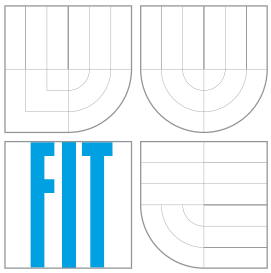
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

FORENZNÍ ANALÝZA SÍTĚ BITCOIN

FORENSIC ANALYSIS OF BITCOIN NETWORK

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. TOMÁŠ DROZDA

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. VLADIMÍR VESELÝ

BRNO 2016

Abstrakt

Táto diplomová práca sa zaoberá problematikou realizácie forenznej analýzy v sieti Bitcoin. Hlavným cieľom práce bolo navrhnúť a následne implementovať prototyp nástroja forenznej analýzy v sieti Bitcoin. Teoretická časť práca je venovaná popisu siete Bitcoin. Následne sú v rámci práce porovnávané existujúce nástroje a špecifikované požiadavky na nástroje forenznej analýzy. Zvyšok práce je zameraný na popis návrhu a implementácie nástroja forenznej analýzy. Nástroj je implementovaný vo forme webovej aplikácie, ktorá umožnuje prehľadávať obsah verejnej účtovnej knihy, graficky vizualizovať toky finančných prostriedkov v sieti, zobrazovať profilové stránky jednotlivých adres uživateľov siete a Bitcoin peňaženiek.

Abstract

This thesis is focused on Bitcoin forensic analysis. Main goal of this thesis was to design and implement system tool for forensic analysis. Theoretical part of this thesis is dedicated to Bitcoin network. Requirements and existing tools are described in following chapters. Rest of the thesis is concerned about implementation of such a tool itself. Implemented system is presented as web application, which allows user to search through blockchain, vizualize financials, display profile pages of Bitcoin users and their Bitcoin wallets.

Kľúčové slová

Bitcoin, forenzná analýza, zhluková analýza, deanonymizácia, transakcie, bloky, blockchain, MongoDB, PHP, laravel, Javascript, Bootstrap

Keywords

Bitcoin, forensic analysis, cluster analysis, deanonymisation, transactions, blocks, blockchain, MongoDB, PHP, laravel, Javascript, Bootstrap

Citácia

DROZDA, Tomáš. *Forenzní analýza sítě Bitcoin.* Brno, 2016. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Veselý Vladimír.

Forenzní analýza sítě Bitcoin

Prehlásenie

Prehlasujem, že som túto diplomovú prácu vypracoval samostatne pod vedením pána Ing. Vladimíra Veselého. Uviedol som všetky literárne zdroje a publikácie, z ktorých som čerpal.

.....
Tomáš Drozda
24. mája 2016

Poďakovanie

Chcel by som sa poďakovať môjmu vedúcemu práce Ing. Vladimírovi Veselému za jeho ústretový prístup, ochotu, vecné pripomienky k riešeniu danej problematiky a motiváciu v priebehu riešenia diplomovej práce.

V neposlednej rade by som sa chcel poďakovať mojej priateľke Martine, ktorá mi ťažké chvíle pri písaní tejto diplomovej práce spríjemňovala svojími povzbudzujúcimi dezertami. Medzi moj najobľúbenejší patrí sladký, nenáročný banánový Cheesecake. Cheesecake sa skladá z dvoch vrstiev - z korpusu pozostavajúci z rozdrobených sušienok a zo syrovej vrstvy. Na korpus je potreba 250 g maslových, rozdrobených sušienok a 120 g rozpusteného masla. Obe ingrediencie zmiešame a postupne ich vtlačíme do tortovej formy vyloženej papierom na pečenie. V predhriatej rúre ho pečieme asi 12 minút. Medzitým si nachystáme syrovú vrstvu, ktorá sa skladá z 500 g syra Philadelphia (môže byť aj Mascaropone), z 1 kyslej smotany, 50 g masla, 2 vajec, 150 g cukru a z 3 roztlačených banánov. Všetky ingrediencie vyšľaháme, vylejeme na upečený korpus a pečieme v rúre vyhriatej na 150 stupňov približne 45 minút. Po upečení ho uložíme na 3 a viac hodín do chladničky, čo je niekedy problém tak dlho vydržať.

