Broadcast de blocs de Bitcoin per un canal òptic.

Kilian Niubó Vinuesa

Resum— A l'ambit informàtic un dels temes claus és el de la seguretat. Qualsevol dispositiu conectat a la xarxa pot ser objectiu d'atac. La preocupació per la seguretat als nostres dispositius s'incrementa quan poden quedar exposats els nostres actius. En el següent treball es presenta una solució per a mantenir actualitzada la blockchain a un equip sense connexió de tal forma que poguem formar i firmar transaccions per a poder enviar-les a la xarxa de Bitcoin per a la seva validació tot mantenint les nostres claus tant públiques com privades offline. Mantenir les claus públiques en un entorn offline impedirà que algun atacant pugui esbrinar el balanç de les direccions associades a aquesta clau mitjançant tècniques de clustering. Per a mantenir la cadena de blocs de Bitcoin actualitzada s'ha creat un entorn connectat a la xarxa P2P que mantindrà actualitzada la blockchain i ens permetrà codificar els blocs en hexadecimal en codis QR. Aquests codis QR els mostrarem un darrere de l'altre per a que des d'un entorn sense connexió i amb una webcam poguem anar llegint aquests codis. Finalment, recomposarem el bloc i l'afegirem a la cadena de blocs, mantenint així un entorn actualitzat.

Paraules clau— Bitcoin, Blockchain, Python, QR, Bloc, Transmissió, Nodejs, Vue, Bitcoin Core, RPC, JSON, QRCode, OpenCV, Pyzbar, Cold Wallet, Sincronització

Abstract— In the IT field, one of the key issues is security. Any device connected to the network can be the target of an attack. The concern for the security of our devices increases when our assets can be exposed. In the following project we present a solution to keep the blockchain updated on an offline device so that we can form and sign transactions to send them to the Bitcoin network for validation, keeping our public and private keys offline. Keeping our public keys in an offline environment will prevent an attacker from discovering the balance of the addresses associated with these keys through clustering techniques. To keep the Bitcoin blockchain updated we have created an environment connected to the P2P network that will keep the blockchain updated and will allow us to encode the blocks in hexadecimal in QR codes. These QR codes will be displayed one after another so that from an offline environment and with a webcam we can read these codes. Finally, we will recompose the block and add it to the blockchain, thus maintaining an updated environment.

Index Terms—Bitcoin, Blockchain, Python, QR, Block, Transmission, Nodejs, Vue, Bitcoin Core, RPC, JSON, QRCode, OpenCV, Pyzbar, Cold Wallet, Synchronization

1 Introducció

Bitcoin va néixer després de la crisi econòmica de 2008, quan una persona (o diverses), sota el pseudònim de Satoshi Nakamoto va idear un sistema per el que una persona podia enviar "monedes" a una altra sense pasar per cap organisme entremig, ja que, en les seves paraules, "*no existeix cap mecanisme capaç de realitzar pagaments a través d'un canal de comunicació sense una part de confiança.*" **[1]**

Des del seu llançament al 2009, Bitcoin ha suposat un canvi de paradigma al model tradicional dels bancs, per la seva descentralització. Aquesta descentralització prové de la tecnologia anomenada cadena de blocs o blockchain. Aquesta blockchain es composa de tots els blocs creats fins a l'actualitat i funciona con un llibre de comptes públic de totes les transaccions que han succeït des de l'inici. Dins un bloc trobem, entre altres coses, les transaccions confirmades i el hash del anterior bloc. La blockchain funciona sobre una xarxa P2P. Els nous blocs són creats pels miners, que s'encarreguen d'incloure les transaccions a aquests blocs. Un cop, mitjançant les proves de treball (Proof of Work), troben una solució al problema criptogràfic actual del bloc que està construint, el nou bloc es considera vàlid i s'envia a tots els nodes als que es troba connectat. Els blocs són validats per tots els nodes abans d'incloure'ls a les seves cadenes, el que fa incrementar la robustesa del sistema.

Les criptomonedes poden ser considerades com un sistema monetari i, com en el cas dels bitllets, no tenen un valor pel material del que estan fetes, com ho podrien ser les monedes d'or, sinó que ve determinat per l'oferta i la demanda, és a dir, per la gent que les ven, les compra i, al mateix temps, els hi dona valor.

Aquestes monedes utilitzen la criptografia com a medi per aconseguir un sistema segur de pagaments i cobraments, que, al mateix temps, atorga anonimat als usuaris que les

- *E-mail de contacte: Kilian.niubo@e-campus.uab.cat*
- *Menció realitzada: Tecnologies de la Informació*
- *Treball tutoritzat per: Jordi Joancomartí Herrera (dEIC)*
- *Curs 2020/21*

utilitzen. Aquest anonimat, però, es pot veure compromès en algunes situacions com veurem més endavant.

Així doncs, ens trobem en un medi descentralitzat en el que la criptografia juga un paper molt important, tant en l'anonimat dels usuaris com en la composició mateixa de la xarxa. Els nodes de la xarxa P2P mantenen tota la cadena de blocs, substituint així les entitats intermèdies i el problema de la confiança en tercers.

