Univerzita Komenského, Bratislava
Fakulta Matemetiky, Fyziky a Informatiky

Kryptografická mena Bitcoin

Bakalárska práca

2012
Albert Herencsár

Univerzita Komenského, Bratislava
Fakulta Matemetiky, Fyziky a Informatiky

Kryptografická mena Bitcoin

Bakalárska práca

Študijný program: Informatika
Študijný odbor: 2508 Informatika
Školiace pracovisko: Katedra informatiky
Školiteľ: RNDr. Peter Gaži, PhD.

Bratislava, 2012
Albert Herencsár

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Albert Herencsár
Študijný program: informatika (Jednoodborové štúdium, bakalársky I. st., denná forma)
Študijný odbor: 9.2.1. informatika
Typ záverečnej práce: bakalárska
Jazyk záverečnej práce: slovenský

Názov: Kryptografická mena Bitcoin

Cieľ: Bitcoin je elektronická mena navrhnutá v roku 2008 Satoshi Nakamotom a v súčasnosti aj reálne implementovaná a konvertibilná na svetové meny. Jej najdôležitejšou vlastnosťou je, že celý systém funguje bez centrálnej autority, ktorá by menu spravovala a bezpečnosť tejto meny stojí výlučne na vhodnom použití rôznych kryptografických nástrojov. Cieľom tejto bakalárskej práce je popísať fungovanie systému Bitcoin z bezpečnostného hľadiska a analyzovať predpoklady, o ktoré sa bezpečnosť tohto systému opiera. Práca by tiež mala podať prehľad známych analýz bezpečnosti a anonymity v prostredí platieb bitcoinmi a prípadne zahŕňať aj vlastné experimenty v tejto oblasti.

Vedúci: Mgr. Peter Gaži, PhD.

Dátum zadania: 25.10.2011

Dátum schválenia: 26.10.2011

doc. RNDr. Daniel Olejár, PhD.
garant študijného programu

.....
študent

.....
vedúci

Čestne prehlasujem, že bakalársku prácu som vypracoval samostatne s použitím uvedenej literatúry a pod dohľadom môjho vedúceho práce.

. . . . . .

Chcel by som sa poďakovať môjmu vedúcemu RNDr. Petrovi Gažimu, PhD. za jeho pomoc, za inšpiratívne rady a za dohľad nad mojou činnosťou.

Abstrakt

Bitcoinový systém je decentralizovaný systém elektronickej peňažnej meny, ktorý v súčasnosti získava čoraz väčšiu popularitu. Táto práca popisuje fungovanie Bitcoinového systému z technického hľadiska a zaoberá sa otázkami bezpečnosti systému a anonymity používateľov. Ako príklad zraniteľnosti systému autor predstavuje svoju softvérovú aplikáciu, ktorá umožňuje vkladať cudzie (nesystémové) dáta do databázy Bitcoinového systému.

Kľúčové slová: Bitcoin, transakcia, blok, ťažba Bitcoinov, bloková reťaz, anonymita, bezpečnosť, zraniteľnosť

Abstract

Bitcoin is a decentralized electronic currency system which is currently gaining increasing popularity. This thesis describes the operation of the Bitcoin system from a technical standpoint and deals with the questions of system security and user anonymity. As an example of a system vulnerability, the author presents his software application which enables to embed alien (non-system) data into the Bitcoin database.

Keywords: Bitcoin, transaction, block, Bitcoin mining, block chain, anonymity, security, vulnerability

Obsah

Úvod

Bitcoinový systém je elektronický peňažný systém, v ktorom používatelia komunikujú vzájomne, spôsobom peer-to-peer, bez existencie centrálneho riadenia. Tento systém, ktorý v súčasnosti získava čoraz väčšiu popularitu, umožňuje posielať on-line platby priamo od jedného používateľa druhému používateľovi bez zapojenia nejakej finančnej inštitúcie. V záujme overenia vlastníctva elektronických peňazí a na zabránenie ich dvojitého minutia (double-spending), tento systém sa spolieha na digitálne podpisy a na verejne známu históriu transakcií.

Systém Bitcoin vyvíjal Satoshi Nakamoto (pravdepodobne ide o pseudonym) od roku 2007 [1]. Prevádzka systému bola spustená začiatkom roka 2009. Satoshi Nakamoto vyvinul aj originálny softvér Bitcoin klient, ale od konca roku 2010 už neparticipuje na rozvoji systému. Identita Satoshiho je neznáma. Systém rozvíjajú už iní dobrovoľníci.

Táto práca popisuje fungovanie Bitcoinového systému z technického hľadiska a sústreďuje sa na otázky bezpečnosti systému a anonymity používateľov. Členenie tejto práce je nasledovné. V prvej kapitole je vysvetlený princíp fungovania Bitcoinového systému. Sú vysvetlené systémové pojmy, napríklad Bitcoinová peňažná mena, Bitcoinové adresy, transakcie, typy transakcií, blok, bloková reťaz a podobne. Druhá kapitola rozoberá problematiku anonymity používateľov. V tretej kapitole je prezentovaná oblasť bezpečnosti a možných ohrození systému. V štvrtej kapitole je popísaná možná spolupráca „baníkov“, t.j. najdôležitejších používateľov, ktorí zabezpečujú chod systému. V piatej kapitole, ako praktická ukážka zraniteľnosti systému, je uvedený autorom vytvorený softvér na vkladanie dát do databázy Bitcoinového systému.

Kapitola 1

Bitcoinový systém

Peniaze v Bitcoinovom systéme sa nazývajú Bitcoiny. Skratka pre Bitcoin je BTC. Najmenšia, ďalej už nedeliteľná jednotka peňazí v Bitcoinovom systéme je Satoshi. Platí, že 1BTC = 10^8Satoshi.

Na preukázanie vlastníctva Bitcoinov sa využívajú digitálne podpisy. Na ukladanie a verifikovanie transakcií sa používa verejne dostupná, distribuovaná databáza transakcií, ktorá slúži aj na prevenciu double-spendingu (dvojnásobného použitia rovnakých Bitcoinov podvodným spôsobom v rôznych transakciách). Táto databáza sa skladá z blokov, ktoré sa na seba nadväzujú, a v ktorých sú uložené transakcie všetkých používateľov. Bloky sa nevytvárajú automaticky, iba niektorí používatelia Bitcoinového systému ich vytvárajú. Vytvorenie platného bloku vyžaduje veľkú výpočtovú prácu. Každý nadväzujúci blok zvyšuje platnosť predchádzajúcich blokov a teda znižuje pravdepodobnosť, že transakcie v predchádzajúcich blokoch stratia svoju platnosť. Preto je prípadnému útočníkovi nepraktické vytvárať bloky na zvrátenie transakcií. Používateľ, ktorému sa podarilo vygenerovať blok, dostane za výpočtovú prácu, ktorú vynaložil na vytvorenie bloku, odmenu. Odmenou sú úplne nové Bitcoiny. Generovanie blokov sa preto nazýva aj ťažba (mining) Bitcoinov, keďže sa pri tom vytvárajú nové (odmenové) Bitcoiny. Používateľ, ktorý vytvára bloky, sa nazýva baník (miner). Veľkosť Bitcoinovej odmeny je presne určená v Bitcoinovom protokole a s postupom času sa bude zmenšovať tak, aby celkový počet Bitcoinov sa zdola približoval k hranici 21 miliónov Bitcoinov. Baníci, okrem vyššie uvedenej odmeny, môžu naviac dostať Bitcoiny aj na základe dobrovoľných transakčných poplatkov od ostatných používateľov systému. Obtiažnosť vygenerovania blokov sa pravidelne aktualizuje podľa výpočtovej sily baníkov. Systém je bezpečný, pokiaľ výpočtová sila poctivých baníkov je väčšia, ako výpočtová sila akejkoľvek skupiny spolupracujúcich útočníkov.

1.1 Bitcoinové adresy

Bitcoinová adresa je reťazec alfanumerických znakov. Jej dĺžka je najčastejšie 34 znakov. Na tieto adresy sa dajú posielať Bitcoiny. Každý používateľ si môže vytvoriť ľubovoľný počet takýchto adries. Z hľadiska anonymity je dokonca odporúčané

pre každý príjem Bitcoinov si vytvoriť novú adresu. Príklad Bitcoinovej adresy: 1JecmBvueZRYLitgSD46Qg6cfB9TksSaSU .

V Bitcoinovom systéme sa využívajú digitálne podpisy na preukázanie vlastníctva Bitcoinov. Bitcoinová adresa je v skutočnosti 160 bitový haš verejného kľúča ECDSA podpisovej schémy. Súkromný kľúč slúži na vytváranie podpisov pri transakciách. Používatelia si teda musia pamätať svoje súkromné kľúče. V prípade straty súkromných kľúčov používatelia prídu o všetky Bitcoiny, ktoré sa nachádzajú na príslušných adresách. Podobne, ak prípadný útočník získa súkromné kľúče, bude mať plnú kontrolu nad Bitcoinmi na príslušných adresách.

Bitcoinové adresy obsahujú v sebe aj kontrolný súčet, aby sa znížila pravdepodobnosť posielania Bitcoinov na zlé adresy (napríklad, ak by niekto omylom urobil preklep niektorého znaku v adrese). Kvôli čitateľnosti, adresy sa skladajú len z alfanumerických znakov a neobsahujú znaky 0, O, l, I, keďže sa tieto vzájomne podobajú.

Vytvorenie adresy prebieha približne nasledovne:

1. Z verejného kľúča sa vytvorí 160 bitový haš: RIPEMD-160(SHA-256(PUB))
2. Pred 160 bitový haš sa pripojí version byte (pre adresy štandardnej hlavnej Bitcoinovej siete je to 0x00).
3. Ako kontrolný súčet sa za číslo získané v 2. kroku pripoja prvé 4 bajty jeho dvojnásobného SHA-256 hašu.
4. Výsledné 25 bajtové číslo je skonvertované na alfanumerický base58 reťazec.

1.2 Transakcie

Pomocou transakcií si používatelia môžu posielať Bitcoiny. V transakcii sa odkazuje na predošlú transakciu, na základe ktorej odosielateľ dostal Bitcoiny. Odosielateľ musí preukázať, že príslušné Bitcoiny z predošlej transakcie naozaj vlastní. Odosielateľ potom zadá množstvo prevádzaných Bitcoinov, a špecifikuje pravidlá, podľa ktorých príjemca bude môcť preukázať nadobudnuté vlastníctvo prevedených Bitcoinov. Najčastejšie sa v pravidlách uvedie adresa, kam sa posielajú Bitcoiny, a príjemca bude musieť preukázať vlastníctvo schopnosťou vytvorenia platného digitálneho podpisu. Údaje uskutočnenej transakcie sa potom rozošlú každému používateľovi. Transakcia sa stane overenou, až keď sa dostane do blokovej reťaze (block chain), ktorá je vlastne verejná distribuovaná databáza transakcií.

1.2.1 Príklad transakcie

```
Vstupná časť transakcie:
Previous tx: f5d8ee39a430901c91a5917b9f2dc19d6d1a0e9cea205b009ca73dd04470b9a6
Index: 0
scriptSig: 304502206e21798a42fae0e854281abd38bacd1aeed3ee3738d9e1446618c4571d10
90db022100e2ac980643b0b82c0e88ffdfec6b64e3e6ba35e7ba5fdd7d5d6cc8d25c6b241501

Výstupná časť transakcie:
Value: 5000000000
scriptPubKey: OP_DUP OP_HASH160 404371705fa9bd789a2fcd52d2c580b65d35549d
OP_EQUALVERIFY OP_CHECKSIG
```

Ako vidíme, transakciu tvorí vstupná časť (skrátene vstup) a výstupná časť (skrátene výstup).

„Previous tx“ odkazuje na predošlú transakciu pomocou hašu predošlej transakcie. Pomocou „Index“ sa určuje koľkáty výstup predošlej transakcie sa má brať ako vstup. Pomocou „scriptSig“ a pomocou „scriptPubKey“ predošlej transakcie sa preukazuje vlastníctvo Bitcoinov.

„Value“ udáva množstvo posielaných Bitcoinov. V našom prípade ide o prevod 5000000000 Satoshi, pričom 1BTC = 10^8Satoshi. Príjemca bude musieť preukázať vlastníctvo podľa pravidiel v „scriptPubKey“. Konkrétne v tomto príklade ide o štandardné posielanie Bitcoinov na Bitcoinovú adresu, kde 160 bitový haš verejného kľúča sa rovná „404371....“.

1.2.2 Typy transakcií

Na preukazovanie vlastníctva Bitcoinov sa využíva vlastný skriptovací jazyk, ktorý je založený na použití zásobníkovej dátovej štruktúry. Do zásobníka sa vloží „scriptSig“ z aktuálnej transakcie, a „scriptPubKey“ z predošlej transakcie. Vlastníctvo je platné vtedy, ak skript vráti kladnú hodnotu. Pomocou tohto skriptovacieho jazyka je možné vytvoriť rôzne zložité podmienky. Momentálne sa však využívajú len 3 základné typy transakcií: Prevod Bitcoinov na Bitcoinovú adresu, Prevod Bitcoinov na IP adresu, Odmenu generujúca transakcia. Nasleduje podrobnejší popis týchto typov transakcií.

Prevod Bitcoinov na Bitcoinovú adresu

```
scriptPubKey: OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG
scriptSig: <sig> <pubKey>
```

Keďže Bitcoinová adresa je len 160 bitový haš verejného kľúča, používateľ, ktorý chce dokázať vlastníctvo Bitcoinov, v „scriptSig“ musí okrem svojho digitálneho podpisu „<sig>“ uviesť aj verejný kľúč „<pubKey>“. Skript skontroluje, či uvedený verejný kľúč skutočne zodpovedá Bitcoinovej adrese „<pubKeyHash>“, t.j. adrese, na ktorú sa prevádzali Bitcoiny. Potom overí pravosť podpisu.

