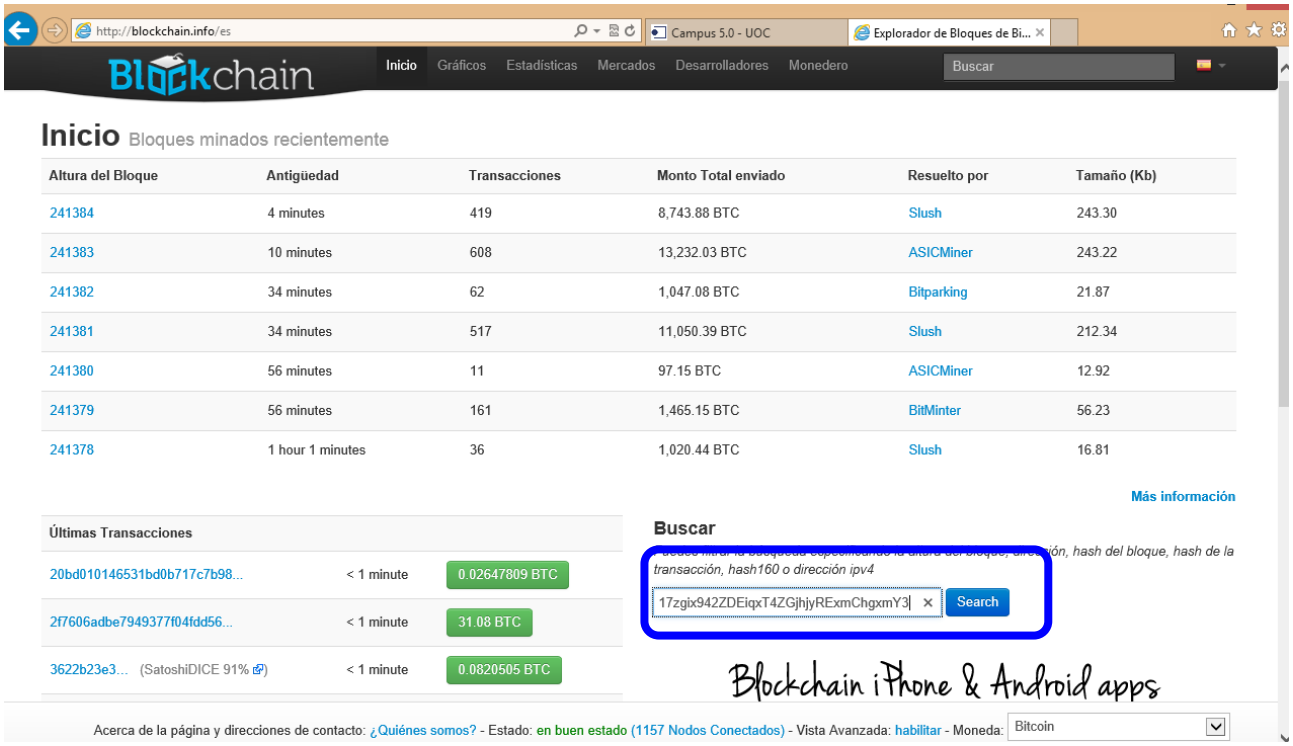

Figura 34

Observem la transacció
Trobem informació d'origen de la transacció inicial i com a destí l'adreça destí que hem introduït.