Inicialment per a treballar amb el sistema bitcoin necessitem generar una clau privada. Amb aquesta clau privada i seleccionant un punt a l'atzar de la corba el·líptica de bitcoin, podem generar una clau pública. Aquesta clau pública ens permetrà compartir informació amb altres usuaris sense exposar les nostres criptomonedes. Amb aquesta clau pública podrem crear les direccions. Una direcció és un identificador que ens permet rebre i enviar monedes.**[2]**

Un wallet és el software que emmagatzema les claus públiques i privades per poder gestionar les nostres monedes. Podem dividir els wallets en dos grups, calents i freds. **[3]**

Els wallets calents són aquells que es troben connectats a la xarxa. Aquests poden ser un Exchange o aplicacions d'escriptori. Aquests Exchanges són un punt d'intercanvi de moneda, ja sigui per altres monedes o per diners fiat. Degut a que la inclusió d'aquests últims a l'equació suposa la participació dels bancs en l'intercanvi, aquests exchanges poden suposar un punt feble en l'anonimat del sistema, ja que per una banda tenim un sistema que ens brinda anonimat personal, exposant les nostres transaccions i per l'altra un sistema que amaga les nostres transaccions públicament però disposa de totes les nostres dades personals.

Els wallets freds són els que no es troben connectats a la xarxa, com els wallet físics.

Aquesta divisió sobre la connectivitat del nostre wallet és rellevant per al tema de la seguretat, ja que la connexió a la xarxa dels nostres dispositius o el nostre software pot comprometre la seguretat de les nostres claus.

Per a realitzar una transacció el que necessitarem serà poder saber quins són els UTXO (Transaccions de sortida no gastades) dels que enviarem les monedes i disposar de la nostra clau privada per a firmar la transacció. Un cop generada i signada es podrà enviar a la xarxa P2P de bitcoin per a que es validi i s'inclogui al següent bloc i, posteriorment, a la cadena.

Tenir un dispositiu connectat a la xarxa P2P de Bitcoin permetrà veure a la cadena de blocs les transaccions que s'estan afegint als blocs, així com les UTXO necessàries per a realitzar una transacció.

Si aquest dispositiu disposa de les claus privades i públiques es podran crear i signar transaccions i enviar-les a la xarxa. Tot i així, estar en possessió de les claus publiques i privades pot suposar un risc per a la seguretat, ja que estar en possessió de la clau privada significa poder controlar els fons de totes les adreces.

Actualment, amb un sistema de wallet offline podem, amb dos dispositius diferents, tenir la clau privada en un sistema que no està connectat, evitant així que es vegi compromesa la nostra clau privada en cas d'intrusió al nostre sistema connectat.

Per a realitzar una transacció, el dispositiu connectat amb les claus públiques podria crear les transaccions però no firmar-les, ja que és necessària la clau privada. Aquesta transacció hauria de ser firmada amb la màquina sense connexió que disposa de la clau privada. Un cop firmada s'hauria d'enviar mitjançant la primera màquina a la xarxa per tal de que s'inclogui al següent bloc de la cadena de blocs.

Degut a que per a realitzar una transacció necessitem tenir la blockchain actualitzada, de forma que puguem recuperar les nostres direccions actualitzades i així crear la transacció, la part de la clau pública es podria veure compromesa.

Tot i que a priori no suposa un problema que algú disposi de la nostra clau pública, ja que a partir d'ella no podrà aconseguir la clau privada, estar en possessió de la nostra clau pública permetria a un atacant esbrinar el balanç total de les direccions creades amb aquestes claus públiques.

2 Objectius

L'objectiu d'aquest treball és aconseguir realitzar transaccions i firmar-les en un entorn totalment offline.

De cara a solucionar els problemes que poden esdevenir de la sincronització via online, durant aquest treball es durà a terme la configuració i implantació d'un sistema d'emissió així com de recepció i sincronització dels blocs de la blockchain.

Per a fer-ho, en una primera part codificarem els blocs en QR. Aquests codis QR s'enviaran per un canal òptic.

Seguidament, el sistema de wallet offline haurà de llegir aquests QR amb una càmera i afegir els blocs a la seva blockchain local per tal de mantenir-la actualitzada.

Finalment, si s'aconsegueix implantar aquest sistema de Bitcoin, s'estudiarà la possibilitat de modificar els sistemes de transmissió com de recepció per ajustar-lo a altres criptomonedes.

3 Estat de l'art

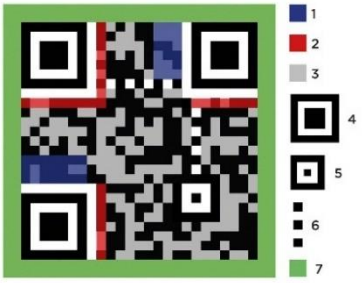
Actualment, amb un sistema de wallets hardware o cold wallets necessitem, per una banda, construir la transacció en una màquina que estigui actualitzada per obtenir les dades de les UTXO. Aquestes són les transaccions de sortida no gastades que serviran com a entrada de la nova transacció. Amb aquestes dades i la direcció on volem moure aquestes monedes podem generar l'esquelet d'una transacció.