Priebeh kontroly:

Zásobník (pravá položka je vrchná)	Skript	Popis
Prázdny	`<sig> <pubKey> OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG`	„scriptSig“ a „scriptPubKey“ sa spoja
`<sig> <pubKey>`	`OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG`	Konštanty sa pridajú do zásobníka
`<sig> <pubKey> <pubKey>`	`OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG`	Vrchná položka sa zduplikuje
`<sig> <pubKey> <pubHashA>`	`<pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG`	Vypočíta sa 160 bitový haš vrchnej položky
`<sig> <pubKey> <pubHashA> <pubKeyHash>`	`OP_EQUALVERIFY OP_CHECKSIG`	Vloží sa 160 bitový haš (adresa z predošlej transakcie)
`<sig> <pubKey>`	`OP_CHECKSIG`	Porovná sa rovnosť dvoch hašov
`pravda(true)`	Prázdny	Overí sa podpis

Prevod Bitcoinov na IP adresu

```
scriptPubKey: <pubKey> OP_CHECKSIG
scriptSig: <sig>
```

Staršie verzie Bitcoinového klienta umožňujú prevod Bitcoinov na IP adresu. Klient odosielateľa sa pripojí na klient príjemcu na zadanej IP adrese a získa od príjemcu niektorý jeho verejný kľúč. Odosielateľ v „scriptPubKey“ uvedie verejný kľúč príjemcu. Príjemca preukáže vlastníctvo Bitcoinov vytvorením platného digitálneho podpisu k danému verejnému kľúču.

Priebeh kontroly:

Zásobník (pravá položka je vrchná)	Skript	Popis
Prázdny	`<sig> <pubKey> OP_CHECKSIG`	„scriptSig“ a „scriptPubKey“ sa spoja
`<sig> <pubKey>`	`OP_CHECKSIG`	Konštanty sa pridajú do zásobníka
`pravda(true)`	Prázdny	Overí sa podpis

Odmenu generujúca transakcia

Každý blok Bitcoinového systému obsahuje v sebe jednu odmenu generujúcu transakciu. Táto slúži na odmenu baníka, ktorý vytvoril blok.

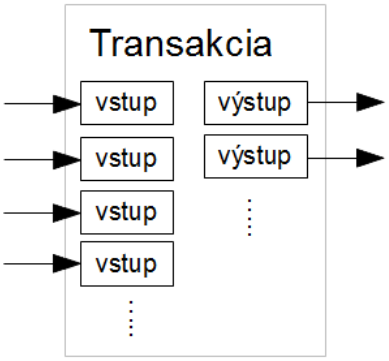
```
Vstupná časť transakcie:
Previous tx:  0000000000000000000000000000000000000000000000000000000000000000
Index: 4294967295
Coinbase: 0402ae091a024a03062f503253482f

Výstupná časť transakcie:
Value: 5000000000
scriptPubKey:
04145f76c60a8a5d57bc86129d4f5a046a8ee2ac03bbfbffd798263a149701cbca0f2d3d0220bca...
OP_CHECKSIG
```

Transakcia má jeden vstup, a namiesto „scriptSig“ má parameter „coinbase“. Hodnota „coinbase“-u nie je dôležitá. Momentálne do „coinbase“-u sa ukladajú informácie napríklad o obtiažnosti generovania. Ukladá sa sem aj „extraNonce“ hodnota, ktorá sa využíva pri generovaní bloku. Výstupné pravidlo v „scriptPubKey“ môže byť ľubovoľné, momentálne sa najčastejšie vytvára výstupné pravidlo podobne ako pri prevode Bitcoinov na IP adresu.

1.2.3 Spájanie a rozdeľovanie Bitcoinov

Teoreticky by bolo možné posielať Bitcoiny jednotlivo po malých častiach, no bolo by to veľmi nepraktické. Aby sa dali spájať a rozdeľovať hodnoty Bitcoinov, transakcie môžu mať viac vstupov aj výstupov. Vo všeobecnosti to znamená, že každá transakcia bude mať jeden alebo viac vstupov, odkiaľ sa berú Bitcoiny, a spravidla 2 výstupy: jeden výstup, kam chceme poslať Bitcoiny, a druhý výstup na vrátenie prebytočných Bitcoinov. Suma výstupných hodnôt nesmie prekročiť sumu vstupných hodnôt. Ak suma výstupných hodnôt je menšia, tak rozdiel sa považuje za transakčný poplatok.



Obr. 1.1: Spájanie a rozdeľovanie Bitcoinov

1.2.4 Transakčné poplatky

Používateľ na základe svojho slobodného rozhodnutia do transakcií môže pridať transakčný poplatok, ktorý potom tvorí dodatočnú odmenu baníka. Ak v nejakej transakcii suma výstupných hodnôt je menšia ako suma vstupných hodnôt, rozdiel je považovaný za transakčný poplatok. Tento transakčný poplatok dostane baník, ktorý vygeneroval blok.

Momentálne nie sú povinné transakčné poplatky. Na druhej strane však baníci nie sú povinní každú transakciu vložiť do bloku. Transakčný poplatok môže slúžiť ako motivácia pre baníkov.

Predpokladá sa, že transakčné poplatky budú neskôr predstavovať hlavnú odmenu baníkov, najmä keď už odmeny novými Bitcoinmi budú nízke.

Poznámka: Najčastejšie baníci používajú pravidlo, že iba prvých 50kB transakcií je v bloku zadarmo, potom sa už požaduje transakčný poplatok. Ak teda používateľ nezadá transakčný poplatok, môže sa stať, že jeho transakciu vložia baníci do nejakého bloku neskôr. Baníci taktiež zvyknú požadovať transakčný poplatok za takú transakciu, ktorá má veľké množstvo dát.

1.2.5 Digitálne podpisy v transakciách

Na dokazovanie vlastníctva Bitcoinov z predošlých transakcií sa pri štandardných typoch transakcií využívajú digitálne podpisy. Pre každý vstupný údaj odosielateľ musí elektronickým podpisom preukázať, že je oprávnený použiť príslušné Bitcoiny. Posledný byte podpisu v skutočnosti už nepochádza z podpisu, ale je to len údaj, ktorým je určený typ podpisovaného hašu. Štandardne ide o haš, ktorý sa vytvára z upravenej kópie transakcie. V tejto upravenej transakcii sú, okrem aktuálneho vstupu, vymazané všetky skripty pre vstupné údaje. Haš, ktorý podpisujeme, teda závisí aj od výstupných údajov transakcie. Prípadný útočník nemôže zmeniť výstupné údaje bez porušenia platnosti podpisu.

Správnosť podpisu v skriptovacom jazyku overuje príkaz `OP_CHECKSIG`. Očakáva 2 parametre zo zásobníka: verejný kľúč a podpis. V prípade, že podpis je korektný, vráti kladnú hodnotu, inak zápornú.

1.3 Bloky

Bloky sú súbory, v ktorých sa uchovávajú transakcie. Ak používateľ spraví transakciu, údaje o transakcii sa rozošlú do celej Bitcoinovej siete. Baníci zbierajú rozoslané transakcie a vkladajú ich do blokov. Bloky majú v sebe odkaz na predchádzajúci blok, a tak bloky vytvárajú jednu dlhú reťaz, ktorú voláme bloková reťaz (block chain). Ak sa vytvorí blok, už sa nikdy neodstráni. Transakcia sa môže vložiť do bloku len ak je platná a ešte sa nenachádza v blokovej reťazi. Transakcia sa stane overenou, ak sa dostane do blokovej reťaze. Čím dlhšia je reťaz blokov, ktorá sa ešte nadviaže na

terajší blok, tým väčšia je miera overenosti terajšieho bloku, a aj transakcií, ktoré sú v ňom.

Bloky v sebe obsahujú aj riešenie ťažkého matematického problému, správnosť riešenia ktorého sa však dá ľahko overiť. Pre každý blok je toto riešenie iné. Baníci hľadajú toto vhodné riešenie, a táto ich činnosť sa nazýva ťažba (mining), keďže úspešný baník je odmenený Bitcoinmi pomocou odmenu generujúcej transakcie. Obtiažnosť problému sa pravidelne aktualizuje, aby sa v priemere vytvorilo 6 blokov za hodinu.

Problém, ktorý musia riešiť baníci, je nájdenie vhodnej Nonce hodnoty, aby haš bloku bol menší než nejaké dané číslo. Toto číslo sa určí na základe nastavenej obtiažnosti. Je jednoduché vypočítať haš hodnotu pre hocijaké dané vstupné dáta. Avšak, podľa dnešného stavu poznania sa predpokladá, že výpočet v opačnom smere, t.j. výpočet možných vstupných dát z výstupných dát, je veľmi zložitý. Podľa dnešného stavu poznania výpočet v opačnom smere sa dá riešiť len postupným skúšaním rôznych vstupných hodnôt pričom sa sleduje, či náhodou nedosiahneme žiadané výstupné dáta. Preto nájdenie vhodnej Nonce hodnoty je z vyššie uvedeného dôvodu veľmi ťažké.

1.3.1 Bloková reťaz (Block chain)

Bloková reťaz je reťaz blokov, t.j. databáza všetkých transakcií. Túto databázu zdieľajú všetci používatelia.

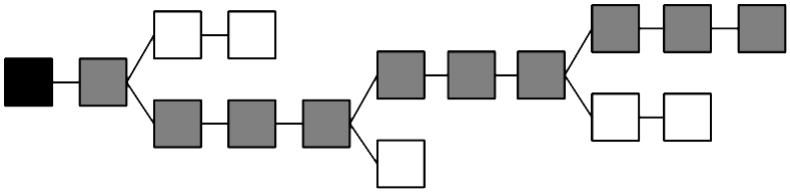
Každý blok obsahuje odkaz na predchádzajúci blok a tento odkaz je vlastne hašom predchádzajúceho bloku. Vďaka tomuto bloky tvoria jednu dlhú reťaz. Blok na začiatku reťaze sa volá blok Genesis (počiatočný blok, Genesis block).

Je garantované, že bloky sa pripájajú na seba podľa času vytvorenia, keďže inak by sa nedal vytvoriť odkaz na predchádzajúci blok.

Poctiví baníci vždy napájajú nové bloky len na vrchný blok, t.j. na blok na konci najdlhšej vetvy blokovej reťaze. Môže sa však stať, že napríklad dvom baníkom sa približne naraz podarí vygenerovať blok. V takomto prípade sa reťaz rozdelí. Baníci budú napájať nové bloky na ten blok, ktorý majú k dispozícií skôr. Akonáhle sa jedna vetva predĺži, tá sa bude považovať za platnú vetvu. Bloky v kratšej vedľajšej vetve sa označujú názvom „osirelé bloky“ (orphan blocks). V osirelých blokoch všetky transakcie strácajú svoju overenosť (pozrite popis overenosti nižšie) a všetky transakcie, okrem odmenu generujúcich transakcií, sa budú znova pridávať do blokov. Z tohto dôvodu odmenu generujúce transakcie musia mať overenosť až 100, a až po takomto overení sa môžu jej Bitcoiny použiť. Potom je už veľmi nepravdepodobné, že by sa dostali do osirelej vetvy.

Dĺžka blokovej reťaze sa neráta podľa počtu blokov, ale podľa obtiažnosti jednotlivých blokov (pozrite popis obtiažnosti nižšie). Toto zabraňuje tomu, aby niekto vytvoril veľa blokov s nízkou obtiažnosťou a tak zmenil hlavnú vetvu.

Príklad blokovej reťaze je na obr. 1.2 . Čierny štvorec označuje Genesis blok, sivé štvorce označujú bloky v hlavnej vetve, biele štvorce označujú osirelé bloky.



Obr. 1.2: Bloková reťaz

1.3.2 Overenosť transakcie

Pri transakciách je veľmi dôležité sledovať ich overenosť. Transakciu môžeme považovať s veľkou pravdepodobnosťou za overenú, keď už istý počet nadväzujúcich blokov v blokovej reťazi potvrdzuje platnosť transakcie. Práve vytvorená nová transakcia, ktorá ešte nie je zaradená do žiadneho bloku, má nulovú overenosť. Ak táto transakcia je už zaradená do bloku v hlavnej vetve blokovej reťaze, transakcia má overenosť 1. Každý ďalší nadväzujúci blok v blokovej reťazi zvyšuje overenosť transakcie o 1. Ak napríklad transakcia je zaradená v hlavnej blokovej reťazi v bloku a v hlavnej vetve existujú za týmto blokom napríklad už 2 ďalšie bloky, tak sledovaná transakcia má overenosť 3.

1.3.3 Hašovanie blokov

Pri generovaní blokov sa len jednoducho opakovane vytvára haš bloku, a kontroluje sa, či haš je dostatočne malý. Ak áno, tak sme práve vygenerovali platný blok. Hašuje sa len hlavička bloku, ktorá sa skladá z nasledovných položiek:

Položka	Popis	Veľkosť (bajty)
Verzia	Číslo verzie bloku	4
Predchádzajúci haš	Haš predchádzajúceho bloku v blokovej reťazi	32
Koreňový haš	Koreň hašového stromu, ktorý sa vytvára z hašov transakcií	32
Časová pečiatka	Aktuálna časová pečiatka	4
“Bits”	Cieľová (Target) hodnota v kompaktnom formáte, podľa ktorej sa určuje aktuálna obtiažnosť	4
Nonce	32 bitové číslo, postupne sa zvyšuje pri hľadaní vhodného hašu	4

Telo bloku obsahuje transakcie, ale v hlavičke tieto transakcie sú reprezentované iba ako koreň hašového stromu, ktorý bol vytvorený z transakcií. To znamená, že hlavička bloku je stále rovnako dlhá a teda na rýchlosť hašovania nemá počet transakcií v bloku žiadny vplyv.