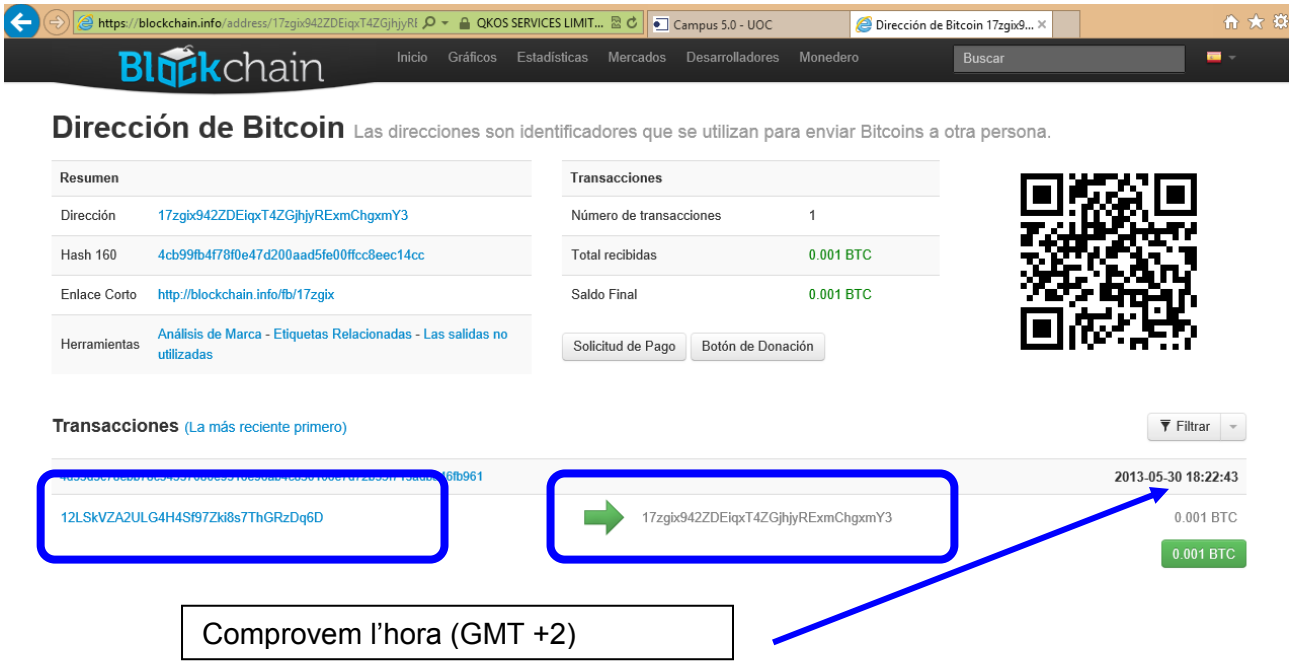

Figura 35

4. Conclusions i línies de treball futur

Amb aquest petit recull d'informació en el que hem pogut observar com estan definits els elements principals de la xarxa bitcoin, com es desenvolupa la negociació de missatges per intercanviar informació, quins són els elements de seguretat de que disposa el sistema o el procés de transacció de bitcoins, podem extreure unes petites conclusions que expressarem a continuació.

El funcionament del sistema aporta característiques de seguretat de la informació que són equiparables a les aportades per les entitats bancaries a l'actualitat.

El sistema ens proporciona confidencialitat, mitjançant l'intercanvi de moneda entre adreces que no corresponen a cap nom , dni ni a cap persona en concret. És similar a l'intercanvi monetari entre persones físiques on aquestes no han de revelar la seva identitat per intercanviar moneda.

També ens proporciona un sistema de certificació, basada en clau pública i privada, que permet aportar veracitat i no repudi a les transaccions que es realitzen entre les adreces. A més si recordem hem parlat de la cadena de blocs on es van escrivint totes les transaccions de manera històrica, des de l'inici del sistema i això permet identificar els elements, les adreces i transaccions implicades en cadascuna de les transaccions del sistema.

Referent a la característica de disponibilitat cal fer èmfasi en la propietat de xarxa P2P del sistema. Aquesta estructura distribuïda de la informació, permet obtenir la màxima disponibilitat possible del servei ja que hi ha molts coneixedors a l'hora de les dades actualitzades del sistema.

D'altra banda i com a punt negatiu sobre la disponibilitat, si féssim un anàlisi de riscos exhaustiu, veuríem que el sistema comporta una dependència molt elevada del treball que realitzen els miners. Informacions descrites anteriorment ens indiquen que a la llarga els beneficis de la mineria apunten a ser mínims i sorgirà la problemàtica del pagament de comissions per mineria. Això podria ser un inconvenient per els usuaris del bitcoin.

Referent a la integritat de la informació, hem de fer recordatori de la part pràctica d'aquest treball on hem revisat la informació de la transacció a nivell baix. Hem pogut verificar que existeixen camps de la informació que s'envien els nodes, que corresponen a les comprovacions de possibles errors entre les dades que es reben i s'envien. Aquests errors que poden ser normals per errors físics de comunicació, també poden ser objecte d'intents de cometre frau sobre les transaccions.

Això, ja s'ha vist que no és viable perquè les transaccions futures es generen de transaccions passades, fent no possible modificar els camps de valor de coins en les transaccions.