Tot i això, aquesta transacció no es pot gastar encara. Per a que la transacció sigui efectiva necessitem firmar aquesta transacció per tal que es pugui demostrar que som els

propietaris de les direccions que diem controlar.

Aquestes transaccions han de ser firmades per la clau privada.

Així doncs, un cop formada la transacció ens l'hauríem d'emportar des de la màquina connectada a la maquina desconnectada amb, per exemple, un USB, firmar-la a la màquina sense connexió i emportar-nos-la a un sistema connectat per tal d'enviar-la a la xarxa de Bitcoin per a la seva validació i inclusió al següent bloc de la cadena.



1-Informació de la versió
2-Informació del format
3-Correcció d'errors i dades
4-Posicionament
5-Alineació
6-Sincronització
7- Zona quieta

Fig. 1:Estructura d'un codi QR

4 METODOLOGIA

Per a realitzar el treball s'ha fet servir una metodologia Kanban.

Aquesta metodologia permet identificar de manera molt visual les tasques del projecte i en quin estat es troben. Mitjançant aquestes etapes podrem aconseguir un canvi incremental i evolutiu al nostre projecte.**[4]**

L'eina utilitzada per a crear les tasques serà un Kanban sobre la plataforma Trello.

Utilitzar aquest entorn ha permès, a banda de realitzar un seguiment de l'estat de les tasques, separar-les en diferents grups de manera que per una part teníem la preparació de la creació i la transmissió de dades i per l'altre la de recepció, ajudant així a la modularització del projecte.

5 ANÀLISIS PREVI

5.1 Codis QR

Per poder solucionar els problemes derivats de l'obtenció de dades per les vies tradicionals i per a mantenir l'entorn totalment desconnectat de la xarxa s'ha optat per la transferència d'aquestes mitjançant codis QR.

Els codis QR van ser creats al 1994. La seva popularització, però, ha arribat al públic més general des de fa pocs anys. Això es deu a que, a diferència dels codis de barres tradicionals, no es necessiten dispositius especials per llegir-los. Amb l'expansió dels mòbils, qualsevol persona amb el software adequat pot llegir un codi QR. En adició, els codis QR disposen d'una major capacitat d'emmagatzematge de dades respecte als seus predecessors.

Aquests codis són una matriu de punts en dos dimensions en els que podem codificar des d'una pagina web, contactes, correus electrònics, fins a llargues cadenes de caràcters.

Les dades teòriques d'emmagatzematge de dades en aquests codis varien depenent del tipus de dades que vulguem emmagatzemar. Així doncs, en un codi QR podem introduir 7089 valors numèrics, 4296 alfanumèrics, 2953 bytes (o, el que és el mateix, 23624 bits) o 1817 Kanjis amb el grau de correcció més baix, un 7% .**[5]**

5.2 Blocs de Bitcoin

Per a poder abordar el problema que volem realitzar, que, al cap i a la fi consistirà en enviar aquests blocs hem de tenir en ment certes característiques.

Bitcoin no creix més ràpid en funció de la potencia de càlcul que tinguin els miners o el seu número. De cara a mantenir un número constant d'emissió de blocs, la dificultat de minar un bloc de Bitcoin és variable. Aquesta dificultat s'ajusta cada 2016 blocs, és a dir, 20.160 minuts o, en termes més comprensibles 14 dies. Aquesta regulació va enfocada a que el temps entre bloc i bloc ha de ser d'una mitja de 10 minuts. En aquests 10 minuts tenim la nostra primera línia vermella al treball, ja que hem de ser capaços de codificar els blocs en un temps menor.

Sabent això l'altre pregunta clau es: Quant ocupa un bloc?

Si la codificació està limitada temporalment, hem de saber quantes dades hem de codificar, si podem fer-ho i com podem fer-ho.

Si bé és cert que la mida màxima d'un bloc en uns inicis era de 1MB, posteriorment es va augmentar aquesta capacitat. Si mirem els valors reals veiem que en el últims tres anys aquest valor es troba entre els 0.8 i 1.30 MB. **[6]**

Fig. 2: Mida mitja d'un bloc

Per tant, tenint aquestes dades presents, el que farem es pot resumir com a codificar 1,30MB de dades en codis QR en un temps límit de 10 minuts.

6 ENTORN DE TREBALL

Com s'ha mencionat anteriorment l'objectiu d'aquest treball és el d'aconseguir crear i firmar transaccions en un entorn completament offline. Per aconseguir això el que necessitem, i que constitueix el gruix del treball, és mantenir la cadena de blocs actualitzada en un entorn offline.

Així doncs necessitarem inicialment dos dispositius, un amb la capacitat de capturar codis QR i que no estigui connectat a la xarxa i l'altre connectat a la xarxa que s'encarregui de mantenir la cadena de blocs actualitzada per a poder crear la codificació i l'emissió d'aquests blocs en codis QR.

Amb aquestes consideracions presents, podem dividir el projecte en dues parts ben diferenciades, la part d'emissió i la part de recepció.

Per a tenir tota la cadena de blocs de bitcoin, i poder treballar amb ella, en els dos entorns es disposa de Bitcoin Core.

6.1 Bitcoin Core

Bitcoin Core **[7]** és un projecte de codi obert que implementa totes les funcionalitats de Bitcoin.