Nonce hodnota najprv má hodnotu 0, a potom sa postupne po každom pokuse zvyšuje.

Ak hodnota pretečie, tak sa upraví „extraNonce“ hodnota v odmenu generujúcej transakcii. Akákoľvek zmena v hlavičke spôsobuje nepredvídateľnú zmenu v haš hodnote.

Ak by každý rátal haš pre tabuľku s rovnakou hlavičkou, tak pravdepodobne by vyhrával stále ten, ktorý má najväčšiu výpočtovú silu. Ale je skoro nemožné, aby dvaja baníci pracovali s rovnakou hlavičkou, keďže odmenu generujúca transakcia bude pre každého používateľa iná, teda aj „koreňový haš“ bude iný.

Bitcoinový systém ráta haš hlavičky nasledovne: SHA-256(SHA-256(hlavička)).

1.3.4 Dôkaz práce baníkov

V časti „Bloky“ je vysvetlené, že pri súčasnom stave poznania je nájdenie vhodnej Nonce hodnoty veľmi obtiažne. Treba spraviť veľa pokusov, kým sa nájde vhodná Nonce hodnota. Z tohto vyplýva, že ak niekto dodá platný blok, dotyčný musel vynaložiť veľkú prácu. Ak niekto chce zmeniť blok, musí zase vykonať podobne zložitú prácu. Ak by niekto chcel zmeniť blok, ktorý už je hlbšie v blokovej reťazi, musel by zmeniť aj všetky bloky, ktoré sú nad týmto blokom.

Celý vývoj blokovej reťaze teda závisí od toho, čo chce prevažná časť výpočtovej sily baníkov. Ak prevažná časť výpočtovej sily je ovládaná poctivými baníkmi, poctivá vetva prerastie všetky ostatné konkurenčné vetvy blokovej reťaze.

1.3.5 Cieľová (Target) hodnota

Cieľ je 256 bitová hodnota zdieľaná všetkými Bitcoinovými používateľmi. Hodnota Cieľa určuje, akú maximálnu hodnotu môže mať haš bloku, aby bol akceptovaný ako platný. Čím menšia je táto hodnota, tým ťažšie sa dá vygenerovať blok. Pri maximálnej hodnote cieľa je vygenerovanie bloku najjednoduchšie.

Keďže výpočtová sila baníkov sa môže meniť, Cieľová hodnota sa vždy po 2016 blokoch aktualizuje. Bitcoinový klient overí, za aký čas sa vygenerovali tieto bloky, a upraví Cieľovú hodnotu, aby sa v priemere vygeneroval jeden blok za 10 minút. Vygenerovanie 2016 blokov teda ideálne trvá 14 dní.

Maximálna hodnota Cieľa je :
`0x00000000ffff0000000000000000000000000000000000000000000000000000`

1.3.6 Obtiažnosť

Obtiažnosť je hodnota, ktorá vyjadruje, koľkokrát je ťažšie vygenerovať blok v porovnaní s najľahším prípadom. Obtiažnosť sa dá určiť nasledovne:

$$\text{obtiaznost} = \text{maximalna\_hodnota\_ciela} / \text{aktualna\_hodnota\_ciela}$$

Ak označíme obtiažnosť písmenom D, tak pravdepodobnosť, že nejaký vygenerovaný haš bude vhodný, je zhruba $1/(2^{32}D)$

1.3.7 Časová pečiatka bloku

Každý blok obsahuje Unixovú časovú pečiatku. Táto slúži aj ako ďalší zdroj náhodnosti pre haš bloku. Časová pečiatka sa využíva aj pri overovaní platnosti bloku, aby prípadný útočník mohol ťažšie modifikovať blokovú reťaz.

Časová pečiatka je platná len ak je väčšia než medián časových pečiatok predošlých 11 blokov a musí byť menšia ako aktuálny sieťový čas + 2 hodiny. Sieťový čas je popísaný v časti Sieťový čas.

Bitcoinový systém používa bezznamienkový integer pre uchovávanie časovej pečiatky v bloku, preto problém roku 2038 je posunutý o 68 rokov.

1.3.8 Koreňový haš transakcií

Hašový strom

Hašový strom je dátová štruktúra, ktorá obsahuje do stromovitej štruktúry organizované súhrnné informácie o nejakej väčšej skupine dát. Skupina dát je rozdelená na menšie časti a listy stromu obsahujú haše týchto častí. Vnútorné uzly stromu obsahujú haše ich detí. Haš strom je väčšinou binárny strom, ale uzly stromu môžu mať aj viac detí. Haš v koreni stromu sa volá koreňový haš.

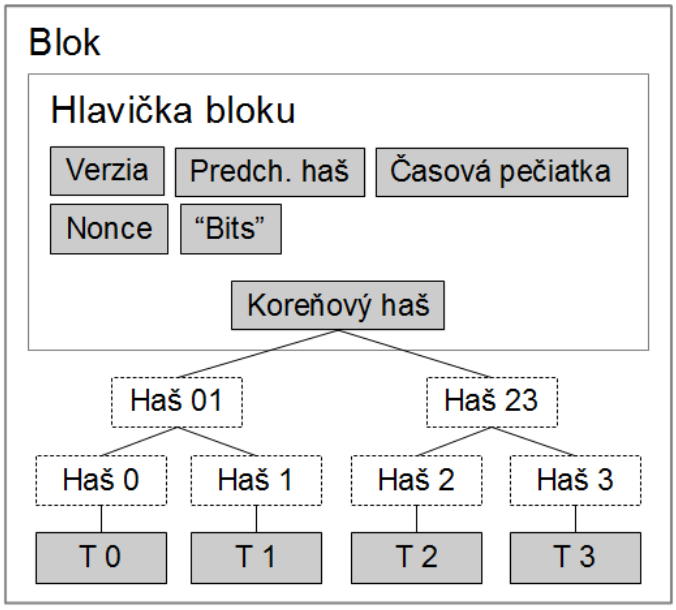
Hašové stromy sa vo všeobecnosti využívajú na overenie skupiny dát, napríklad pri prenášaní dát medzi počítačmi. Haš stromy sa využívajú napríklad pri peer-to-peer komunikácií, aby sa overil, či skupina dát je nepoškodená, nezmenená a či náhodou uzly v sieti sa nám nepokúšajú poslať falošné dáta.

Výhodou haš stromov je, že stačí aby len koreňový haš bol získaný z dôveryhodného zdroja. Keď už je koreňový haš k dispozícii, strom sa môže stiahnuť z akéhokoľvek zdroja. Potom sa skontroluje, či získané údaje majú naozaj taký koreňový haš, aký sa získal z dôveryhodného zdroja. Ďalšou výhodou haš stromov je, že na skontrolovanie dát v niektorej časti skupiny dát, netreba mať k dispozícii celú skupinu dát a ani celý hašový strom. Stačí, ak máme k dispozícii príslušnú vetvu hašového stromu. Ďalšia výhoda je, že ak niektorá časť skupiny údajov je poškodená, netreba stiahnuť znova celú skupinu údajov, stačí len nahradiť poškodenú časť.

Haš strom transakcií

V hlavičke bloku sa nachádza koreňový haš z hašového stromu, ktorý je vytvorený z transakcií. Vďaka vlastnostiam haš stromov sa koreňový haš dá využiť na rôzne optimalizácie a na zjednodušenie niektorých operácií. Napríklad sa dá využiť na zníže-

nie potrebného diskového priestoru na uchovávanie blokov, alebo na jednoduchšie overovanie, či nejaká transakcia sa nachádza v nejakom bloku.



Obr. 1.3: Štruktúra bloku a koreňový haš transakcií

1.4 Bitcoinová sieť

Bitcoinový systém na svoju prevádzku využíva počítačovú sieť. Pomocou TCP komunikácie uzly Bitcoinového systému si navzájom posielajú jednoduché správy a šíria údaje o nových transakciách a blokoch.

Pri naštartovaní Bitcoinového klienta, ten sa pokúsi pripojiť sa k ostatným uzlom. Na získanie IP adries existuje viac spôsobov:

- klient sa náhodne pokúsi pripojiť sa na adresy, ku ktorým už bol v minulosti pripojený
- nejaké adresy sú zakódované v klientskom programe
- adresy sa hľadajú aj na základe IP adries nejakých doménových mien
- v dávnejších verziách klienta sa využíval aj IRC (Internet Relay Chat)

Klient rozošle haš posledného známeho bloku. Ak existujú už novšie bloky, tak pripojené uzly ho na to upozornia a klient si môže postupne stiahnuť z týchto uzlov nové bloky. Uzly si pravidelne posielajú aj prázdne správy, aby TCP spojenie zostalo otvorené.

1.4.1 Sieťový čas

Po vytvorení pripojenia si klient získa aktuálny UTC čas pripojených uzlov a ich medián bude určovať sieťový čas. Sieťový čas sa však nenastaví takýmto spôsobom, ak je rozdiel medzi týmto mediánom a časom klienta väčší než 70 minút. V takomto prípade sa sieťový čas nastaví podľa aktuálneho času klienta.

1.4.2 Šírenie informácií

Keď niektorý Bitcoinový uzol spraví transakciu, tak rozošle správu všetkým pripojeným uzlom. Tieto uzly si potom údaje o transakcii od neho stiahnu. Ak považujú transakciu za korektnú, aj oni rozošlú údaje o príslušnej transakcii uzlom, s ktorými majú spojenie. Uzly stiahnu a rozošlú len takú transakciu, o ktorej ešte nevedeli. Ak sa transakcia nedostane do bloku, môže sa stať, že sa na danú transakciu zabudne. Odosielateľ a prijímateľ transakcie však v takomto prípade iniciuje transakciu znova.

Baníci zbierajú nové transakcie a vkladajú ich do blokov. Ak sa im podarí vytvoriť blok, tak ho rozošlú a blok sa bude šíriť podobným spôsobom, ako sa to v predošlom odseku popísalo pri transakciách.

Uzly si taktiež pravidelne rozosielajú a šíria svoje IP adresy.

Môže sa stať, že rozosielané transakcie sa nedostanú ku každému uzlu. Ak sa však dostali do väčšieho počtu uzlov, pravdepodobne sa čoskoro dostanú do bloku. Aj šírenie blokov je odolné voči výpadkom. Ak uzol nedostane informáciu o nejakom bloku, tak pri nasledujúcom bloku, ktorý dostane, si všimne, že mu chýbajú nejaké bloky a dodatočne si ich stiahne.

1.5 Možnosti optimalizácie

Vďaka tomu, že v hlavičke bloku sú transakcie reprezentované koreňovým hašom, môžeme niektoré operácie optimalizovať. Ide napríklad o redukovanie diskového priestoru a zjednodušenie verifikácie platieb.

1.5.1 Redukovanie diskového priestoru

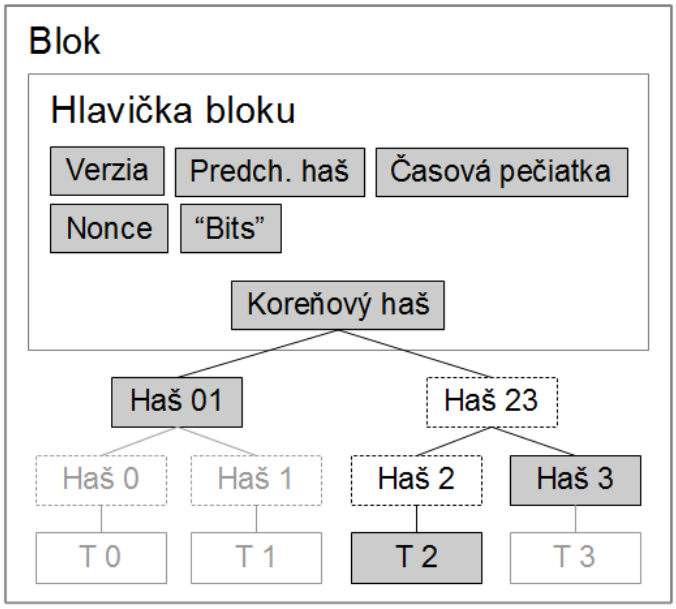
Bloková reťaz neustále rastie, keďže sa stále vytvárajú nové a nové bloky. Celá bloková reťaz už zaberá viac ako 1,5 GB. Aby si nemusel každý používateľ uchovávať také množstvo dát, treba sa nejakých nepotrebných údajov zbaviť.

Keď nejaká konkrétna transakcia Bitcoinu je už dostatočne hlboko v blokovej reťazi, existuje len veľmi malá pravdepodobnosť toho, že sa dostane do osiroteného bloku. Preto všetky transakcie, ktoré sú pred touto konkrétnou transakciou, a ktorých všetky výstupy boli už použité taktiež hlboko v reťazi, sa môžu zmazať, a to v záujme šetrenia diskového priestoru. Aby sme nenarušili koreňový haš bloku, odstrán-

ime len nepotrebné podstromy hašového stromu. Korene odstránených podstromov ponecháme, keďže ich hodnoty potrebujeme pri výpočte hašov.

Táto funkčnosť momentálne ešte nie je implementovaná do Bitcoinových klientov.

Na obr. 1.4 je znázornený príklad redukovania diskového priestoru. Transakcie 0, 1 a 3 boli vymazané. Odstránené transakcie sa nahradili hašmi Haš 01 a Haš 3.