Com a característica innovadora de transaccions monetàries, hem pogut observar mitjançant la cadena de blocs i l'esnifatge de paquets, que la informació que es transmeten els nodes i s'emmagatzema a la cadena de blocs es troba sense xifrar. És a dir, manté una estructura que permet aportar transparència a les transaccions, tot mantenint les quatre característiques de seguretat de la informació abans comentades.

Referent a possibles línies de futur, caldria plantejar les fites que no hem pogut finalitzar amb aquest projecte com ara l'estudi de la relació de la xarxa TOR i Bitcoin així com diverses proves d'injecció d'informació modificada cap a la xarxa.

Altres possibles vies d'investigació podrien ser investigar el risc i la probabilitat de que un "virus" s'escrigui sobre les transaccions mitjançant proves sobre certs missatges, un anàlisi més a fons dels missatges d'alerta, verificar sistemes per identificar nodes miners o per exemple identificar les latències que provocaria una injecció massiva de missatges erronis al sistema.

El que podem afirmar però, és que per les possibles línies de futur que es puguin plantejar, aquest treball de fi de Màster pretén apropar el coneixement suficient perquè continuïn la feina.

5. Glossari

P2P→ Mètode de connexió Punt a punt. Permet gestionar xarxes distribuïdes.

TOR→ Xarxa que permet l'anonimat de les connexions a internet mitjançant la col·laboració de nodes.

Fees→ Comissió per transferència.

Sniffer→Software que permet llegir paquets que es transfereixen per un medi físic.

Wireshark→Software d'esnifatge de xarxes

Diagrama de GANTT→ Diagrama utilitzat en la planificació de projectes.

Autoritat Certificadora i Reguladora→Entitat que s'encarrega de verificar i confirmar les transaccions.

Repte Criptogràfic→Problema matemàtic introduït al sistema perquè els miners el resolguin creant blocs.

Hash→Part de codi que identifica una transacció o adreça.(Va signada amb la clau privada)

BTC→Inicials del Bitcoin.

AES256→Mètode de xifrat basat en 256 bits

SHA512→Mètode de xifrat basat en 512 bits

Salt→Número aleatori per la generació d'una password.

Força Bruta→Mètode per intentar esbrinar una password mitjançant la prova i error de totes les combinacions possibles.

TCP→Protocol fonamental d'internet i orientat a connexió.

IP→Adreça identificadora d'un node a internet.

Satoshi→ Nom de la persona o grup de persones pares del Bitcoin

Nodes→ Computadores connectades a la xarxa Bitcoin.

DNS→ Sistema de noms de dominis a internet.

IRC→Servei de missatgeria Chat.

WHO→Comanda de IRC per saber qui hi ha connectat.

Hard Coded→Expressió que es refereix a la informació incrustada dins el codi font d'una

aplicació.

HTTP→Protocol de connexió Web.

Paquet→Conjunt d'informació emmagatzemada i preparada per ser enviada.

Signatures→Tros de codi on s'ha aplicat un algorisme resultat de la barreja de dades i clau privada.

Bitcoin-qt→Client bitcoin amb GUI.

DynDns→Servei de resolució d'Ips de dominis.

Frames→Paquet de dades enviades en una mateixa vegada i esnifada amb wireshark

Dump→Bolcat d'informació

Saldo Confirmat→Saldo verificat una vegada s'han completat 6 blocs a la cadena de blocs principal.

6. Bibliografia

Informació general del Bitcoin

http://www.redusers.com/noticias/todo-lo-que-siempre-quisiste-saber-sobre-las-bitcoins/
https://en.bitcoin.it/wiki/Introduction
https://en.bitcoin.it/wiki/Getting_started
https://en.bitcoin.it/wiki/Mining
https://en.bitcoin.it/wiki/Alerts
https://en.bitcoin.it/wiki/Block_chain
https://en.bitcoin.it/wiki/Blocks
http://www.queesbitcoin.info/coacutemo-funciona-bitcoin.html
http://es.wikipedia.org/wiki/Bitcoin
http://www.bitcoinchart.co.uk/bitcoins/bitcoin-infographic-how-bitcoins-work/

Xarxa P2P de Bitcoin

https://en.bitcoin.it/wiki/Network
https://en.bitcoin.it/wiki/Protocol_Specification
https://en.bitcoin.it/wiki/Satoshi_Client_Node_Discovery
https://en.bitcoin.it/wiki/IP_Transactions
https://en.bitcoin.it/wiki/Transactions
http://blockchain.info/