Bitcoin Core es connecta a la xarxa com un node complet, emmagatzema i valida els blocs i les transaccions de la cadena de blocs. Això significa que, amb un client de Bitcoin Core actualitzat, tindrem accés a tots els blocs i tota la informació que aquests contenen.

El fet de mantenir tota la cadena de blocs suposa que hem de reservar més de 350 Gb d'espai per a tenir el node complet.

Addicionalment, Bitcoin Core inclou un daemon que permet realitzar crides RPC mitjançant una línia de comandes al client de bitcoin per a poder obtenir informació.

6.2 Entorn online

Per a l'entorn online s'ha utilitzat un ordinador amb 16 GB de RAM, 4 nuclis i 8 fils.

Es disposa d'un node de Bitcoin Core actualitzat i una versió de python 3.8 sobre spyder.

Aquest ordinador serà l'encarregat de realitzar les crides RPC per a obtenir informació sobre la cadena de blocs. Un cop s'hagi aconseguit la informació necessària sobre els blocs a codificar, aquests es convertiran en codis QR. Posteriorment i des d'aquesta màquina es mostraran els codis QR resultants per una pantalla auxiliar de forma que puguin ser escanejats i recompostos a l'entorn Offline.

6.3 Entorn offline

Per a l'entorn offline s'ha utilitzat un ordinador amb 8GB de RAM, 4 nuclis i 8 fils.

En aquest cas, el Bitcoin Core ha estat actualitzar fins a un cert block, sobre l'altura 687500, de manera que es puguin realitzar les proves de la inclusió dels següents blocs. Addicionalment, també es disposa d'una versió 3.8 de python per a la descodificació i inclusió dels blocs al client de Bitcoin Core.

Aquest ordinador disposa d'una càmera web amb 30 fps i addicionalment es disposa d'una altre de 60 fps connectada via USB per a realitzar les proves de lectura dels codis QR.

7 RESOLUCIÓ DEL PROBLEMA

Per a la resolució del problema el dividirem en diferents parts:

- Obtenció del bloc en cru
- Creació dels codis QR/ Trencament del bloc
- Emissió dels codis QR
- Lectura dels codis QR
- Inclusió dels codis QR a la cadena de blocs

Les tasques fins a la emissió, inclosa, es treballaran a la part de l'entorn online.

Tot i que es podrien treballar en un entorn offline, la idea és poder obtenir fins a l'últim bloc per a codificar-lo.

7.1 Obtenció del bloc en cru

Per a realitzar la obtenció del bloc s'ha programat una solució en python y amb l'ajuda de la llibreria bitcoinrpc. **[8]**

Abans de poder utilitzar aquesta llibreria s'ha hagut de generar un fitxer anomenat bitcoin.conf, en el que s'especifiquen dades com el rpcuser i la seva contrasenya, i l'habilitació del daemon per a poder fer les crides. Aquesta configuració serà carregada durant l'arrencada del Bitcoin Core.

La llibreria bitcoinrpc permet treballar amb el nostre node especificant la ip (en aquest cas, localhost), el port (per defecte 8332) i les dades introduïdes per a la connexió RPC.

Per a poder utilitzar les funcions implementades a la llibreria, hem de fer crides asíncrones, pel que necessitarem també de la llibreria asyncio.[9]

Les funcions que utilitzarem en aquesta part des de python són les que podríem usar en línies de comandes amb el client de bitcoin core. Aquestes són:

Getblockcount: Ens retorna el número de blocs totals, és a dir, l'alçada de la blockchain. Donat que els blocs es van posicionant correlativament, aquesta funció ens retorna un nombre enter que representa l'altura de l'últim bloc. Cadascun dels blocs estarà numerat amb un número que ens servirà com a identificador del bloc.

Getblockhash(altura): Aquesta comanda, passant-li com a paràmetre l'altura del bloc que volem, ens retorna el hash d'aquest bloc.

Getblock(hash,verbosity): Finalment, amb la comanda

getblockhash podrem obtenir totes les dades del bloc esperat. L'argument hash és el hash del bloc sol·licitat amb la comanda getblockhash. L'argument verbosity és un numèric que modifica la forma com ens seran retornades les dades. Per defecte és 1 i ens retorna un objecte JSON. En el nostre cas, farem servir un 0 ja que amb això el resultat de la consulta seran les dades codificades en hexadecimal.

7.2 Creació dels codis QR / Trencament del bloc

Donat a que l'espai per a les dades en un codi QR esta limitat i que el resultat del getblock ens retorna un hexadecimal de més de 3 milions de caràcters, a l'hora de la creació d'aquests QR hem de dividir el resultat en fragments que puguem codificar.

Teòricament i degut a que el resultat és en hexadecimal, inicialment es va estimar que la codificació de tota la cadena que conforma el bloc, és a dir, aquests 3.000.000 de caràcters, es podrien codificar tal qual estaven en codis QR de 4296 caràcters alfanumèrics, donant com a resultat uns 699 codis QR. L'altre opció que es va estudiar va ser la de convertir els hexadecimals a bits, ja que dos hexadecimals corresponen a un byte, per tant hauríem d'enviar 1.5 milions de bytes en codis de 2953, el que teòricament serien uns 507 codis QR.