Obr. 1.4: Redukovanie diskového priestoru

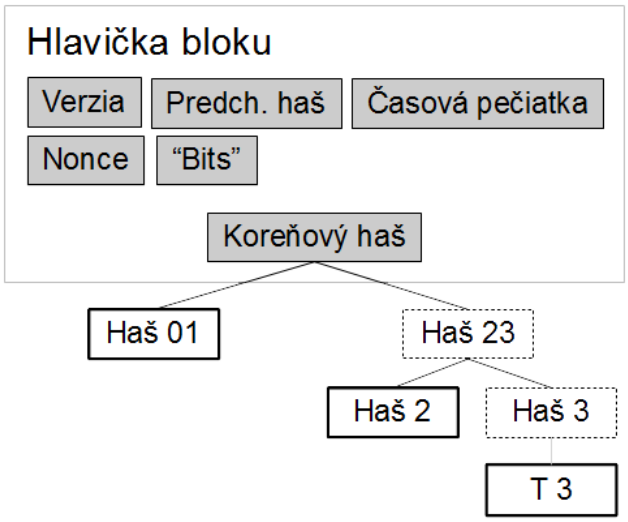
1.5.2 Zjednodušená verifikácia platieb

Aby sme overili, či nejaká transakcia je uložená v nejakom bloku, nepotrebujeme veľa údajov. Stačí ak používateľ má k dispozícii hlavičky blokov hlavnej blokovej reťaze. Používateľ si potom požiada nejakého iného používateľa o informáciu o tom, v ktorom bloku sa u neho nachádza transakcia. Taktiež si vyžiada a stiahne potrebnú vetvu hašového stromu. Na základe tejto vetvy a koreňového hašu v bloku dokáže overiť, či transakcia sa skutočne nachádza v danom bloku. Používateľ nevie priamo overiť platnosť transakcie, ale skutočnosť, že transakcia je v bloku, znamená, že Bitcoinová sieť akceptovala transakciu a každý ďalší napojený blok ďalej potvrdzuje túto skutočnosť.

Takáto zjednodušená verifikácia je spoľahlivá len vtedy, keď poctiví používatelia majú pod kontrolou väčšinu výpočtovej sily. Ak by prípadný útočník bol silnejší, tak táto verifikácia by už nebola spoľahlivá. Útočník by mohol podstrčiť používateľovi falošné údaje.

Pre tých používateľov, ktorí vyžadujú vyššiu bezpečnosť a rýchlejšie overenie, sa odporúča použiť štandardnú verifikáciu.

Na obr. 1.5 je znázornená zjednodušená verifikácia platieb. Na overenie, či transakcia 3 je skutočne v bloku, postačí Haš 01 a Haš 2.



Obr. 1.5: Zjednodušená verifikácia platieb

Kapitola 2

Anonymita

Každý používateľ Bitcoinového systému môže vlastniť ľubovoľne veľa Bitcoinových adries. Tieto adresy sú len náhodné reťazce a priamo z týchto adries sa nedá zistiť identita ich vlastníkov. Tento fakt mnohých ľudí priviedla k mylnej myšlienke, že Bitcoinový platobný systém ponúka dobrú anonymitu.

Pri návrhu systému sa anonymita nebrala ako hlavná priorita. Všetky transakcie v Bitcoinovom systéme sú uložené v blokovej reťazi, ktorá je verejne prístupná komukoľvek. Pomocou blokovej reťaze sa dá presne určiť tok Bitcoinov medzi adresami. Už vieme, že tieto adresy sú len náhodné reťazce. Ak sa však niekomu podarí identifikovať vlastníka niektorej adresy, dá sa to použiť ako štartovací bod na postupné vypátranie všetkých vlastníkov adries z minulých aj budúcich transakcií. Každý predsa pravdepodobne vie, že od koho dostal a komu posielal Bitcoiny. Ak používateľ chce sťažiť takéto zisťovanie identity, mal by vytvoriť vždy novú adresu pre príjem transakcie.

Uvažujme nasledovný jednoduchý príklad: Používateľ X prevádzkuje zmenáreň Bitcoinov a taktiež aj jeden obchod, kde ponúka možnosť platiť Bitcoinmi. Používateľ Y nakúpi Bitcoiny pomocou zmenárne X a potom tieto Bitcoiny použije na nákup niečoho v obchode X. Keďže používateľ Y platil z adresy na ktorú mu používateľ X poslal Bitcoiny, používateľ X vie, že v obchode nakupoval používateľ Y.

Ďalší príklad: Predpokladajme, že niekto je podvedený a zverejní Bitcoinovú adresu podvodníka. Keďže databáza transakcií je verejná, každý môže vidieť, na aké adresy sa posielajú Bitcoiny z adresy podvodníka. Aj tieto adresy môžu byť zverejnené. Takto sa môže pokračovať ďalej a všetky Bitcoiny na týchto adresách sa budú považovať za špinavé. Ak poctivý používateľ si všimne, že aj jeho adresa sa dostala na zoznam, môže zverejniť, že od koho dostal tie špinavé Bitcoiny. Môžu začať zisťovať od tohto používateľa, od koho on dostal špinavé Bitcoiny. Takto sa pokračuje, až kým sa dostanú k podvodníkovi.

Ďalší príklad: Používateľ pošle Bitcoiny z adresy, ktorá je napríklad zverejnená v jeho profile na nejakom online fóre. Jednoduchým hľadaním (napr. pomocou Google) sa dá zistiť, komu patrí daná adresa.

Identita používateľa je odhaliteľná teda aj na základe analýzy transakcií, ktoré prebehli pred tým, než sa k danému používateľovi tie Bitcoiny dostali, ale aj na základe

analýzy transakcií, ktoré sa uskutočnia neskôr.

2.1 Ďalšie možnosti analýzy

Existujú rôzne spôsoby analýzy používateľov aj transakcií, ktoré sa medzi nimi uskutočnili. V ďalších častiach nasleduje ich popis. Tieto spôsoby analýzy boli uvedené v literatúre [16]. Autori v tomto článku analyzovali anonymitu Bitcoinového systému a túto analýzu ilustrovali na skutočnom príklade veľkej krádeže Bitcoinov.

2.1.1 Graf transakcií

Z blokovej reťaze sa dá vytvoriť graf transakcií. Každý vrchol grafu označuje jednu transakciu. Hrany grafu sú orientované a ak dva vrcholy A a B sú spojené orientovanou hranou (A→B), znamená to, že niektorý výstup transakcie A sa použil ako vstup transakcie B. Hrany môžeme ohodnotiť podľa hodnoty prenášaných Bitcoinov. Graf neobsahuje viacnásobné hrany, ani cykly. Ide o orientovaný acyklický graf, keďže výstup transakcie nemôže byť vstupom tej istej transakcie ani priamo ani nepriamo.

2.1.2 Graf používateľov

Vrcholy grafu používateľov reprezentujú používateľov Bitcoinového systému a orientovaná hrana znamená prevod Bitcoinov od jedného používateľa k druhému. Ku každej hrane je priradená hodnota posielaných Bitcoinov a časový údaj prevodu.

Keďže používatelia môžu mať veľa adries a adresy neobsahujú žiadnu informáciu o vlastníkovi, graf používateľov sa nedá len tak jednoducho vyrobiť. Môžeme však vyrobiť približný graf používateľov na základe nasledovného faktu: Ak používateľ chce poslať väčšiu sumu Bitcoinov, ale daný používateľ má k dispozícii adresy, na ktorých sú len malé sumy, často vytvorí transakciu, ktorá má viac vstupov. Viacvstupové transakcie teda nám hovoria, že vstupné adresy s veľkou pravdepodobnosťou patria rovnakému používateľovi.

Vytvoríme si teda najprv zjednodušený graf používateľov, kde každý vrchol grafu bude reprezentovať Bitcoinovú adresu. Adresy, ktoré sú spolu vo vstupnej časti nejakej transakcie, spojíme neorientovanou hranou.

Sledujme len vrcholy a neorientované hrany grafu. Každý komponent grafu bude predstavovať používateľa a vrcholy v tom komponente sú jeho Bitcoinové adresy.

Samozrejme získaný graf nie je presný. Najväčší problém je, že používateľ pravdepodobne má aj také adresy, ktoré neboli v transakciách „spájané“ s inými adresami.

Graf používateľov môže obsahovať viacnásobné hrany, slučky a cykly.

2.1.3 Analýza používateľov

Z grafu používateľov môžeme získať informácie o používateľoch mnohými spôsobmi.

Aby sme napríklad našli mimoriadne aktívnych používateľov po nejakej udalosti, môžeme skúmať stupne vrcholov a hľadať nejaké nezvyčajné hodnoty. Z grafu môžeme zistiť, s akými inými používateľmi spolupracuje sledovaný používateľ. Ďalej môžeme analyzovať, aký bol tok Bitcoinov medzi skupinami ľudí v nejakom čase. Môžeme určiť koľko Bitcoinov vlastní každý používateľ.

Údaje môžeme zbierať nielen z Bitcoinovej siete. Obchody, zmenárne, e-peňaženky, a iné služby, ktoré pracujú s Bitcoinami, majú často prístup aj k iným dátam: napríklad emailové adresy, dodacie adresy, čísla kreditných kariet a bankových účtov, IP adresy, atď. Ak by tieto informácie boli verejne dostupné, alebo prístupné pre rôzne organizácie, tak aj anonymita všetkých používateľov vo všetkých súvisiacich transakciách by bola ohrozená.

Ako ilustráciu môžeme uviesť službu „Bitcoin Faucet“. Ide o webstránku, kde každý môže dostať malé množstvo Bitcoinov zadarmo. Aby služba nebola zneužitá, používateľ musí zadať svoju emailovú adresu a kontroluje sa aj jeho IP adresa. Po odoslaní Bitcoinov sa na webstránke zverejní časť emailovej adresy a objaví sa celá IP adresa používateľa. Služba často rozosiela Bitcoiny používateľom jednotlivo. Na základe publikovaných údajov sa dá spraviť jednoznačné spojenie medzi Bitcoinovou adresou a IP adresou používateľa.

Veľa ľudí si uvedie svoju Bitcoinovú adresu napríklad v online fóre, Twitter správach, rôznych zoznamoch a pod. Bitcoinové adresy sú reťaze alfanumerických znakov, ktoré webové vyhľadávače (Google a pod.) celkom dobre zaindexujú. Anonymita používateľa sa môže teda odhaliť aj na základe jednoduchého vyhľadávania Bitcoinových adries.

V Bitcoinových zmenárňach sú často objednávky Bitcoinov verejne publikované. Hodnota objednaných Bitcoinov je často určená v nejakej peňažnej mene (napr. USD, EUR) a potom je prerátaná na hodnotu v Bitcoinoch. Príslušná Bitcoinová hodnota má preto často veľmi veľa desatinných miest. Dá sa potom nájsť transakcia s rovnakou sumou, a tak sa zistí väzba na zmenáreň.

Môžu existovať aj oveľa zložitejšie útoky na anonymitu. Útočník sa nemusí obmedziť len na pasívnu analýzu, môže sa aj aktívne zapájať, rozosielať Bitcoiny a sledovať ich pohyb a pod.

2.2 Ako zostať v anonymite

Iba z tej adresy môžeme poslať ďalej Bitcoiny, na ktorú nám boli poslané. Ak niekto má viac adries, tak samozrejme môže sa rozhodnúť, z ktorej adresy by chcel poslať Bitcoiny. Ak používateľ posiela Bitcoiny z takej adresy, o ktorej vie, že neprezradí o ňom žiadnu informáciu, alebo ak by posielal Bitcoiny, ktoré dostal ako odmenu baníka, tak anonymita by bola zhruba zabezpečená. Musí sa však zvážiť, či sa nedá

ľahko dopracovať k identite nedávnych majiteľov a najbližších budúcich majiteľov posielaných Bitcoinov. Ak áno, tak to by mohlo narušiť aj anonymitu daného používateľa. Poznámka: V súčasnosti originálny Bitcoinový klient pri prevode Bitcoinov neumožňuje používateľovi vybrať adresu, z ktorej sa má daný prevod uskutočniť.

Na anonymizáciu svojich Bitcoinov, používateľ môže skúsiť nasledovný postup: Používateľ pošle svoje Bitcoiny do e-peňaženky (popis je v nasledujúcej časti), a potom postupne prevedie ich späť na svoje nové adresy. Ak e-peňaženka má relatívne veľa Bitcoinov, tak je veľká pravdepodobnosť toho, že používateľ dostane späť úplne iné Bitcoiny. Aby prípadný útočník mohol ďalej sledovať používateľa, musí mať prístup k záznamom e-peňaženky. Aby sa používateľ ešte viac anonymizoval, môže poslať svoje Bitcoiny cez dve e-peňaženky a až potom ich získať späť. Samozrejme, ak prevádzkovateľ e-peňaženky poskytne prípadnému útočníkovi záznamy systému, tak tento spôsob anonymizácie nie je vhodný.

2.3 E-peňaženka (EWallet)

E-peňaženka je online služba prevádzkovaná väčšinou súkromnými organizáciami. Umožňuje ukladanie a manažovanie Bitcoinov na online účte.

Výhody e-peňaženky sú:

- Môže zlepšiť anonymitu používateľov vo vzťahu k iným používateľom
- Účet v e-peňaženke môže používateľ celkom rýchlo vytvoriť
- Používateľ sa nemusí starať o bezpečnosť svojej peňaženky (peňaženka je súbor, ktorý obsahuje súkromné kľúče patriace k Bitcoinovým adresám používateľa)
- Výber Bitcoinov z e-peňaženky môže používateľ nasmerovať na ľubovoľnú adresu, čiže výber z e-peňaženkového účtu na cudziu adresu je rovnocenný so štandardným prevodom Bitcoinov
- Prevody v rámci e-peňaženky sa často spracujú okamžite, netreba čakať na potvrdenia blokmi

Používateľ však musí byť opatrný pri práci s e-peňaženkami. Prevádzkovateľ získa plnú kontrolu nad jeho Bitcoinmi. Na rozdiel, ak používateľ používa štandardný Bitcoinový program, tak iba on má kontrolu nad svojimi Bitcoinmi.

Ďalšie nevýhody E-peňaženky:

- Používateľ má menšiu anonymitu voči prevádzkovateľovi služby
- Ak sa vykoná platba z e-peňaženky, tak adresa odosielateľa v transakcii bude adresa prevádzkovateľa, keďže používatelia e-peňaženky často nemajú skutočnú adresu. Preto, ak by príjemca vrátil Bitcoiny, ich vlastník by nebol identifikovaný a nemusel by preto dostať späť svoje Bitcoiny.