Obsah

Kapitola 1

Úvod

V súčasnej dobe narastá popularita v oblasti používania kryptomen pre realizáciu platieb. Kryptomena je digitálna mena založená na použití kryptografie za účelom dosiahnutia bezpečného používania tohoto platidla. Najpopulárnejšou kryptomenou súčasnosti je mena Bitcoin, ktorá ku svojej činnosti využíva decentralizovanú peer-to-peer sieť (P2P) označovanú ako Bitcoin sieť.

Hlavným rozdielom kryptomen od štandardného bankového systému je absencia centrálnej autority, ktorá by spravovala finančné prostriedky (tvorba nových prostriedkov, verifikácia transakcií atď.). Na základe toho nie je možné zmraziť účet žiadneho z užívateľov krpytomen. Platobné transakcie sú realizované priamo medzi účastníkmi siete bez prostredníka, a tak je rýchlosť ich vykonávania neporovnateľne vyššia ako v štandardnom bankovom systéme. Výška poplatkov za realizáciu transakcie je minimálna. V prípade, že klient chce využívať služby siete Bitcoin, klient neposkytuje žiadne osobné údaje, a tak aktivita na sieti je považovaná za anonymnú. Zoznam všetkých vykonaných transakcií v sieti je uložený vo verejnej účtovnej knihe.

S kryptomenami súvisí pojem *Dark web*, ktorý označuje webový obsah umiestnený na počítačových sieťach označovaných ako *darknet*. *Darknet* označuje sieť vytvorenú nad už existujúcou sieťou, pričom hlavným účelom takýchto sietí je skrytie existencie a obsahu realizovaných sieťových komunikácií. Z toho plynie fakt, že veľká časť obsahu *Dark webu* je tvorená webovými službami zaoberajúcimi sa predajom drog, zbraní, hazardom atď. Platby na takýchto službách sú často realizované za pomoci kryptomen.[19]

Cieľom tejto práce je navrhnúť a následne implementovať forenzný nástroj, ktorý by umožňoval prehľadávať obsah verejnej účtovnej knihy. Na základe toho by bolo možné sledovať presuny finančných prostriedkov naprieč sieťou. Ďalším cieľom tejto práce je implementácia funkcií za pomoci, ktorých by bolo možné vykonávať dátové analýzy nad obsahom verejnej účtovnej knihy, hlavne za účelom deanonymizácie účastníkov siete Bitcoin.

Text práce je členený do troch kapitol. Druhá kapitola sa venuje popisu fungovania kryptomeny Bitcoin predovšetkým z technického hľadiska. Dôraz je kladený na popis adresácie klientov, realizácie peňaženiek, transakcií a spôsob evidencie transakcií.

Tretia kapitola sa zameriava na popis návrhu, implementácie a testovaniu nástroja pre podporu realizácie forenznej činností v sieti Bitcoin. Začiatok kapitoly je venovaný popisu aktuálneho stavu v oblasti nástrojov forenznej analýzy pre siete kryptomen. Následne sa kapitola venuje analýze požiadavkov kladených na takéto nástroje a implementácií vlastného nástroja.

Záverečná kapitola práce je obsiahnuté zhrnutie výsledkov práce a prebrané možnosti jej ďalšieho rozšírenia v budúcnosti.

Kapitola 2

Platobný systém Bitcoin

Obsah kapitoly vychádza predovšetkým zo zdroja [1]. Bitcoin je platobný systém ktorý využíva decentralizovanú P2P sieť. Jedná sa o prvú digitálnu menu. Základy tejto meny položila osoba vystupujúca pod pseudonymom (skutočná identita nie je známa) Satoshi Nakamoto v roku 2008 v článku [18]. Spustenie siete Bitcoin prebehlo v roku 2009.

Zásadným rozdielom systému Bitcoin od štandardného bankového systému, s ktorým sa stretávame v bežnom živote je to, že chod systému nie je riadený centrálnou autoritou (banky a vlády krajín). Transakcie sú realizované priamo medzi klientmi čo má za dôsledok malé alebo často krát žiadne poplatky za vykonanie transakcie. Rýchlosť vykonania platobných transakcií je neporovnateľne vyššia. Zvyčajne sa jedná len o pár minút.