Durant la creació dels codis QR, però, es va veure que aquesta capacitat teòrica queda lluny del que realment podíem codificar. Després de provar amb diferents llibreries, jugant amb el tipus de dades a codificar i la capacitat correctora d'errors per tal d'augmentar aquestes dades dins el codi QR, finalment s'ha optat per a utilitzar la llibreria qrcode de python.

Qrcode **[10]** de python és un generador de codis QR que inclou Pillow per a la generació d'imatges. Qrcode ens permet incloure uns 2330 caràcters hexadecimals a cada codi. Es va decidir fer codis de 2250 caràcters + 4 caràcters a l'inici que serveixen com a identificador de la part del bloc. Els QR contenen directament la informació en hexadecimal.

En contraposició dels càlculs inicials, per a codificar un bloc sencer necessitem generar al voltant de 1300 codis, el que provoca una desviació de gairebé un 90%.

Degut a aquest augment i la limitació temporal deguda a la generació dels blocs de Bitcoin, una tasca molt important ha estat tractar de reduir aquest temps mitjançant una programació eficient i diferents estratègies per aprofitar els recursos de l'ordinador que generava els codis.

S'han codificat dues solucions, ja que durant les proves de generació es va veure que un factor important era la CPU del nostre sistema.

La primera solució, per a ordinadors amb una capacitat de càlcul i de crides a sistema més elevades, consisteix en un programa que agafa tot el bloc, crea un directori que s'anomena com l'altura del bloc i crea les imatges dels codis en png. Aquestes imatges es creen amb el nom de la posició que ocupen aquests fragments de blocs i que correspon amb els 4 caràcters inclosos a l'inici del codi.

Addicionalment es creen dos codis més; un a l'inici que indicarà quin és el bloc que s'està emetent i un al final que servirà per indicar al lector que la transmissió del bloc ha arribat al seu final i que així pugui incloure les dades llegides com a un nou bloc a la seva cadena.

Per altra banda i per a solucionar els problemes derivats les restriccions temporals s'han crear 4 arxius python. Aquests arxius realitzen la mateixa funció que l'anterior, amb la particularitat que un cop que tenen el bloc, el divideixen en 4 parts iguals i cadascun s'encarrega de crear els codis QR d'aquestes parts. Amb aquesta divisió s'aconsegueixen aprofitar més els recursos disponibles del sistema i es poden generar tots els codis QR d'un bloc en un temps d'uns 340 segons, o el que és el mateix, uns cinc minuts i mig. Aquest resultat es considera correcte, tenint en compte que el temps de generació de cada bloc és d'uns 600 segons.

A la següent figura es mostra un exemple d'un codi generat. Si s'escaneja, es pot comprovar que els 4 primers dígits són 0621, és a dir, correspon al codi número 621 d'aquest bloc.

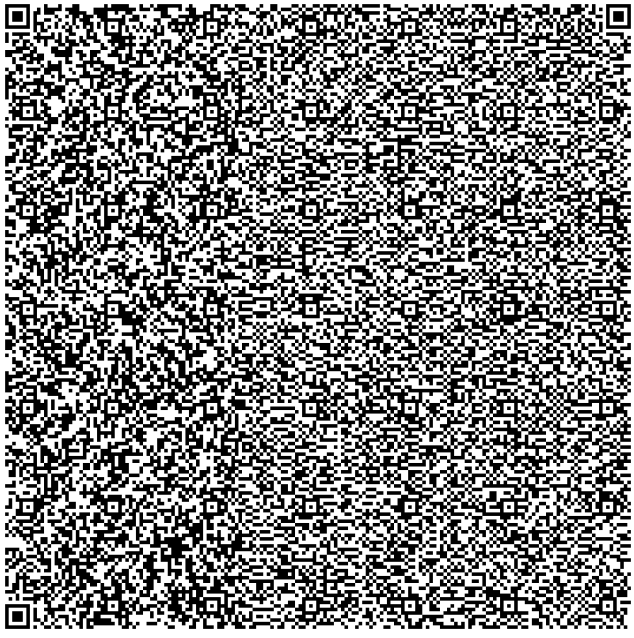


Fig. 3: Codi QR número 621 del bloc 688094

Cada imatge dels codis QR ocupen uns 12KB. Això es tradueix en que emmagatzemar un bloc de la cadena en format QR ocupa uns 15MB. Emmagatzemar tots els blocs de la cadena per a poder transmetre'ls en aquest format ocuparia uns 3.75TB.

7.3 Emissió dels codis QR

El mostreig dels codis QR generats s'ha modificat al llarg del projecte. Inicialment, es va preparar un entorn per a la plataforma Twitch, de manera que els codis s'anessin mostrant en temps real conforme s'anaven generant mitjançant

la llibreria Tkinter, ocupant els deu minuts entre bloc i bloc.

D'aquesta solució es van derivar dos grans problemes: Per una part, necessitàvem tenir l'ordinador offline completament actualitzat a l'hora de llegir els codis a més era molt probable que al començar a llegir els codis es trobés que estava llegint la meitat d'un bloc, fent inservible l'emmagatzematge de les dades llegides. Per l'altra part, els problemes derivats de les transmissions en viu feien si hi havia un petit retràs tant en l'emissió com en la recepció i es perdia un codi, tots els següents ja no eren vàlids.