- Adresa používateľa v e-peňaženke, ktorá bola vygenerovaná na príjem Bitcoinov, môže raz stratiť svoju platnosť.
- Neexistuje žiadna garancia, že prevádzkovateľ skutočne má toľko Bitcoinov, koľko nám ukazuje.
- Prevod Bitcoinov z e-peňaženky na adresu, ktorú taktiež má pod kontrolou prevádzkovateľ, často prebehnú interne a nebudú spracované v blokovej reťazi.
- Často sa dá bez väčších nákladov a okamžite zistiť, či niekto má účet v niektorej konkrétnej E-peňaženke, a to nasledovne: Prípadný útočník po vytvorení vlastného konta v E-peňaženke, môže previesť nejakú malú sumu na zisťovanú adresu, a keďže na „vnútorné“ adresy dovoľuje prevádzkovateľ poslať aj menšie sumy, úspešný prevod malej sumy dokazuje, že prijímateľ je v danej E-peňaženke.
- Môže dôjsť k narušeniu bezpečnosti prevádzkovateľa a Bitcoiny sa môžu stratiť, alebo môže ich niekto ukradnúť. Keďže Bitcoinové transakcie sa nedajú zmeniť, ak niekto ukradne Bitcoiny prevádzkovateľovi E-peňaženky, často neexistuje žiadna možnosť na obnovenie stavu pred krádežou.

2.4 Miešacia služba (Mixing service)

Miešacia služba za malý poplatok „zmieša“ Bitcoiny viacerých používateľov. Takto sa sťaží vystopovanie originálneho zdroja Bitcoinov. Miešanie pomôže pri ochrane súkromia, ale na druhej strane tento proces je využiteľný aj na nelegálnu činnosť, napríklad na pranie špinavých peňazí.

Kapitola 3

Bezpečnosť

V tejto kapitole sa skúma zraniteľnosť Bitcoinového systému a zraniteľnosť originálneho Bitcoinového klienta.

3.1 Možnosti ohrozenia Bitcoinového Systému

3.1.1 Vystopovanie histórie Bitcoinov

Bloková reťaz uchováva všetky transakcie a je prístupná komukoľvek. Ktokoľvek si teda môže pozrieť všetky transakcie. Tento problém je popísaný v časti „Anonymita“.

3.1.2 Falošné uzly v sieti

Prípadný útočník vcelku jednoducho môže naplniť sieť falošnými uzlami, nad ktorými má plnú kontrolu. Potom sa môže stať, že používateľ sa napojí len na falošné uzly.

Útočník môže tento stav rôznymi spôsobmi zneužiť. Príklady možného zneužitia:

- Útočník môže zablokovať prenos dát napríklad o transakciách a blokoch smerom k obeti aj v opačnom smere, a tak odpojiť obeť od Bitcoinovej siete.
- Útočník môže prepúšťať obeti iba tie bloky, ktoré vytvoril on sám. Obeť sa tak môže dostať do oddelenej siete a bude pracovať s inou blokovou reťazou než ostatní používatelia. Obeť v tejto oddelenej sieti bude ohrozený útokmi typu double-spending (dvojnásobné použitie rovnakých Bitcoinov falošným spôsobom v rôznych transakciách). Kontrola pravidelnej tvorby blokov môže pomôcť odhaliť takéto útoky.
- Ak používateľ akceptuje transakcie s nulovou overenosťou, útočník môže jednoduchým odfiltrovaním niektorých transakcií vykonať útok typu double-spending.

- Ak Bitcoinový klient používa také šifrovanie/anonymizáciu Bitcoinovej komunikácie (napr. pomocou systému Tor), ktorá má nízku latenciu, útočník dokáže jednoducho zistiť, či používateľ spravil nejakú transakciu. Útočníkovi stačí, aby používateľ bol pripojený na niektoré útočníkove falošné uzly a aby útočník dokázal sledovať, či k používateľovi prichádzajú alebo od neho odchádzajú nejaké dáta. Ak útočník zistí, že k používateľovi žiadne údaje neprichádzajú, ale útočníkov falošný uzol dostal od používateľa transakciu, tak túto transakciu vytvoril daný používateľ. Týmto spôsobom útočník zistí aj to, že všetky vstupné adresy transakcie patria danému používateľovi.

Bitcoinový systém sťažuje vyššie uvádzané útoky tým, že vytvára maximálne len jedno pripojenie na IP adresy, ktorých prvé 2 bajty sa zhodujú. Bitcoinový systém nelimituje týmto spôsobom prichádzajúce spojenia, ale toto by mohlo spôsobiť ohrozenie iba v 4. vyššie uvedenom prípade. Väčšinou by to však nemalo vadiť, lebo ak používateľ využíva anonymizačné služby, tak pravdepodobne nemôže prijímať spojenia.

3.1.3 Odpočúvanie paketov

Ak útočník vidí prichádzajúce a odchádzajúce pakety nejakého používateľa, vidí, kedy tento používateľ odosiela ostatným používateľom takú transakciu, ktorú neprijal. Z tohto útočník vie, že táto transakcia nepochádza od iných používateľov, ale vytvoril ju daný používateľ. Šifrovanie medzi uzlami siete sťažuje útočníkovi takéto zisťovanie. Pri šifrovaní však útočník môže postupovať metódou vytvorenia falošných uzlov, ktorá je popísaná vyššie.

3.1.4 Útok posunutím sieťového času

V Bitcoinovom systéme podmienky akceptovania blokov súvisia aj s časom. Každý blok obsahuje Unixovú časovú pečiatku, ktorej podrobnejší popis je v časti „Časová pečiatka bloku“. Sieťový čas je popísaný v časti „Sieťový čas“. Spôsob určenia sieťového času je možné zneužiť [21].

Príprava útoku posunutím sieťového času

Predpokladom útoku je, aby útočník mal vytvorené množstvo falošných Bitcoinových uzlov, a dokázal vytvoriť z nich spojenie na obeť a tiež na ostatné uzly. Ak útočník sa pripojí na nejaký uzol, tak daný klient získa aktuálny UTC čas útočníka a určí sieťový čas ako medián pripojených uzlov. Útočníkov čas takto môže ovplyvniť sieťový čas uzla, na ktorý sa pripojil. Útočník si nastaví vždy nejaké vhodné časy pre svoj klient a postupne sa pripája na rôzne uzly a takto zvýši alebo zníži sieťový čas daných uzlov.

Predpokladajme, že skutočný čas má hodnotu x. Útočník vyššie popísaným spôsobom nastaví sieťový čas obete na hodnotu $x - 70$ minút a zmení čas dôležitých iných uzlov na hodnotu $x + 70$ minút. Dôležitými sú uzly baníkov, keďže oni vytvárajú bloky.

Okrem obete útočníkovi stačí zmeniť čas pre väčšinu baníkov. Časový rozdiel medzi obeťou a baníkmi bude teda 140 minút.

Útočník potom vytvorí klamlivý blok, ktorému nastaví časovú pečiatku $x+190$ minút.

Uzly odmietnu také bloky, ktorých časová pečiatka má o 120 minút väčšiu hodnotu než sieťový čas. Bloky, ktorých časová pečiatka je menšia ako medián časových pečiatok posledných 11 blokov, sú tiež odmietnuté.

Obeť odmietne útočníkom vytvorený klamlivý blok, lebo z pohľadu obete časová pečiatka bloku obsahuje až o 260 minút väčší časový údaj než aký je sieťový čas obete. Uzly so sieťovým časom $x + 70$ minút akceptujú útočníkov klamlivý blok, keďže časová pečiatka bloku je ešte v 120 minútovom povolenom intervale.

Obeť nebude mať k dispozícii hlavnú blokovú reťaz. Klient obete odmietne aj všetky bloky, ktoré vytvoria baníci so sieťovým časom $x + 140$ minút, keďže rozdiel medzi časovou pečiatkou baníkových blokov a sieťového času obete je viac než 120 minút. Obeť je v nebezpečnom stave, lebo platné bloky bude považovať za neplatné a nemôže zistiť/načítať skutočnú hlavnú vetvu blokovej reťaze.

Obeť sa dostane von z tohto nebezpečného stavu až keď:

- neovplyvnený baník vytvorí blok, alebo
- samotná obeť vytvorí blok, alebo
- sieťové časy sa znova nastavia na správne hodnoty (napríklad keď na uzol so zlým sieťovým časom sa pripoja uzly so správnym časom), alebo
- sama obeť zistí tento stav a urobí nejaké protiopatrenia.

Aj v prípade, že neovplyvnený baník so správnym časom vytvorí blok, nebezpečný stav u obete bude trvať minimálne 140 minút, keďže útočníkov klamlivý blok bol vytvorený s časovou pečiatkou, ktorá je o 260 minút väčšia než aký bol sieťový čas obete. Od začiatočného času tohto nebezpečného stavu musí uplynúť aspoň 140 minút, aby obeť mohla akceptovať útočníkov klamlivý blok. Po akceptovaní klamlivého bloku obeť už dokáže načítať skutočnú hlavnú vetvu, čím sa dostane von z ohrozenia.

Poznámka: Dá sa spraviť aj jednoduchšia verzia útoku posunutím sieťového času, ak útočník zmení iba čas obete (čas baníkov nebude meniť). Ak neovplyvnený baník so správnym časom vytvorí blok, tak v tomto prípade bude obeť zaručene v nebezpečnom stave iba po dobu 70 minút, čo znižuje pravdepodobnosť úspechu útočníka.

Popis realizácie útoku typu double-spending

Po uskutočnení vyššie popísanej prípravy útočník dosiahne, že po istú dobu obeť nedostane bloky z hlavnej vetvy blokovej reťaze. Útočník počas tejto doby môže vytvárať a posielať obeti vlastné bloky, pričom poctivé uzly nezasiahnu, lebo z ich pohľadu toto prebieha vo vedľajšej vetve. Útočník vie, že overenia obeti poslaných transakcií budú

zrušené, lebo ide o vedľajšiu vetvu. Vo vedľajšej vetve útočník uskutoční smerom k obeti prevod Bitcoinov, ale v hlavnej vetve útočník pre rovnaké Bitcoiny zadá prevod na svoju adresu.

Útočník v štandardnom prípade overovania potrebuje vyrobiť 6 blokov (overení), aby používateľ považoval transakciu za platnú. Obeť potom môže útočníkovi napríklad zaslať objednaný tovar v domnienke, že ten je riadne zaplatený.

Ak útočník má 10 percent celkovej výpočtovej sily baníkov, tak dokáže vyrobiť za 330 minút aspoň 6 potvrdení s viac ako 10 percentnou pravdepodobnosťou. Za 200 minút útočník dokáže vyrobiť aspoň 6 potvrdení s pravdepodobnosťou vyššou než 1% . Ak útočník má k dispozícii len 140 minút, tak má len 0,147 percentnú šancu na vytvorenie 6 transakcií, ale aj táto pravdepodobnosť je 6-krát vyššia v porovnaní s pravdepodobnosťou úspechu pri štandardnom útoku typu „double-spending“.

Dôsledky útoku v prípade baníkov

Ak obeťou je baník, tak jeho výpočtový výkon po dobu útoku sa vynakladá zbytočne, lebo baník nebude hľadať riešenia pre hlavnú vetvu blokovej reťaze, ale pre vedľajšiu vetvu. Ak útočníkom je taktiež baník, tak útočník môže takýto útok využiť na vyradenie „konkurenčných“ baníkov a získať tak väčší podiel pri generovaní blokov.

Takýto útok by mohol teoreticky fungovať aj vtedy, keby obeť (baník) mala len o pár sekúnd skorší sieťový čas v porovnaní so sieťovým časom baníkov, ktorí tvoria väčšinu výpočtovej sily (väčšinoví baníci). Predpokladajme, že väčšinoví baníci majú sieťový čas y. Útočník by vytvoril blok s časovou pečiatkou y+120 minút. Väčšinoví baníci by tento blok akceptovali, ale napadnutý baníka nie a opäť by pracoval zbytočne vo vedľajšej vetve. Podobne môže nastať problém aj keď napadnutý baník má čas posunutý k trošku neskoršiemu údaju v porovnaní so sieťovým časom väčšinových baníkov. V tomto prípade napadnutý baník by mohol akceptovať blok útočníka, ktorý väčšinoví baníci odmietnu.

Takýto útok na baníkov často spôsobí rozvetvenie blokovej reťaze.

Mnohí baníci preto na určenie svojho sieťového času používajú iné kritériá, napríklad namiesto sieťového času stále používajú lokálny čas. Toto môže zmierniť dôsledky niektorých ohrození, nezabráni však rozvetveniu blokovej reťaze.

3.1.5 Zneužitie Bitcoinového systému uložením cudzích dát v blokovej reťazi

V niektorých krajinách je nelegálne mať na počítači uložené určité druhy súborov, alebo takéto súbory distribuovať. Takýto zákaz môže niekto skúsiť obísť zneužitím Bitcoinového systému tak, že medzi údaje transakcií vloží aj (pre transakcie nepotrebné) cudzie dáta. Keďže v súčasnosti všetci používatelia systému sú povinní uchovávať celú blokovú reťaz, takto uložené cudzie dáta sú k dispozícii pre všetkých používateľov systému a toto môže viesť k rôznym právnym komplikáciám.

Baníci sa môžu rozhodnúť odmietnuť vložiť do blokov transakcie s cudzími dátami, ale toto je ťažkopádne, lebo cudzie dáta sa dajú len ťažko identifikovať.