Redacció, edició i disseny de la memòria

Material de suport de l'assignatura del Treball de Fi de Màster

Imatges i gràfics

http://en.wikipedia.org/wiki/File:Blockchain.png
http://www.bitcoinchart.co.uk/wp-content/uploads/2013/04/bitcoin-infographic.jpg

7. Annex

Estructura dels missatges

L'estructura bàsica d'un missatge de bitcoin és la següent:

Tamany	Descripció	Tipus de dada	Comentari
4	magic	uint32_t	Aquest valor identifica l'entorn on s'està enviant el missatge (MAIN/TEST)
12	command	char[12]	Defineix el tipus de missatge que s'està enviant
4	length	uint32_t	Longitud del payload
4	checksum	uint32_t	Sha256 del payload
?	payload	uchar[]	La resta de dades que venen definides per el tipus de missatge.

Tots els missatges tenen com a base l'estructura inicial de missatge abans comentada. A més depenen del tipus de missatge categoritzat, transmetrà una o altre informació per la xarxa.

version: Aquest és un missatge que es realitza quan es crea una connexió amb un node. El node local envia el missatge *version* amb les dades de versió per tal que el node remot li respongui amb les seves dades i el node local pugui verificar la compatibilitat de versió per realitzar-ne la connexió.

Tamany	Descripció	Tipus de dada	Comentari
4	version	int32_t	Versió utilitzada per aquell node
8	services	uint64_t	Característiques per la versió actual
8	timestamp	int64_t	Timestamp
26	addr_recv	net_addr	Adreça IP del node destí.
version >= 106			
26	addr_from	net_addr	Adreça IP del node origen.
8	nonce	uint64_t	Node random nonce, randomly generated every time a version packet is sent. This nonce is used to detect connections to self.
?	user_agent	var_str	User Agent (0x00 if string is 0 bytes long)
4	start_height	int32_t	Últim bloc rebut pel node emisor
1	relay	bool	Whether the remote peer should announce relayed transactions or not, see BIP 0037, since version >= 70001

verack: Aquest és un missatge de retorn d'un *version* que serveix per indicar que les versions dels dos nodes que volen connectar són compatibles. Observarem que només està format per una capçalera de missatge amb el *string verack* , a més de l'estructura bàsica dels missatges dins la xarxa.

Tamany	Descripció	Tipus de dada	Comentari
12	command	int32_t	Identifica la cadena verack com a ok de connexió
4	Payload	int32_t	Payload a 0 bytes de longitud
4	Checksum		

addr: Aquest missatge prové informació dels nodes de la xarxa. Si algun dels nodes no contesta en 3 hores és automàticament obviat.

Tamany	Descripció	Tipus de dada	Comentari

12	command	int32_t	Identifica el tipus de missatge addr
31	Payload	int32_t	Payload a 31 bytes de longitud
4	Checksum		

Payload

Tamany	Descripció	Tipus de dada	Comentari
1+	count	var_int	Número d'adreces incloses en aquest missatge
30x?	addr_list	(uint32_t + net_addr)[]	Adreces de la resta de nodes.

inv: De manera altruista o després de realitzar-ne una petició *getblocks*, un node pot rebre informació sobre un o més objectes mitjançant aquest missatge.

Tamany	Descripció	Tipus de dada	Comentari
?	count	var_int_	Número d'entrades d'inventari
36x?	inventory	inv_vect[]_	Inventari

Cal dir però que com a màxim es poden enviar 50000 entrades o l'equivalent en 1.8Megabytes.

getdata: Aquest missatge s'utilitza normalment després de rebre un inventari mitjançant el missatge *inv*. Una vegada s'han filtrat els objectes del qual el node disposa fa una petició *getdata* per demanar el contingut d'algun dels elements.

Tamany	Descripció	Tipus de dada	Comentari
?	count	var_int	Número d'entrades d'inventari
36x?	inventory	inv_vect[]	Inventari

notfound: Com a resposta d'una petició *getdata*, s'envia aquest missatge si no es pot retransmetre la informació demanada per el node que ha enviat el *getdata*. Un exemple d'això seria que l'objecte demanat no està ja a la memòria de la cua.

Tamany	Descripció	Tipus de dada	Comentari
?	count	var_int	Número d'entrades d'inventari
36x?	inventory	inv_vect[]	Inventari

getblocks: Mitjançant aquest missatge es retorna un *packet inv* amb una quantitat de blocs determinada per l'últim *hash* conegut de bloc que el client te a la llista de *hashs*. Aquesta llista arribarà fins al *hash_stop* o fins a 500 el que sigui primer.