Per abordar les dues qüestions alhora, s'ha dissenyat una aplicació web amb Node i Vue.js.

En aquesta aplicació ens trobem inicialment amb un rectangle buit, que ens indicarà on apareixeran els blocs per a descodificar. A la part superior esquerra trobem a més un comptador que mostrara a quin dels QR ens trobem. És merament informatiu ja que el valor màxim d'aquest comptador pot variar en funció de quants codis s'hagin generat.

Per altra banda, trobem un requadre on hi haurem d'ingressar el número de bloc que volem escanejar. És important que el bloc s'hagi generat amb anterioritat ja que, pel contrari, no es mostrarà cap imatge.

Per sota d'aquest número de bloc, hem d'ingressar el temps entre imatges en milisegons. Aquest paràmetre és especialment important ja que determinarà el temps que tardarem en poder recomposar el bloc. S'ha deixat per defecte un valor de 400 ms, el que ens permetrà llegir fins a 1500 codis en 10 minuts. No s'ha trobat durant l'experimentació cap bloc que arribi als 1400 codis.

Finalment disposem d'un botó start que servirà per que es comencin a mostrar els blocs.

En el moment de l'inici del mostreig és important que l'entorn de recepció estigui preparat i centrat amb la imatge per a poder començar a descodificar els blocs.

Fig. 4: Part de la plataforma web per a l'emissió

7.4 Lectura dels codis QR

La lectura dels codis QR és el primer pas dins l'entorn offline.

Per a poder realitzar tant la lectura com les descodificacions dels codis QR necessitarem dues llibreries claus: opencv-python i pyzbar.

Opencv és una llibreria de visió per computador que ens permetrà interactuar amb les imatges que es capturin desde la nostra càmera web. **[11]**

Pyzbar per la seva banda ens permet descodificar un codi QR en una imatge. **[12]**

En el nostre cas, ambdues llibreries han de treballar alhora. En un primer moment cv2 s'encarrega d'iniciar-nos la nostra càmera i posar-la a funcionar, definint les dimensions de la captura. Per a poder enfocar la càmera i centrar el dispositiu d'emisió a la imatge, al iniciar la captura es mostra una finestra amb el que esta capturant la càmera. És en aquest moment quan hem de centrar la imatge al cuadrat mostrat anteriorment a l'entorn web d'emissió.

Un cop comença la captura d'imatges, la llibreria pyzbar buscarà al frame que li passi la nostra càmera algun codi QR i el descodificará.

Un cop feta aquesta descodificació, compararà els 4 primers dígits amb els 4 digits llegits anteriorment, si són els mateixos, descodificarà un altre cop el codi que aparegui al frame.

En el moment en que es trobi una capçalera del bloc que sigui diferent a l'anterior, afegirà la part restant a un txt. Aquest txt es crea en el moment en que s'escaneja el primer bloc i que conté el nom que portarà l'arxiu, corresponent a l'alçada del bloc.

Finalment, quan el bloc final es mostri per pantalla, es cridarà a la funció per afegir el bloc al la cadena de blocs en local.

7.5 Inclusió dels codis QR a la cadena de blocs

En aquesta última part i amb tots els codis llegits i emmagatzemats en un .txt, l'últim pas que ens queda és afegir-lo a la nostra cadena.

Per a realitzar aquesta operació es torna a fer servir la llibreria bitcoinrpc.

En aquesta ocasió, però, el mètode que necessitem per a incloure un bloc a la nostra cadena no està definit a la llibreria. Tot i això, aquesta llibreria ens permet definir noves funcions per interactuar amb el client de bitcoin Core. Així doncs, s'ha creat una nova funció anomenada submitblock.

Submitblock(bloc): És una funció existent a Bitcoin Core que rep com a paràmetre el bloc en hexadecimal. En aquest cas, si els codis QR que formen el bloc han estat llegit correctament, disposarem d'un arxiu que s'anomena com l'identificador del bloc, i que idealment hauria de ser el resultat de la consulta getblockcount+1. Al executar aquesta funció podem obtenir dos resultats:

RPCError: (-22, 'Block decode failed'): En aquest cas el problema es deu a que el codi esta mal format i no s'ha pogut incloure a la cadena.

Si el bloc s'inclou correctament a la cadena el Bitcoin Core no ens retorna cap missatge.

Finalment, si tractem d'afegir el mateix codi dues vegades, el sistema ens retorna un missatge de "duplicated", impedint-nos tornar-lo a afegir.

Dins el programa s'ha afegit una petita validació, que

consta de la comparació de l'altura que tenia abans la cadena de blocs i es compara amb l'actual, si no és la mateixa significa que la cadena és més llarga i, per tant, s'ha afegit el bloc correctament. Es mostra un missatge per consola per a que l'usuari tingui algun feedback d'aquest inclusió.