Bitcoin je mena s deflačnou tendenciou, v ktorej je známy maximálny počet finančných prostriedkov s ktorými bude systém pracovať. Jedná sa o približne 21 miliónov jednotiek. Na základe tejto skutočnosti nebude možné vytvárať nové prostriedky po prekročení tohoto limitu čím sa zabráni umelo vytváranej inflácií. V prípade systému Bitcoin nie je možné zmraziť účet.

Mena systému Bitcoin je označovaná ako BTC (Bitcoin). Základnou jednotkou, s ktorou systém Bitcoin pracuje je označovaná ako Satoshi. 1 BTC je možné rozdeliť na celkovo 10^8 Satoshi. V dôsledku toho že 1 BTC je tvorený 10^8 Satoshi tak je možné vytvárať transakcie o veľmi malých čiastkach.

Finančné prostriedky zo siete Bitcoin je možné zameniť za bežné meny ako napríklad dolár, euro a iné podľa hodnoty aktuálneho konverzného kurzu. K tomuto účelu sú využívané takzvané zmenárne. konverzného kurzu. K tomuto účelu sú využívané takzvané zmenárne.

Klienti v systéme Bitcoin

V P2P sieti systému Bitcoin nachádzame dva druhy klientov a to:

Koncový užívatelia
: Užívatelia, ktorý využívajú systém pre vykonávanie transakcií.

Baníci
: Klienti, ktorý zabezpečujú bezpečný chod siete pomocou svojej výpočtovej sily. Spravujú obsah spoločnej účtovnej knihy označovanej ako `blockchain`. Za prácu, ktorú títo klienti vykonávajú dostávajú odmenu. Baníci touto prácou zabezpečujú vytváranie nových finančných prostriedkov, s ktorými systém bude pracovať.

2.1 Adresácia klientov a peňaženky

Nasledujúca sekcia sa bude venovať popisu adresácie klientských účtov v sieti Bitcoin a realizácie peňaženiek používaných v sieti Bitcoin.

Komunikácia klienta so sieťou Bitcoin je realizovaná pomocou tzv. peňaženiek. Ku každej peňaženke je priradených niekoľko verejných adries (kľúčov), ktoré tvoria analógiu ku číslam bankových účtov. Súčet prostriedkov na všetkých adresách spravovaných jednou peňaženkou reprezentuje celkové množstvo finančných prostriedkov, s ktorými klient môže disponovať.

Pre úspešne vykonávanie transakcií peňaženka musí obsahovať privátny kľúč, pomocou ktorého sú transakcie podpisované a následne je aj pomocou nich overovaná dôveryhodnosť transakcie.

2.1.1 Tvorba adresy

V tejto sekcii bude popísaný spôsob vytvárania adries používaných pre adresáciu klientov. Adresa je reprezentovaná ako alfanumerický reťazec o dĺžke 25 - 34 znakov. V Bitcoin adrese nie sú obsiahnuté znaky 0, O, l a znak I, pretože často vznikajú preklepy počas ich zadávania (opisovania).

Každá z adries obsahuje kontrolný súčet pomocou ktorého je overovaná správnosť vygenerovanej adresy čím sa predíde odoslaniu finančných prostriedkov na neexistujúcu adresu. Príklad Bitcoin adresy: `3J98t1WpEZ73CNmQviecrnyiWrnqRhWNLy`.

Adresa klienta (verejný kľúč) v sieti Bitcoin je vytváraná pomocou nasledujúceho algoritmu [11].

1. Pomocou algoritmu `ECDSA` [20] je vygenerovaná dvojica kľúčov (verejný a privátny).
2. Verejný kľúč je zahashovaný pomocou algoritmu `SHA-256`.
3. Výsledok druhého kroku je zahashovaný pomocou algoritmu `RIPEMD-160` [9].
4. Ku výsledku tretieho kroku je pridaný prefix o dĺžke 1B reprezentujúci aktuálne verziu protokolu (aktuálne 0x00).
5. $2\star$ je zasahovaný výsledok predchádzajúceho kroku pomocou algoritmu `SHA-256`.
6. Prvé 2 bajty z kroku pat sa použijú ako kontrolný súčet.
7. Kontrolný súčet z kroku šesť sa pripojí na koniec reťazca z kroku štyri čím sa vytvorí adresa o dĺžke 25B.
8. 25B adresa sa zakóduje pomocou `Base58Check` [2] kódovania na alfanumerický reťazec.