Zneužívateľ Bitcoinového systému môže cudzie dáta uložiť v transakcii napríklad ako falošnú (neexistujúcu) adresu príjemcu. V záujme ochrany Bitcoinového systému takéto transakcie by sa teoreticky mohli vymazať zo systému. Pravdepodobnosť, že niekomu sa podarí vygenerovať reálnu adresu, ktorá by bola totožná s už použitou falošnou adresou je zanedbateľná. Ide preto o transakcie, ktorých výstup s veľkou pravdepodobnosťou nikdy nebude použitý. Identifikácia takýchto falošných adries by však bola ťažká, obzvlášť ak by útočník vkladané cudzie dáta šifroval.

Ďalšie možnosti uloženia cudzích dát do blokovej reťaze sú skúmané v kapitole „Softvér na ukladanie cudzích dát do Bitcoinového systému“.

3.1.6 Zlomenie kryptografie

V Bitcoinovom systéme má kryptografia veľmi široké využitie. Využíva sa hlavne hašovacia funkcia SHA-256. Pri generovaní Bitcoinovej adresy z verejného kľúča používateľa sa používa aj hašovacia funkcia RIPEMD-160. Na generovanie súkromného a verejného kľúča a na podpisovanie transakcií sa používa asymetrická schéma ECDSA.

Hašovacia funkcia SHA-256 aj asymetrická schéma ECDSA sú považované za silné. Hašovacia funkcia RIPEMD-160 je pravdepodobne o niečo slabšia než SHA-256. Avšak spôsob použitia RIPEMD-160 minimalizuje možnosť vzniku problémov z hľadiska bezpečnosti. Samozrejme v budúcnosti tieto algoritmy môžu byť predsa len zlomené. V takomto prípade by sa musel Bitcoinový systém preprogramovať a museli by sa začať používať nové, silnejšie algoritmy.

3.1.7 Ohrozenie systému segmentáciou

Ak by sa Bitcoinová sieť rozdelila (napríklad prerušením komunikačných kanálov) na 2 alebo viac segmentov, tak by sa v oddelených segmentoch systému začala bloková reťaz vyvíjať vzájomne odlišne. Po znovuspojení segmentov blokové reťaze sa skombinujú, a dlhšia výsledná vetva sa stane hlavnou vetvou. Transakcie, ktoré boli v kratších vetvách, stratia svoju overenosť a budú sa zase pridávať do hlavnej vetvy blokovej reťaze.

Existujú dve možnosti podľa dĺžky trvania segmentácie siete:

Sieť je segmentovaná na relatívne krátky čas

Ak segmentácia bude mať dobu trvania kratšiu než čas, za ktorý sa vytvorí 100 blokov hlavnej vetvy ľubovolného segmentu, tak nebudú existovať také transakcie, ktoré by využívali výstup z odmenu generujúcich transakcií. Odmenu generujúce transakcie musia totiž dosiahnuť aspoň 100 overení, aby sa ich výstup dal použiť ako vstup transakcie. To znamená, že až na nižšie uvedené výnimky, všetky transakcie budú v

takomto prípade platné a po spojení segmentov sa všetky tieto postupne znovu overia. Medzi výnimky patria (čiže nebudú platné) napríklad:

- Odmenu generujúce transakcie vedľajších vetiev
- Ak útočník využil segmentáciu na prevedenie útoku typu double-spending (v dvoch segmentoch previedol rovnaké Bitcoiny na rôzne adresy), tak z týchto transakcií zostane platná len tá, ktorá je v najdlhšej vetve. Taktiež všetky ďalšie transakcie týchto Bitcoinov v kratšej vetve sa stanú neplatnými.

Sieť je segmentovaná na dlhší čas

Ak segmentácia bude mať dobu trvania dlhšiu než čas, za ktorý sa vytvorí 100 blokov hlavnej vetvy ľubovolného segmentu, tak sa už môže stať, že okrem vyššie uvedených výnimiek aj ďalšie transakcie stratia svoju platnosť. Ide napríklad o transakcie, ktoré využívajú výstup odmenu generujúcich transakcií.

3.1.8 Útok na používateľov podľa ich IP adresy

IP adresy väčšiny používateľov sú verejné, čo prípadný útočník môže zneužiť na nejaký útok (napríklad na ich anonymitu). Používatelia môžu síce používať programy typu Tor na anonymizáciu ich IP adresy, avšak používať Tor a podobné programy nemôžu všetci používatelia naraz, lebo Bitcoinová sieť by prestala fungovať.

3.1.9 Ignorovanie niektorých transakcií baníkmi

Baníci sa môžu rozhodnúť, že niektoré transakcie odignorujú a nevložia ich do bloku. Takéto transakcie zostanú aktívne a môžu sa vložiť do neskorších blokov. Ignorovanie transakcií môžu baníci využiť na to, aby niekomu uškodili. Baníci však priamo nemôžu získať veľké výhody z toho, či vložia alebo nevložia nejaké transakcie do bloku, keďže sa hašuje iba hlavička bloku, a teda počet transakcií v bloku neovplyvňuje rýchlosť vygenerovania hašu.

3.1.10 Ohrozenie na základe veľkej výpočtovej sily útočníka

Ak útočník má k dispozícii viac ako 50 percent výpočtovej sily baníkov, môže vynechať alebo upraviť poradie transakcií v blokovej reťazi. Toto mu umožňuje:

- Zrušiť svoje transakcie
- Za istých podmienok zrušiť niektoré transakcie ostatných používateľov
- Zabrániť tomu, aby bloky baníkov sa dostali do hlavnej vetvy (Útočník bude vytvárať a pripojovať svoje bloky len na také bloky, ktoré vygeneroval on sám.)

- Zabrániť tomu, aby všetky alebo len vybrané transakcie dostávali overenia

Keďže uzly akceptujú iba platné transakcie a bloky, aj keď má útočník takúto veľkú silu, nedokáže:

- Zabrániť tomu, aby sa transakcie dali odoslať (stále budú mať aspoň nulovú overenosť)
- Zmeniť počet odmenových Bitcoinov v bloku
- Vytvárať nové Bitcoiny z ničoho
- Odoslať Bitcoiny, ktoré útočníkovi nikdy nepatrili

Je ťažké zmeniť už dávnejšie vytvorené bloky, a táto obtiažnosť sa exponenciálne zvyšuje, čím hlbšiu zmenu by chcel útočník spraviť. Ak však útok je úspešne vykonaný, tak len ťažko, alebo dokonca vôbec nebude možné dať do poriadku takýmto útokom upravené transakcie.

Pravdepodobnosť úspechu útočníka s veľkou výpočtovou silou

Útočník môže vykonať úspešný útok typu double-spending aj keď má k dispozícii menej než 50 percent celkovej výpočtovej sily baníkov.

Štandardne sa čaká 6 overení pred potvrdením platnosti transakcie. Deje sa to kvôli tomu, že v blokovej reťazi sa môžu vyskytnúť malé bočné rozvetvenia z hlavnej reťaze. Je to aj ochrana proti tomu, aby útočník mohol ľahko vybudovať novú dlhšiu vetvu v blokovej reťazi. Keby sa to útočníkovi predsa podarilo, tak by dostal pôvodnú transakciu do vedľajšej vetvy a tým pádom by vedel vykonať úspešný útok typu double-spending.

V literatúre [1] sa uvádza, aká je pravdepodobnosť toho, že útočníkovi sa podarí vygenerovať rovnako dlhú vedľajšiu vetvu.

Súťaž vo vytváraní blokov medzi útočníkom a poctivými baníkmi si môžeme predstaviť ako náhodnú prechádzku. Ak sa poctivým baníkom podarí vytvoriť blok, tak sa rozdiel medzi dĺžkou hlavnej vetvy a útočníkovej vedľajšej vetvy sa zvýši o 1. Naopak, ak sa útočníkovi podarí vytvoriť blok, tak tento rozdiel sa zmenší o 1.

Pravdepodobnosť, že sa útočníkovi vo svojej vedľajšej vetve podarí dosiahnuť taký počet blokov, aký je v hlavnej vetve, môžeme prirovnať k problému krachovania hazardného hráča. Predstavme si, že hazardný hráč má neobmedzene veľa peňazí, a bude hrať (možno až nekonečne veľa krát), až kým jeho celkový zisk bude nejaká daná suma peňazí. Pravdepodobnosť, že sa mu to podarí niekedy dosiahnuť, sa rovná pravdepodobnosti, že útočníkovi sa podarí dobehnúť hlavnú vetvu. Podľa [22] platí pre túto pravdepodobnosť:

$$q_z = \left\{ \begin{array}{ll} 1 & \text{ak } p \leq q \\ (q/p)^z & \text{ak } p > q \end{array} \right\}$$

kde p je pravdepodobnosť toho, že sa poctivému používateľovi podarí vygenerovať nasledujúci blok
q je pravdepodobnosť toho, že sa útočníkovi podarí vygenerovať nasledujúci blok
q_z je pravdepodobnosť toho, že útočník vôbec niekedy sa vyrovná úrovni hlavnej vetvy, vychádzajúc z pozície o z blokov pozadu.

Keďže vychádzame z predpokladu, že $p > q$, pravdepodobnosť s nárastom hodnoty z klesá exponenciálne.

Vyhodnoťme teraz, na koľko overení by teda mal príjemca čakať, aby mohol si byť dostatočne istý, že nebude obeťou útoku typu double-spending:

Predpokladajme, že odosielateľ je útočník, ktorý chce dosiahnuť, aby príjemca transakcie si na chvíľu myslel, že od útočníka skutočne dostal nejaké Bitcoiny. Potom ale útočník rýchlo vytvorí dlhšiu vedľajšiu vetvu v blokovej reťazi, v ktorej bude rovnaká transakcia, a ktorej výstup však už bude smerovať na útočníkovu adresu. Príjemca bude upozornený, keď k tomuto dôjde, ale odosielateľ predpokladá, že toto upozornenie dostane príjemca už neskoro. Platba útočníka bude teda neplatná, ale medzičasom útočník už mohol od prijímateľa dostať nejakú protihodnotu za falošnú platbu.

Detailnejší popis útoku:

Predpokladajme, že obeť (príjemca) vždy vytvára novú adresu pre príjem transakcie. Túto novú adresu dostane útočník len pred vytvorením transakcie. Toto zabráni útočníkovi, aby mohol začať predvytvárať bloky (transakciu by v opačnom prípade útočník vždy chcel vykonať iba vtedy, keď sa mu šťastnou náhodou podarí viac blokov predvytvoriť). Útočník spraví smerom k obeti transakciu, ktorú označme symbolom T. Po vykonaní transakcie T, útočník začne v tajnosti vytvárať bloky pre vedľajšiu vetvu, kde bude mať upravenú transakciu U, ktorá smeruje už na útočníka. Obeť čaká až kým transakcia T dostane z overení. Príjemca nevie, koľko blokov sa podarilo už útočníkovi v tajnosti vygenerovať. Ak predpokladáme, že sa bloky vygenerovali v priemernom čase, tak útočníkove napredovanie môžeme vyjadriť pomocou náhodnej premennej s Poissonovou distribúciou so strednou hodnotou:

$$\lambda = z\frac{q}{p}$$

Aby sme vypočítali pravdepodobnosť toho, že sa útočníkovi podarí vyrovnať sa úrovni hlavnej vetvy, vynásobíme pravdepodobnosť všetkých možných napredovaní vykonaných v tajnosti s pravdepodobnosťou toho, že útočníkovi z daného bodu sa ešte podarí vyrovnať sa úrovni hlavnej vetvy:

$$\sum_{k=0}^{\infty} \frac{\lambda^k e^{-\lambda}}{k!} \cdot \left\{ \begin{array}{ll} (q/p)^{z-k} & \text{ak } k \leq z \\ 1 & \text{ak } k > z \end{array} \right\}$$

Po úprave dostaneme:

$$1 - \sum_{k=0}^{z} \frac{\lambda^k e^{-\lambda}}{k!} \left(1 - (q/p)^{(z-k)}\right)$$

Pravdepodobnosť sa dá vypočítať pomocou jednoduchého programu. Z výsledkov

takéhoto výpočtu vyplýva, že pravdepodobnosť klesá exponenciálne s nárastom hodnoty z.

Pravdepodobnosť, že sa útočníkovi s 10 percentnou výpočtovou silou ($p = 10\%$) podarí vykonať úspešný double-spending útok:

Hodnota z	Pravdepodobnosť úspešného double-spending útoku, ak transakcia má z overení
0	100 %
1	20.45873 %
2	5.09779 %
3	1.31722 %
4	0.34552 %
5	0.09137 %
6	0.02428 %
7	0.00647 %
8	0.00173 %
9	0.00046 %
10	0.00012 %

Pravdepodobnosť, že sa útočníkovi s 30 percentnou výpočtovou silou ($p = 30\%$) podarí vykonať úspešný double-spending útok:

Hodnota z	Pravdepodobnosť úspešného double-spending útoku, ak transakcia má z overení
0	100 %
5	17.73523 %
10	4.16605 %
15	1.01008 %
20	0.24804 %
25	0.06132 %
30	0.01522 %
35	0.00379 %
40	0.00095 %

Potrebný počet overení, aby pravdepodobnosť úspešnosti útoku bola menšia ako 0,1%, je:

Hodnota p (relatívna sila útočníka)	Potrebný počet overení (hodnota z)
5 %	4
10 %	5
15 %	8
20 %	11
25 %	15
30 %	24
35 %	41
40 %	89
45 %	340

3.1.11 Útok zaťažením systému veľkým počtom transakcií

Na Bitcoinový systém je možné zaútočiť aj využitím toho, že útočník môže celkom jednoducho opakovane posielať transakcie medzi svojimi adresami. Ak by tieto transakcie naplnili maximálnu veľkosť bloku (1 MB), už by nezostalo miesto pre ostatné transakcie a tieto by mohli byť pridané iba do ďalšieho bloku, čo by viedlo k oneskoreniu riadnych transakcií. Aj proti takémuto útoku slúžia transakčné poplatky. Iba prvých 50kB transakcií je v bloku zadarmo, potom sa už väčšinou požaduje transakčný poplatok. Útočník by teda pri takomto útoku pomaly minul svoje Bitcoiny. Existencia transakčného poplatku síce „trestá“ aj riadnych používateľov, no našťastie tieto poplatky sú nízke. Transakcie sú zaraďované do blokov aj podľa priority (aj na základe veku Bitcoinov), takže ak by útočník posielal len tie isté Bitcoiny dokola, tak jeho útok by bol menej efektívny.