A la següent figura es mostra l'esquema de tot l'entorn en marxa, diferenciant les parts segons el seu apartat:

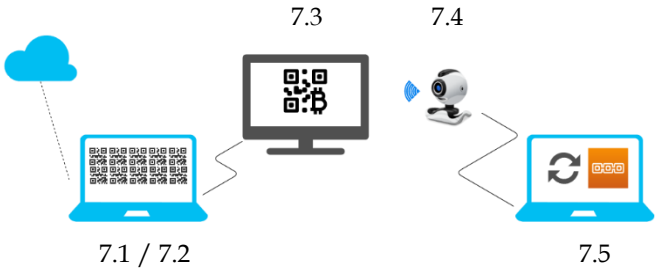


Fig. 5: Esquema de l'entorn d'emissió i recepció

8 Resultats

Per a realitzar una valoració dels resultats els dividirem com hem fet durant tot el treball en els dos entorns disponibles.
Inicialment, durant la recuperació del bloc les crides per a obtenir la informació del bitcoin core han estat en tots els casos molt ràpides i sense problemes més enllà de la configuració del fitxer config.

A l'etapa de la fragmentació dels blocs, treballant amb la primera versió s'ha obtingut una mitja dels resultats obtinguts en la generació de 10 blocs d'uns 630 segons per blocs. Tot i que aquesta mitja no s'allunya massa dels 600 segons necessaris per a considerar-la vàlida, veient el creixement que estan tenint els blocs i la seva variabilitat és del tot insuficient per a mantenir un ritme de creació òptim. Per altra banda, la limitació de l'equip disponible ha jugat un paper important en l'obtenció d'aquests temps, podent-se millorar amb hardware més potent.

La no conformitat amb aquests resultats va derivar en la creació de la segona versió de la creació de blocs. Es va detectar que amb la primera versió la CPU treballava amb un rendiment del 20% durant tota la creació, pel que es va decidir dividir tot el bloc en quatre parts. Amb aquesta segona versió la CPU treballa a uns valors per sobre del 80% i el temps de la generació dels blocs es redueix fins a una mitja d'uns 343 segons. La reducció en un gairebé 50% del temps necessari en la generació dels blocs esdevé un resultat molt satisfactori i que pot suportar la crescuda dels blocs de Bitcoin fins a uns 2MB sense necessitat d'actualitzacions.

Durant la part d'emissió els resultats són senzills però efectius. Tot i que els blocs no passen successivament, l'entorn creat ens serveix per veure y poder comprovar la funcionalitat de les altres parts del projecte.

La part de recepció és la més delicada de totes. Mentre que a la part d'emissió un petit retràs en la generació d'una imatge no suposa cap problema, a l'hora de llegir-los sí. Durant les proves de recepció s'ha jugat sobretot en la manera en com tractar les dades per fer el més àgil possible el programa i que tingui una gran capacitat de lectura per tal de poder accelerar el mostreig dels QR reduint així el temps total que necessitem per a rebre tot un bloc. Les proves experimentals en aquest aspecte no han sigut gaire bones. S'han tractat d'emmagatzemar temporalment les dades i afegir-les totes juntes amb diferents mètodes però finalment l'única manera que s'ha trobat de fer possible l'emmagatzematge de tots els trossos de bloc ha estat guardant cada resultat llegit un cop es llegeix.

Tot i això, aquests resultats no han estat en tots els casos positius. Si bé és cert que en més de la meitat dels casos el bloc s'ha recompost i afegit a la cadena correctament, un dels factors més importants ha estat el de la velocitat d'emissió. Amb un paràmetre de 600ms entre imatge i imatge el factor de recepció és del 100%. Aquesta velocitat és insuficient per equiparar el ritme de la inclusió al de la seva generació.

Les proves realitzades amb 400 ms entre imatges no ens ha donat sempre resultats satisfactoris. Durant aquestes proves, els resultats de la lectura dels QR ha oscil·lat entre un 95% i un 100% dels blocs recompostos correctament. La impossibilitat de llegir correctament el 100% dels blocs totes les vegades fa que no puguem considerar òptim aquesta velocitat de transmissió per a l'equip amb que disposàvem.

La velocitat pràctica al llegir els codis, es troba entre 2.5 i 3 KB per segon. Aquesta velocitat és baixa si la comparem amb velocitats de transmissió actuals, però degut a la capacitat dels codis i la nostra capacitat de lectura no s'ha pogut incrementar més.

A l'última part, a l'hora d'incloure el bloc, no s'ha detectat cap problema sempre que aquest bloc estigui ben format i no li falti cap caràcter hexadecimal.

Així doncs, comptant que es realitzin les proves d'emissió en 400 ms i no hi hagi errades, si tenim que llegir 1300 codis, el temps que trigaríem a actualitzar un bloc des de que apareix a la cadena de blocs fins que el tenim actualitzat a la nostra blockchain local, seria d'entre 14 i 15 minuts, comptant la creació, l'emissió, recepció i inclusió del bloc.

9 Conclusió

La transmissió de dades mitjançant codis QR és un sistema altament ineficient si el que estem cercant és la velocitat. Tot i això, s'ha pogut crear un sistema amb un elevat percentatge d'èxit i amb potencial de millora.

És possible mantenir la cadena de blocs actualitzada des de un entorn offline i, en conseqüència, poder treballar amb aquest entorn per a formar i signar transaccions, tot mantenint les claus públiques i privades sense necessitat d'estar en un entorn connectat a la xarxa.