2.1.2 Realizácia Bitcoin peňaženky

Bitcoin peňaženka môže byť realizovaná v papierovej forme alebo vo forme programu.

Peňaženka vo forme programu vytvára verejné kľúče (adresy) pomocou, ktorých je možné prijímať finančné prostriedky. Zároveň obsahuje privátne kľúče za pomoci, ktorých je možné prostriedky utrácať.

Papierová peňaženka obsahuje len verejný kľúč (adresu) peňaženky a privátny kľúč. Pre utratenie peňazí je potrebné spolu s papierovou peňaženkou využiť ďalší softvér, ktorý rozšíri informáciu o požiadavku na vykonanie transakcie do siete Bitcoin.

Existujú implementácie Bitcoin peňaženiek prakticky pre všetky typy platforiem. V prípade, že neexistuje klient určený priamo pre danú platformu, tak je možné použiť webovú peňaženku.

Logické komponenty Bitcoin peňaženky

Bitcoin peňaženka môže byť reprezentovaná ako sada troch logických komponent. Každá z komponent má špecifické vlastnosti a činnosti, za ktoré zodpovedá. Komponenty Bitcoin peňaženky sú nasledovné:

- Komponenta realizujúca prácu s verejnými kľúčmi. Za pomoci nej je možné klientské stanice adresovať v platobných transakciách.
- Komponenta realizujúca podpisovanie odchádzajúcich transakcií. Za pomoci danej komponenty je možné odosielať finančné prostriedky na iné Bitcoin adresy.
- Komponenta zapuzdrujúca činnosť dvoch predchádzajúcich komponent. Jedná sa o komponentu, ktorá ako jediná vyžaduje ku svojej činnosti pripojenie ku P2P sieti Bitcoin. Komponenta informuje sieť Bitcoin o novo vzniknutých platobných požiadavkoch. Získava informácie o zmenách vo verejnej účtovnej knihe a zaznamenáva ich do svojej lokálnej kópie.

Spôsob implementácie peňaženky

V súčasnej dobe sa stretávame s dvoma základnými spôsobmi implementácie Bitcoin peňaženky a to vo forme *Full-Service* a vo forme *Signing-Only* peňaženky. Peňaženky typu *Full-Service* sú implementované ako jeden samostatný celok (softvér, hardvér), ktorý integruje všetky funkcie. V prípade peňaženiek typu *Signing-Only* je implementácia peňaženky rozdelené do dvoch komponent (samostatných peňaženiek), kde jedna realizuje podpisovanie transakcií a druhá je určená pre komunikáciu so sieťou Bitcoin.

***Full-Service* peňaženka**
: Jedná sa o najjednoduchší typ peňaženky, ktorý v sebe zahŕňa všetky tri funkcie: generovanie párov privátný-verejný kľúč, distribúcia verejných kľúčov v prípade potreby, vytváranie a podpisovanie transakcií a ich následné šírenie v sieti Bitcoin. Výhodou takéhoto typu peňaženky je jednoduchosť použitia, ktorá je zabezpečená tým, že máme jednu aplikáciu, ktorá vykonáva všetky potrebné činnosti. To že sú privátne kľúče uložené priamo na zariadení, ktoré je pripojené do siete Internet predstavuje bezpečnostnú hrozbu. V prípade, že by sa útočníkovi podarilo získať privátne kľúče, tak by následne mohol vykonať presun finančných prostriedkov na účet/účty útočníka. Tento fakt predstavuje najväčšiu nevýhodu implementácie peňaženky ako *Full-Service*. Príkladom takejto peňaženky je online peňaženka Coinbase[1].

***Signing-Only* peňaženka**
: Jedná sa o typ peňaženky kde proces generovania a následného ukladania privátnych

[1] Popis: https://www.coinbase.com

a verejných kľúčov je realizovaný ako samostatný program, ktorý je umiestnený v prostredí s zvýšenou mierou bezpečnosti. Funkčnosť tohto typu peňaženky je možná len vtedy keď kooperuje so sieťovou časťou peňaženky, ktorá komunikuje s P2P sieťou Bitcoin.