3.1.12 Finney-ho útok

Tento typ útoku ako prvý popísal pán Hal Finney [23], preto je podľa neho pomenovaný. Ide o útok typu double-spending, pri ktorom prijímateľ akceptuje neoverené transakcie.

Predpokladajme, že útočníkovi sa občas podarí vygenerovať blok. Do každého bloku vloží transakciu z adresy A na adresu B, pričom obe adresy sú útočníkove. Ak sa útočníkovi podarí vygenerovať blok, tak ho nerozošle hneď, ale najprv nakúpi niečo od obete: pošle Bitcoiny z adresy A na adresu C, kde C je adresa obete, o ktorej útočník vie, že akceptuje aj v danom čase ešte neoverené transakcie. Obeť nepočká na overenie transakcie, a dá útočníkovi nakúpené produkty. Útočník potom rýchlo rozošle svoj utajený blok a transakcia z A na C sa stane neplatnou.

Odporúča sa preto akceptovať len také transakcie, ktoré už majú istú overenosť. V časti „Ohrozenie na základe veľkej výpočtovej sily útočníka“ je popísané, ako ovplyvňuje miera overenosti transakcie pravdepodobnosť toho, že sa útočníkovi podarí vybudovaním dlhšej vedľajšej vetvy vykonať útok typu double-sending. Originálny Bitcoinový klient začne považovať transakciu za overenú, keď tá už získala overenosť 6.

3.1.13 Zlomyseľný klientský softvér

Každý konkurenčný klientský softvér musí dodržiavať pravidlá Bitcoinového systému, v opačnom prípade súčasné Bitcoinové klienty budú tento softvér ignorovať. Existuje však možnosť, že nejaký programátor vytvorí veľmi kvalitný klientský software, ktorý zdanlivo dodržiava pravidlá systému, obsahuje však skrytú výnimku, o ktorej vie len autor programu. Takýto softvér by sa mohol stať veľmi rozšíreným na základe svojej kvality. V čase, keď už takýto software by bol veľmi rozšírený, môže autor programu začať zneužívať zakódovanú výnimku, čo by bolo dosť ťažké odhaliť. Odporúča sa preto používanie iba open-source Bitcoinových klientov, lebo u týchto je možná detailná kontrola softvéru analýzou zdrojového kódu.

3.1.14 Deflácia hodnoty Bitcoinov

V systéme Bitcoin bude maximálne 21 miliónov Bitcoinov, a tieto sú rozmeniteľné na $2.1 \cdot 10^{15}$ ďalej nedeliteľných, minimálnych jednotiek (Satoshi). Maximálny počet minimálnych jednotiek nemusí postačovať, ak celý svet začne používať Bitcoinový systém. Ak deflácia v Bitcoinovom systéme dosiahne veľmi vysokú úroveň, bude treba prejsť na používanie menších jednotiek.

3.2 Možnosti ohrozenia originálneho Bitcoinového klienta

3.2.1 Peňaženka nie je chránená proti krádeži

Peňaženka je súbor, v ktorom sú uchovávané súkromné kľúče používateľa. V originálnom Bitcoinovom kliente tieto údaje sú uložené v počítači nezašifrovane. Hocikto, kto získa prístup k počítaču, môže získať súkromné kľúče a ukradnúť všetky Bitcoiny z príslušných adries. Do novších verzií Bitcoinových klientov už bola pridaná možnosť šifrovania peňaženky, ale táto možnosť nie je štandardne zapnutá.

3.2.2 Ohrozenie pri prevode Bitcoinov na IP adresu

Staršie verzie Bitcoinového klienta umožňujú prevod Bitcoinov na IP adresu. Pri prevode Bitcoinov na IP adresu (pozrite časť „Prevod Bitcoinov na IP adresu“), pri vzájomnej komunikácii medzi odosielateľom a prijímateľom transakcie sa nevykonáva žiadna autentifikácia používateľov. Bitcoinový klient len jednoducho skontaktuje klienta na danej IP adrese a požiada ho o jeho verejný kľúč. Útočník môže spraviť jednoduchý útok typu „man in the middle“ a ukradnúť Bitcoiny.

3.2.3 Útoky typu „Denial Of Service“ (DOS)

Útočník na niektorý uzol Bitcoinovej siete začne posielať veľké množstvo údajov, aby uzol zahltil. Uzol tak nebude môcť spracovávať normálne Bitcoinové správy. Bitcoinový klient obsahuje v sebe jednoduchú ochranu proti útokom typu DOS: odpojí taký uzol, z ktorého začne prichádzať veľké množstvo údajov. Táto jednoduchá ochrana sa však pravdepodobne dá obísť sofistikovanejšími útokmi typu DOS.

Kapitola 4

Združená ťažba Bitcoinov (Pooled mining)

Pri veľkej obtiažnosti má baník s malou výpočtovou silou malú pravdepodobnosť toho, že sa mu v prijateľnom čase podarí vygenerovať blok. Možno bude musieť pracovať aj viac mesiacov, kým sa mu to podarí.

Aby baníci s menšou výpočtovou silou mohli mať pravidelnejší príjem z ťažby, môžu sa združiť a ťažiť (t.j. vytvárať nové bloky a získavať za to Bitcoiny) spoločne. Po úspešnom vygenerovaní bloku si rozdelia zisk vzájomne medzi sebou. Operátor združenej skupiny si však pravdepodobne bude nárokovať istý percentuálny podiel na zisku. Združenú ťažbu Bitcoinov rozoberá literatúra [24].

Pri združenej ťažbe baníci klasickým spôsobom hľadajú vhodnú Nonce hodnotu, ale len s obtiažnosťou $D = 1$. To znamená, že pravdepodobnosť toho, že vygenerovaný blok bude vhodný, je zhruba $1/2^{32}$. Keďže nie je známe, pre akú Nonce hodnotu bude mať haš funkcia dostatočne nízku hodnotu, baníci musia v priemere spraviť 2^{32} výpočtov, aby našli vhodnú Nonce hodnotu. Ak sa baníkovi podarí nájsť vhodnú Nonce hodnotu, tak odošle toto riešenie operátorovi združenej skupiny, ktorý môže jednoducho skontrolovať platnosť predloženého riešenia. Predložené riešenie sa považuje za príspevok (share) baníka. Podľa počtu platných príspevkov konkrétneho baníka sa teda dá celkom presne určiť, aké je množstvo ním vykonanej práce. Na nájdenie bloku baníci v priemere budú musieť odovzdať operátorovi taký počet príspevkov, aká je momentálna hodnota obtiažnosti v Bitcoinovom systéme.

Nájdenie vhodnej metódy rozdelenia odmeny medzi prispievajúcimi baníkmi vôbec nie je jednoduché. Používa sa viac metód, každá z nich má svoje výhody, ale aj nevýhody. Najčastejšie sa používajú nasledovné metódy: úmerné rozdelenie, platba za príspevok, metóda Slush, platba za posledných N príspevkov.

4.1 Úmerné rozdelenie odmeny

Ide asi o najjednoduchší spôsob rozdelenia odmeny. Hľadanie bloku sa uskutočňuje v kolách, každé kolo sa končí, keď sa nájde platná Nonce hodnota. Vyplácanie Bitcoinov sa uskutočňuje na konci každého kola, zisk sa rozdelí priamo úmerne podľa počtu príspevkov jednotlivých baníkov v danom kole.

Ak skupina má šťastie a podarí sa jej vygenerovať blok rýchlejšie, tak baníci dostanú za relatívne malú prácu nejakú odmenu. Ak by skupina nemala šťastie, tak baníci musia viac pracovať, pričom celková odmena zostane rovnaká. Prefíkaný baník môže využiť tento fakt a zapojiť sa do práce vždy len na začiatku kola, ale keď zistí, že sa skupine nedarí, tak v nej prestane pracovať a začne pracovať v inej skupine. Takto si môže zvýšiť svoj príjem na úkor poctivých baníkov. Takáto činnosť baníka sa volá preskakovanie medzi skupinami (pool-hopping).

Úmerné rozdeľovanie zisku by fungovalo dobre v takom prípade, ak by sa dalo presne dopredu určiť, koľko práce treba spraviť, aby sa dokončilo kolo. Úspech pri hľadaní Nonce hodnoty je však úplne náhodný. Nedá sa dopredu určiť potrebné množstvo práce. V priemere bude potrebných D príspevkov v jednom kole. Počet potrebných príspevkov má však geometrické rozdelenie, ktoré je jediné diskrétne náhodné rozdelenie bez pamäte [25]. Teda, ak v kole sa už vyrátalo X príspevkov, priemerný počet ešte potrebných príspevkov na nájdenie Nonce hodnoty nebude menší, budeme potrebovať stále ešte v priemere D ďalších príspevkov.

Niektoré skupiny si vytvárajú rôzne pravidlá na zamedzenie preskakovania medzi skupinami, no tieto majú rôznu účinnosť a často poškodzujú aj poctivých používateľov. Problémy s preskakovaním viedli k vytvoreniu iných, odolnejších a spravodlivejších metód rozdelenia odmeny.

4.2 Platba za príspevok (Pay Per Share - PPS)

Pri tejto metóde rozdelenia odmeny baník okamžite dostane odmenu za každý odovzdaný príspevok. Operátor si teda prenáša všetky riziká na seba. Pri vygenerovaní bloku však celá odmena patrí operátorovi.

Pri metóde PPS je odmena za príspevok pevne daná. Toto má viac výhod pre baníkov:

- Nulová variancia odmeny za príspevok
- Nemusí sa čakať na odmenu, až kým sa nájde blok
- Jednoducho sa dá overiť či operátor nepodvádza a či skutočne vypláca odmenu
- Preskakovanie medzi skupinami nepoškodzuje skupinu

Tento systém je však dosť riskantný pre operátora, keďže iba on zabezpečuje nulovú varianciu odmeny baníkov. Ak skupina má šťastie a bloky sa nájdu rýchlo, tak operátor môže rýchlo zarobiť. Ak však skupina nemá šťastie a bloky sa budú dlho hľadať,

tak operátor môže aj zbankrotovať. Kompenzuje sa to tým, že operátor pri tejto metóde často platí podpriemernú odmenu baníkom.

Pri tejto metóde môže dochádzať aj k útoku na operátora zo strany nepoctivého baníka. Takýto nepoctivý baník sa zapojí do PPS skupiny. Pred odovzdaním svojho príspevku operátorovi nepoctivý baník skontroluje, či našiel skutočné riešenie bloku. Ak áno, tak príspevok neodovzdá. Odovzdá vždy len neúspešné riešenia. Takýmto postupom nepoctivý baník nič nezíska, ale operátora môže finančne poškodiť, lebo ten nedostane odmenu za vyriešené bloky.

4.3 Metóda Slush

Táto metóda rozdelenia odmeny bola navrhnutá práve na vylúčenie výhodnosti preskakovania medzi skupinami (pool-hopping). Metóda využíva štandardný úmerný princíp, platí sa na konci každého kola, rozdelenie odmeny je však nasledovné: Pre každého používateľa sa vypočítava jeho skóre. Ak používateľ odovzdá príspevok, tak sa mu zvýši skóre, pričom veľkosť zvýšenia závisí od uplynutého času od začiatku kola. Príspevky odovzdané neskôr majú väčšie ohodnotenie, ohodnotenie narastá exponenciálne. Na konci kola sa odmena rozdeľuje priamo úmerne podľa toho, aké skóre dosiahol konkrétny baník.

Tento spôsob zaručuje, že po začatí kola, keď sa už systém odmeňovania dostal do stabilného stavu, je skoro úplne jedno, či používateľ pridáva príspevky na začiatku alebo až neskôr na konci kola.

Táto metóda rozdeľovania odmeny teda vylučuje výhodnosť preskakovania, má však nasledovné nedostatky:

- Do stabilného stavu sa systém odmeňovania dostane len po nejakom čase po začatí kola. Ak kolo je krátke, existuje len málo predložených príspevkov, a odmena sa delí len medzi malým množstvom používateľov. Vytváranie príspevkov úplne na začiatku kola teda bude vždy výnosnejšie.
- Hodnota predloženého príspevku závisí od času, ktorý uplynul od začiatku kola, a nie od počtu príspevkov, ktoré boli predložené od začiatku kola. Prefíkaný baník by mohol sledovať celkový výpočtový výkon skupiny a podľa toho robiť pool-hopping.

4.4 Platba za posledných N príspevkov (Pay Per Last N Shares - PPLNS)

Systémy typu PPLNS nefungujú na princípe kôl. Namiesto rozdelenia odmeny podľa počtu príspevkov baníkov v danom kole, rozdeľuje sa podľa toho, koľko príspevkov predložili baníci v poslednom čase. Tento typ rozdeľovania zisku odstraňuje nevýhody takých metód, pri ktorých sa viac oplatí pracovať na začiatku kola.

Pri tejto metóde tiež môže dochádzať aj k útoku na operátora zo strany nepoctivého baníka. Nepoctivý baník by mohol príspevky, ktoré sú riešeniami bloku, neodovzdať operátorovi. Z tohto nepoctivý baník nebude mať zisk. Takýto útok sa môže využiť na to, aby sa celkový výkon skupiny znížil o útočníkovu silu, a tým pádom členovia skupiny by dostávali menšie odmeny.