El fet d'aconseguir transmetre dades d'aquesta via obre les portes a noves possibilitats per a les wallets offline i, si tot

això ve acompanyat d'una optimització en la recepció, es podrien estudiar altres aplicacions d'aquesta tecnologia.

10 Futures línies de treball

10.1 Treballant sobre el projecte realitzat

Tot i que els resultats finals són satisfactoris en molts aspectes i els mòduls per separat funcionen correctament, de cara a que aquest projecte sigui totalment funcional i automàtic hi ha diversos aspectes en els que es podria treballar per a fer-lo un projecte sòlid.

Per una part, s'haurien de crear els codis QR de tots els blocs de la cadena per tal de que es pugui actualitzar des del punt en el que es vulgui cap endavant ja que durant la realització del treball s'han generat només els necessaris per a realitzar les proves d'inclusió a la cadena desactualitzada.

Aquesta generació que ara s'està fent sobre l'últim bloc disponible, s'hauria d'estudiar si es convenient fer-la sobre els blocs que estan sis posicions per sota, ja que un bloc es considera validat després de sis confirmacions. Amb això aniríem una hora per darrera de la cadena però ens estalviaríem problemes amb possibles Forks.

Per la part d'emissió s'hauria de treballar desenvolupant un entorn agradable per a la part de recepció. Això passaria per desenvolupar una plataforma web més depurada on es pugui escollir des de quin bloc es vol actualitzar la cadena i que automàticament es vagin actualitzant els successius i no un a un com s'està fent ara. A més, l'ideal seria poder accedir-hi des d'internet i no treballar només a local com s'ha fet fins ara.

Així mateix a la implantació offline s'hauria de depurar el mètode de lectura de blocs per tal d'optimitzar més el temps i poder fer una actualització de la cadena més ràpida, ja que si s'han d'actualitzar molts blocs el temps requerit ara és massa elevat com per guanyar temps a la pròpia generació de blocs de Bitcoin.

10.2 Altres cadenes de blocs

El fet que la majoria de criptomonedes treballin amb cadenes de blocs podria significar l'adaptació d'aquesta tecnologia de transmissió per a elles.

Primer de tot, i amb totes les monedes amb les que vulguem fer una adaptació per a aquest tipus de transmissió haurem de recodificar tota la part d'obtenció dels blocs, veure com podem obtenir-los i com els podem tractar i per la part de recepció, com els podem incloure en un entorn offline.

Si mirem monedes com Ethereum, els blocs són del voltant de 50Kb, és a dir, molt mes petits, però la seva generació és molt més ràpida, i, tot i que és variable, actualment es produeix un bloc cada 15 segons aproximadament. Això suposaria una transmissió de dades més elevada respecte a Bitcoin, pel que prèviament hauríem de millorar el programa.

En altres monedes, com Litecoin o Dogecoin, ens trobem amb un problema similat. Tot i que els blocs són similars en mida als de Bitcoin la seva generació és més rapida, fent que les possibles adaptacions per a aquestes monedes passin necessàriament per una millora en els mètodes de preparació dels codis a transmetre i en la seva recepció.

Agraïments

Voldria agrair especialment a en Jordi Herrera Joancomartí per endinsar-me en el món de les tecnologies blockchain i les criptomonedes i per la seva infinita paciència al treballar en la seva funció pedagògica.

Per altra banda, agrair a tots aquells que han hagut d'escoltar amb cara d'interessats els interminables problemes sorgits durant la realització del treball i a tots aquells que m'han animat a finalitzar-lo i a donar-li tota la meva dedicació quan els ànims i el temps era el que més em mancava.

Bibliografia

[1] Satoshi Nakamoto. Bitcoin: A Peer-to-Peer Electronic Cash System [En línia] Disponible a: https://bitcoin.org/bitcoin.pdf

[2] How does Bitcoin work? [En línia] Disponible a: https://bitcoin.org/en/how-it-works

[3] Cómo se pueden almacenar bitcoin y otras monedes. [En línia] Disponible a: https://academy.bit2me.com/como-almacenar-bitcoins/

[4] David Anderson (10 de Desembre de 2010). The principles of the Kanban method. [En línia] Disponible a: https://web.archive.org/web/20140114161522/http://www.djaa.com/principles-kanban-method

[5] Information capacity and versions of the QR Code [En línia] Disponible a: https://www.qrcode.com/en/about/version.html

[6] Tamaño del bloque promedio (MB) [En línia] Disponible a: https://www.blockchain.com/charts/avg-block-size

[7] Bitcoin core [En línia] Disponible a: https://bitcoin.org/en/bitcoin-core/

[8] Bitcoinrpc [En línia] Disponible a: https://pypi.org/project/bitcoinrpc/

[9] Asyncio – Asynchronous I/O [En línia] Disponible a: https://docs.python.org/3/library/asyncio.html

[10] Qrcode 6.1 [En línia] Disponible a: https://pypi.org/project/qrcode/

[11] Opencv-Python 4.5.1.48 [En línia] https://pypi.org/project/opencv-python/

[12] Pyzbar 0.1.8 [En línia] https://pypi.org/project/pyzbar/