Realizácia peňaženiek typu Signing-Only

Peňaženky typu *Signing-Only* sú najčastejšie realizované v troch podobách označovaných ako *Offline*, *Hardware* alebo *Distributing-Only*.

***Offline* peňaženky**

Offline peňaženka je realizovaná za pomoci zariadenia, ktoré nie je pripojené ku počítačovej sieti napr. za pomoci USB kľúča. Takýto typ peňaženky realizuje podpisovanie transakcií. Pre následnú distribúciu transakcie v sieti Bitcoin je potreba použiť tzv. *Online* peňaženku, pomocou ktorej sú dáta z peňaženky distribuované do P2P siete. Výhodou *Offline* peňaženiek je vysoká miera zabezpečenia. Aby získal útočník prístup ku prostriedkom obete útočník musí vlastniť oba druhy peňaženiek. Nevýhodou použitia *Offline* peňaženky je to, že je potrebná komunikácia s ďalšiou peňaženkou, ktorá zabezpečuje komunikáciu so sieťou Bitcoin. Príkladom *offline* peňaženky je Bitcoin peňaženka bitkey[2].

***Hardwarové* peňaženky**

Jedná sa o implementáciu Bitcoin peňaženky vo forme špecializovaného hardwearového zariadenia, ktoré funguje ako *Signing-Only* peňaženka. Za pomoci toho, že tento hardwear je priamo určený len pre vykonávanie platieb tak je zabezpečená ochrana voči známym zraniteľnostiam z prostredia štandardných operačných systémov, s ktorými sa stretávame (Microsoft Windows, OS X atď.). Výhodou tejto implementácie je ďaleko vyššia miera bezpečnosti oproti implementácii peňaženky ako *Full-Service*. Nevýhodou tejto implementácie je nutnosť vlastniť špecializované hardwearové zariadenie. Tento fakt môže spôsobovať určité problémy, avšak použitie je jednoduchšie ako v prípade peňaženky ako *Offline*. Príkladom *hardwarovej* peňaženky je Bitcoin peňaženka Trezor[3].

***Distributing-Only* peňaženky**

Jedná sa o typ peňaženkového programu, ktorý beží v obtiažne zabezpečovanom prostredí. Napríklad sa môže jednať o webové servery, ktoré sú navrhnuté pre distribúciu verejných kľúčov.

2.2 Transakcie

Transakcie[12] predstavujú prostriedok pomocou, ktorého si klienti platobnej siete Bitcoin posielajú finančné prostriedky.

Transakcie v sieti Bitcoin sú charakteristické tým, že môžu obsahovať ľubovoľný počet vstupov a výstupov.

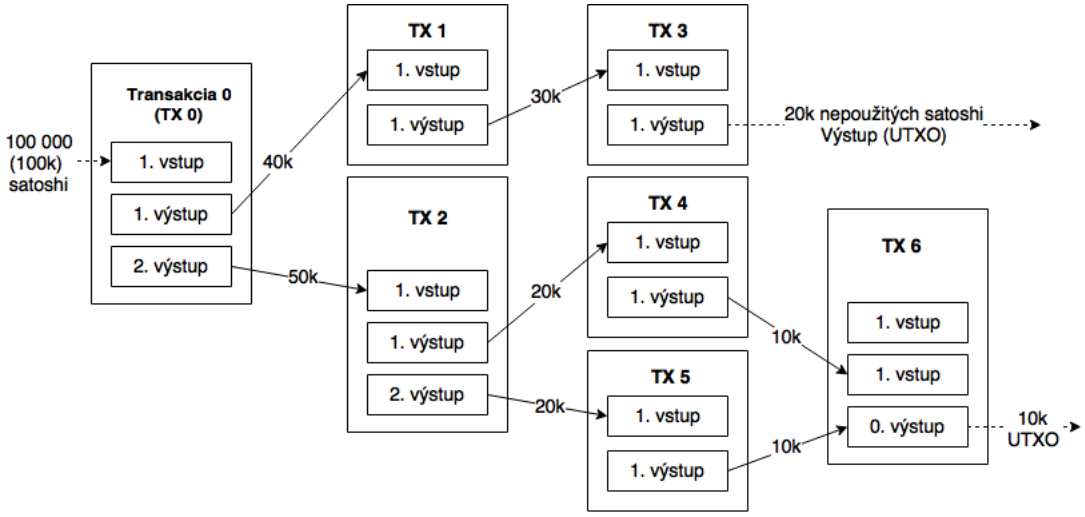
Vstupy transakcií sú tvorené výstupmi už zrealizovaných transakcií, ku ktorým má odosielateľ dispozičné práva a ešte neboli použité pre realizáciu transkacie. Zoznam výstupov