Tamany	Descripció	Tipus de dada	Comentari
4	version	uint32_t	Versió del protocol
1+	hash count	var_int	Número de hashos de blocs
32+	block locator hashes	char[32]	Hash del bloc actual des de l'inici
32	hash_stop	char[32]	Hash del bloc final a descarregar. Màxim 500.

getheaders: Retorna un paquet de capçaleres on estan les capçaleres dels blocs a partir de l'últim conegut fins al hash_stop o 2000 blocs.

En comptes de rebre el bloc sencer, els clients més lleugers prefereixen demanar les capçaleres dels blocs per descarregar la cadena de blocs de manera mes ràpida. El client que nosaltres

utilitzem realitza aquesta petició per descarregar-se les capçaleres dels blocs.

Payload:

Tamany	Descripció	Tipus de dada	Comentari
4	version	uint32_t	Versió del protocol
1+	hash count	var_int	Número de hashos de blocs
32+	block locator hashes	char[32]	Hash de la capçalera bloc actual des de l'inici
32	hash_stop	char[32]	Hash de la capçalera de bloc final a descarregar. Màxim 2000.

Tx: Com a resposta d'un *getdata* rebem un objecte tipus tx o transacció de bitcoin.

Tamany	Descripció	Tipus de dada	Comentari
4	version	uint32_t	Versió de la transacció
1+	tx_in count	var_int	Número de transaccions d'entrada
41+	tx_in	tx_in[]	Les transaccions d'entrada
1+	tx_out count	var_int	Número de transaccions de sortida
9+	tx_out	tx_out[]	Transaccions de sortida
4	lock_time	uint32_t	Variable que bloqueja el fet d'afegir una transacció a un bloc. Si la seqüència és ffff és una variable obviable sinó no afegeix la transacció al bloc fins que hi sigui oportú.

Estructura *TxIn*

Tamany	Descripció	Tipus de dada	Comentari
36	previous_output	outpoint	Punt de sortida de la transacció o transacció previa.
1+	script length	var_int	Longitud de la signatura
?	signature script	uchar[]	Signatura
4	sequence	uint32_t	Transaction version as defined by the sender. Intended for "replacement" of transactions when information is updated before inclusion into a block.

Estructura OutPoint:

Tamany	Descripció	Tipus de dada	Comentari
32	hash	char[32]	El hash de la transacció de referencia
4	index	uint32_t	Un índex de la transacció de sortida.

Estructura TxOut

Tamany	Descripció	Tipus de dada	Comentari
8	value	int64_t	Valor en BTC de la transacció
1+	pk_script length	var_int	Longitud del pk_script
?	pk_script	uchar[]	Clau pública

blocks: Com a resposta d'un missatge getdata que demana informació transaccional sobre un hash de bloc es rep el següent missatge.

Tamany	Descripció	Tipus de dada	Comentari
4	version	uint32_t	Versió de bloc, quan va ser generat per el client que el va generar

32	prev_block	char[32]	Hash del bloc anterior.
32	merkle_root	char[32]	Referencia a l'arbre Merkle
4	timestamp	uint32_t	Moment de creació del bloc
4	bits	uint32_t	Dificultat de creació del bloc
4	nonce	uint32_t	
?	txn_count	var_int	Número d'entrades de transacció
?	txns	tx[]	Transaccions de bloc.

headers: És la resposta al missatge de *getheaders*, retorna les capçaleres de bloc mitjançant la següent estructura.

getaddr: Rep com a resposta un missatge *addr* amb la informació referent als nodes que estan actius a la xarxa per tal d'actualitzar la llista de nodes amb qui connectar.

checkorder, submitorder, reply: Aquests tipus de missatges serveixen (en transaccions de tipus IP) perquè els nodes cerciorin, realitzin i retransmetin les dades referents a transferències de coins mitjançant el protocol IP.

ping: Els missatges de ping permeten al client comprovar la connectivitat dels nodes remots i e cas de no trobar-ne, esborrar-ho de la llista.

alert: Les alertes són missatges que s'envien per la xarxa i que han de ser legítims dels administradors de bitcoin. Si un missatge no és legítim, no es fa arribar a cap node actiu.

Per verificar que un missatge es legítim, el client que ha de retransmetre el missatge valida que la seva signatura hash és vàlida mitjançant la clau pública del grup de desenvolupament del nucli de Bitcoin.