Pri inom spôsobe útoku nepoctivý baník môže spraviť nasledovný trik na zvýšenie svojich príjmov: Zapojí sa do viac PPLNS skupín a v každej začne vytvárať príspevky rovnakou výpočtovou silou. Ak sa mu podarí nájsť príspevok, ktorý je aj riešením bloku, tak dané riešenie neodovzdá operátorovi hneď. Zameria svoju celú výpočtovú silu na vytváranie príspevkov v danej skupine. Takto zvýši frekvenciu príspevkov v poslednom čase a až potom odovzdá riešenie. Takto dostane vyššiu odmenu. Samozrejme existuje aj riziko, že medzičasom niekto úplne iný nájde riešenie a teda útočníkovo riešenie sa dostane na vedľajšiu vetvu.

4.5 Zdanlivo možný nekalý postup baníka

Na prvý pohľad sa zdá, že pri všetkých vyššie uvedených spôsoboch rozdeľovania odmeny baník môže podvádzať aj nasledovnými spôsobmi:

- baník odovzdá vyrátaný príspevok do viacerých skupín
- baník neodovzdá príspevok, ktorý je aj riešením bloku, príslušnej skupine, ale pokúsi sa získať celú odmenu z odmenu generujúcej transakcie len pre seba

V skutočnosti však tieto nekalé spôsoby neprichádzajú do úvahy. S veľmi veľkou pravdepodobnosťou každá skupina dáva totiž iné vstupné dáta na vyrátanie príspevku (bude iná prinajmenšom aspoň odmenu generujúca transakcia), teda baník nemôže úspešne odovzdať príspevok do iných skupín. Ak by nepoctivý baník chcel získať celú odmenu pre seba, tak by musel zmeniť výstup odmenu generujúcej transakcie. To by mu však pokazilo správnosť vyrátanej Nonce hodnoty.

Kapitola 5

Softvér na ukladanie cudzích dát do Bitcoinového systému

V časti „Bezpečnosť“ je vysvetlené, že do blokovej reťaze sa dajú uložiť aj cudzie (nesystémové) dáta. Ako praktická ukážka takejto zraniteľnosti systému je uvedený autorom vytvorený softvér na vkladanie cudzích dát do blokovej reťaze. V nasledujúcich častiach sú najprv popísané rôzne možnosti vkladania cudzích dát do blokovej reťaze. Potom nasleduje popis fungovania softvéru.

5.1 Možnosti ukladania cudzích dát do blokovej reťaze

Bloková reťaz sa skladá z blokov a bloky obsahujú transakcie, ktoré vytvárajú používatelia Bitcoinovej siete. Cudzie dáta do blokovej reťaze sa najjednoduchšie dajú vkladať ich vkladaním priamo do transakcií. Na uloženie cudzích dát do transakcie je viac spôsobov:

5.1.1 Uloženie dát do adries vo výstupnej časti transakcie

Asi najjednoduchší spôsob ukladania dát je použitie štandardnej transakcie ako pri klasickom prevode Bitcoinov na Bitcoinovú adresu. Prevod sa však nebude uskutočňovať na skutočne existujúce adresy, výstupné adresy budú tvoriť cudzie dáta. Bitcoinové adresy majú veľkosť len 20 B, čiže do každého výstupu transakcie môžeme vložiť len 20 B dát. Ďalšou nevýhodou je, že posielané Bitcoiny sa prevádzajú na neexistujúcu adresu, a tak dôjde k ich strate. Túto stratu môžeme minimalizovať prevodom minimálnych jednotiek (1 Satoshi). Ďalšou nevýhodou je, že každý výstup transakcie musí okrem adresy obsahovať aj povinné systémové údaje, ktoré sú vždy pridané ku každým 20 B cudzích dát. Toto znižuje efektívnosť tejto metódy vkladania cudzích dát.

5.1.2 Uloženie dát do verejného kľúča vo výstupnej časti transakcie

Ďalší jednoduchý spôsob ukladania dát je použitie transakcie ako pri prevode Bitcoinov na IP adresu. Namiesto výstupných adries cudzie dáta v tomto prípade ukladáme do verejných kľúčov vo výstupnej časti transakcie. Verejné kľúče majú veľkosť 65 B, preto tento spôsob ukladania je efektívnejší než pri ukladaní cudzích dát do výstupných adries. Aj v tomto prípade však existuje tá nevýhoda, že posielané Bitcoiny ďalej už nebudú môcť byť využívané. Množstvo stratených Bitcoinov môžeme opäť minimalizovať prevodom minimálnych jednotiek.

5.1.3 Uloženie dát do skriptu vo výstupnej časti transakcie

Ďalšou možnosťou vkladania cudzích dát je, že vo výstupnej časti transakcie, namiesto klasického skriptu na prevod (viď. popis v časti „Typy transakcií“), si vytvoríme vlastný skript, ktorý bude obsahovať cudzie dáta. Skript môže obsahovať až 10kB údajov, čo dáva dobrý pomer medzi množstvom vkladaných cudzích dát a množstvom povinných dát transakcie. Ak cieľom je iba vkladanie cudzích dát, cez každý výstup transakcie by sa opäť mali posielať iba minimálne jednotky Bitcoinov. Skript môžeme navrhnúť aj tak, aby sme posielané Bitcoiny mohli opäť využiť. Nemusí to však znamenať ušetrenie Bitcoinov, lebo potrebný transakčný poplatok (z dôvodu nadmerného počtu vstupných údajov transakcie) by mohol byť väčší, než množstvo ušetrených Bitcoinov.

Príklad skriptu:

```
<CUDZIE DÁTA> OP_DROP <CUDZIE DÁTA> OP_DROP ... <CUDZIE DÁTA> OP_DROP OP_DUP
OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG
```

Každá položka v skripte môže mať maximálne 520 B, preto, ak ukladáme väčšie množstvo cudzích dát, tieto musia byť rozdelené do viacerých častí po 520 B (podľa vyššie uvedeného príkladu). Príkaz `OP_DROP` vymaže predchádzajúcu položku zo zásobníka. Vyššie uvedený skript funguje rovnako ako posielanie Bitcoinov na Bitcoinovú adresu, a tak posielané Bitcoiny sa dajú opäť použiť.

Nevýhoda tejto možnosti vkladania cudzích dát je v tom, že ide o neštandardnú transakciu. Väčšina Bitcoinových klientov (a možno aj baníkov) nešíri ďalej a odmieta neštandardné transakcie. Dostať takúto transakciu do bloku bude preto celkom problematické. V súčasnosti sa v Bitcoinovom systéme používajú iba 3 druhy transakcií (viď. popis v časti „Typy transakcií“), preto sa jednoducho identifikuje takáto neštandardná transakcia. S ďalším vývojom Bitcoinového systému sa to však môže ešte zmeniť a možno budú povolené aj vlastné skripty používateľov. Takýto spôsob vkladania cudzích dát však môže realizovať baník, lebo on môže vložiť túto neštandardnú transakciu do bloku, ktorý on sám vytvára.

5.1.4 Uloženie dát do skriptu vo vstupnej časti transakcie

Podobne ako v predchádzajúcom prípade, mohli by sme uložiť nejaké dáta aj do skriptov, ktoré sú vo vstupnej časti transakcie. Podmienky vkladania cudzích dát sú podobné ako pri využívaní skrípt vo výstupnej časti transakcie.

5.1.5 Uloženie dát do odmenu generujúcej transakcie

Baníci môžu uložiť cudzie dáta do parametra „coinbase“ v odmene generujúcej transakcii.

5.2 Popis softvéru

Vytvorený softvér umožňuje experimentovať s ukladaním cudzích dát do blokovej reťaze. Softvér sa nazýva BAW (Bitcoin Alien data Writer), a podporuje prvé 3 spôsoby ukladania dát, ktoré boli popísané v predchádzajúcej časti:

- Uloženie dát do adries vo výstupnej časti transakcie
- Uloženie dát do verejného kľúča vo výstupnej časti transakcie
- Uloženie dát do skriptu vo výstupnej časti transakcie

Softvér BAW umožňuje pri vkladaní údajov nastaviť rôzne parametre, napríklad množstvo posielaných Bitcoinov, veľkosť transakčného poplatku. Ak ukladáme dáta do skriptu vo výstupnej časti transakcie, tak softvér umožňuje nastaviť aj výstupnú adresu.

Okrem vyššie uvedenej funkcionality softvér BAW ukazuje aktuálny stav blokovej reťaze, počet pripojených uzlov, aktuálny stav účtu a stavy odoslaných transakcií. Softvér obsahuje aj možnosť pripojiť sa na také uzly, ktoré podporujú FTRP (Free Transaction Relay Policy). Ide o používateľov a baníkov, ktorí šíria ďalej a vkladajú do blokov aj neštandardné transakcie.

Softvér BAW bol naprogramovaný v jazyku Java, využíva knižnicu „bitcoinj“ [26].

Autor vytvoril aj softvér BAR (Bitcoin Alien data Reader), ktorý umožňuje z blokovej reťaze načítať cudzie dáta, ktoré boli vložené softvérom BAW. Cudzie dáta sa načítajú na základe hašu transakcie, pomocou ktorej boli uložené.

5.3 Praktické testy vkladania cudzích dát

Pomocou softvéru BAW sme testovali vkladanie dát do blokovej reťaze a potom sme testovali aj ich načítanie pomocou softvéru BAR.

Testy ukladania dát do adries vo výstupnej časti transakcie: Aj vkladanie aj načítanie cudzích dát bolo úspešné.

Testy ukladania dát do verejného kľúča vo výstupnej časti transakcie: Aj v tomto prípade bolo úspešné aj vkladanie aj načítanie cudzích dát.

Testy ukladania dát do skriptu vo výstupnej časti transakcie: Vkladané dáta sa zapísali do transakcie, ale daná transakcia potom už nebola zapísaná do blokovej reťaze. Na stránke blockchain.info sa podarilo nájsť danú transakciu medzi odmietnutými transakciami. Transakcia bola odmietnutá z dôvodu neštandardnosti. Testy boli opakované aj pripojením sa na uzol FTRP (Free Transaction Relay Policy), výsledok však bol rovnaký.

Záver

V tejto práci bolo vysvetlené fungovanie Bitcoinového systému z technického hľadiska a so zameraním sa na otázky jeho anonymity a bezpečnosti.

Tvorca systému pri jeho návrhu nebral za prvoradý cieľ zabezpečenie anonymity používateľov. Preto používatelia musia rátať s tým, že môže dôjsť k odhaleniu ich identity.

Bitcoinový systém je možné považovať za dostatočne bezpečný, ak používateľ vždy pracuje s dostatočne overenými transakciami.

Na záver, ako príklad využitia zraniteľnosti systému, sú v tejto práci úspešne prezentované autorom vytvorené softvérové aplikácie, ktoré umožňujú vkladať a načítať cudzie dáta do/z databázy Bitcoinového systému. Naša aplikácia ukazuje, že využitie uvedenej zraniteľnosti je teoreticky možné (proof of concept), ale bez ďalších vylepšení nie je dostatočne praktické.

Literatúra

[1] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. http://bitcoin.org/bitcoin.pdf, 2008.

[2] Technical background of version 1 bitcoin addresses - bitcoin. https://en.bitcoin.it/wiki/Technical_background_of_Bitcoin_addresses.

[3] Transactions - bitcoin. https://en.bitcoin.it/wiki/Transactions.

[4] Script - bitcoin. https://en.bitcoin.it/wiki/Script.

[5] Op checksig - bitcoin. https://en.bitcoin.it/wiki/OP_CHECKSIG.

[6] Blocks - bitcoin. https://en.bitcoin.it/wiki/Blocks.

[7] Block chain - bitcoin. https://en.bitcoin.it/wiki/Block_chain.

[8] Block hashing algorithm - bitcoin. https://en.bitcoin.it/wiki/Block_hashing_algorithm.

[9] Target - bitcoin. https://en.bitcoin.it/wiki/Target.

[10] Difficulty - bitcoin. https://en.bitcoin.it/wiki/Difficulty.

[11] Nonce - bitcoin. https://en.bitcoin.it/wiki/Nonce.

[12] Block timestamp - bitcoin. https://en.bitcoin.it/wiki/Block_timestamp.

[13] Protocol specification - bitcoin. https://en.bitcoin.it/wiki/Protocol_specification.

[14] Hash tree. http://en.wikipedia.org/wiki/Hash_tree.

[15] Network - bitcoin. https://en.bitcoin.it/wiki/Network.

[16] Martin Harrigan Fergal Reid. An analysis of anonymity in the bitcoin system. http://arxiv.org/abs/1107.4524, 2011.

[17] Anonymity - bitcoin. https://en.bitcoin.it/wiki/Anonymity.

[18] Browser-based wallet - bitcoin. https://en.bitcoin.it/wiki/EWallet.

[19] Mixing service - bitcoin. https://en.bitcoin.it/wiki/Mixing_service.

[20] Weaknesses - bitcoin. https://en.bitcoin.it/wiki/Weaknesses.

[21] corbixgwelt. Timejacking & bitcoin. http://culubas.blogspot.com/2011/05/timejacking-bitcoin_802.html, 2011.

[22] Karl Sigman. Gambler's ruin problem. www.columbia.edu/~ks20/FE-Notes/4700-07-Notes-GR.pdf.

[23] The finney attack. https://bitcointalk.org/index.php?topic=3441.msg48384#msg48384.

[24] Meni Rosenfeld. Analysis of bitcoin pooled mining reward systems. http://bitcoil.co.il/pool_analysis.pdf, 2011.

[25] Geometric distribution - from wolfram mathworld. http://mathworld.wolfram.com/GeometricDistribution.html.

[26] bitcoinj - a java implementation of a bitcoin client-only node. http://code.google.com/p/bitcoinj/.

Príloha

CD

Všetky zdrojové súbory programov BAW a BAR sú k dispozícii na priloženom CD. Toto CD obsahuje aj skompilovanú verziu týchto programov.