[2]Popis: http://bitkey.io
[3]Popis: https://www.bitcointrezor.com

transakcií, ktoré môžu byť použité ako vstupy ďalších transakcií označujeme ako `Unspent Transaction Output - UTXO`.

Za pomoci obrázku 2.1 je ilustrované to, ako sú transakcie v sieti Bitcoin navzájom previazané. Na obrázku je vidieť že z transakcie označovanej ako TX 0 vznikli ďalšie transakcie. Zároveň tu vidíme, že príjemca finančných prostriedkov po úspešnom vykonaní transakcie TX 3 bude môcť disponovať s 20k Satoshi a v prípade príjemcu po vykonaní transakcie TX 6 sa jedná o 10k Satoshi.

Ako je vidieť aj z predchádzajúceho obrázku platí, že súčet prostriedkov na všetkých `UTXO` klienta predstavuje celkovú čiastku, ktorou klient disponuje.

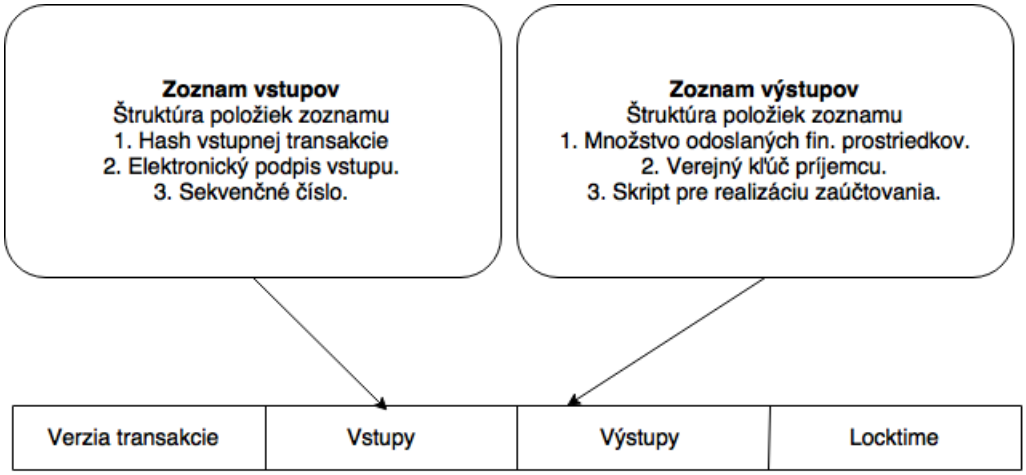


Obr. 2.1: Štruktúra transakcií v sieti Bitcoin.

2.2.1 Správa popisujúca transakciu

Ako už bolo spomenuté platobná sieť Bitcoin je distribuovaná. Pre zabezpečenie správnej činnosti siete Bitcoin každý z klientov obsahuje kópiu verejnej účtovnej knihy označovanej ako `blockchain`. Všetky novo vzniknuté a overené transakcie sú následne vložené do `blockchainu`. Bližšiemu popisu `blockchainu` je venovaná kapitola 2.3.

Aby transakcia mohla byť úspešne vykonaná musia byť o nej informovaný všetci účastníci siete. Každý klient môžu overiť či klient skutočne disponuje prostriedkami, ktoré chce presunúť na cieľovú adresu. Účastníci systému sú informovaný za pomoci `broadcastového` vysielania, že klient chce vykonať platobnú transakcií.



Obr. 2.2: Štruktúra správy popisujúca transakciu v sieti Bitcoin.

Pri vytváraní nových transakcií klient na vstup novo vzniknutej transakcie nastaví niektoré zo svojich `UTXO`, a na výstup nastaví minimálne jednu adresu príjemcu. Na základe toho vzniká zoznam navzájom prepojených transakcií.

V prípade, že by transakcia mala práve jeden výstup, tak by bolo umožnené len preposlať celú čiastku zo vstupu na výstup, a preto sa najčastejšie stretávame s transakciami, kde počet vstupov je minimálne jeden a práve dva výstupy. Jeden z výstupov je používaný pre adresáciu príjemcu a druhý výstup slúži pre poslanie zvyšných prostriedkov na účet odosielateľa.

Rozdiel medzi súčtami množstva finančných prostriedkov na všetkých vstupoch a výstupoch transakcie je použitý ako poplatok za vykonanie transakcie. Finančné prostriedky definujúce poplatky sú pripisované klientom označovaným ako baníci.

Tabuľka 2.1 popisuje štruktúru správy pomocou, ktorej sú klienti informovaný o požiadavku pre realizáciu transakcie. Obzvlášť významnými časťami transakcie sú položky `in[]`, `out[]` a `lock_time`.

Názov položky		Typ (Veľkosť[B])	Popis
version		int (4)	Verzia formátu transakcie.
#in		var_int (1-9)	Počet vstupných transakcií.
in[]	hash	uint256 (32)	Dvojitý SHA-256 hash transakcie.
	index	uint (4)	Index vstupnej transakcie.
	signature_script_length	var_int (1-9)	Dĺžka podpisujúceho skriptu.
	signature_script	CScript (variabilná)	Podpis transakcie.
	nSequence	uint (4)	Sekvnenčné číslo.
#out		var_int (1-9)	Počet výstupných transakcií.
out[]	value	var_int (1-9)	Množstvo Satoshi, ktoré chce užívateľ odoslať.
	pk_script_length	var_int (1-9)	Dĺžka pk_script.
	pk_script	CScript (variabilná)	Skript definujúci podmienky, ktoré musia byť splnené, aby požadovaná čiastka bola zaúčtovaná.
lock_time		uint (4)	Časové razítko alebo číslo bloku, v ktorom je transakcia zahrnutá.

Tabuľka 2.1: Štruktúra správy popisujúca transakciu.

Tabuľka 2.2 obsahuje popis dátových typov používaných v správach popisujúcich transkcie v sieti Bitcoin.

Dátový typ (veľkosť)	Popis
int (4B)	Celé čísla.
var_int (1-9B)	Celé čísla s premenlivou dĺžkou.
uint256 (32B)	Nezmamienkové celé čísla.
CScript	Skript v skriptovacom jazyku `Bitcoin Script`.

Tabuľka 2.2: Popis použitých dátových typov.

Položka `in[]` predstavuje vektor vstupných transakcií, ktorých UTXO budú použité pre realizáciu transakcie. U každého zo vstupov nachádzame položky `hash` a `index` za pomoci, ktorých sú identifikované vstupné transakcie v UTXO klienta. Obzvlášť dôležitou

položkou v prípade vstupných transakcií je položka `signature_script`, ktorá obsahuje podpis transakcie. Podpisy transakcie sú následne používané pre overovanie toho či klient skutočne disponuje finančnými prostriedkami, ktoré uviedol na vstupoch novo vznikajúcej transakcie.

Položka `out[]` predstavuje vektor výstupných transakcií. U každej výstupnej transakcie nachádzame položky `value` a `pk_script`. Hodnota položky `value` udáva množstvo Satoshi, ktoré chce užívateľ odoslať na cieľovú adresu. Hodnota položky `pk_script` obsahuje `Bitcoin Script`, za pomoci ktorého je definovaný spôsob zaúčtovania prijatých finančných prostriedkov na strane príjemcu.

Podpisovanie transakcií

Za účeľom overovania toho, že odosielateľ skutočne disponuje finančnýmy prostriedkami, ktoré tvoria vstup novo vzniknutej transakcie je používaný mechanizmus kryptografického podpisovania. Podpis transakcie je prenášaný v položke `signature_script`. Táto položka obsahuje podpis transakcie a adresu príjemcu. Na základe obsahu uvedenej položky môže každý z klientov v sieti overiť či odosielateľ má skutočne právo manipulovať s výstupmi transakcií, ktoré tvoria vstupy novej transakcie.

Podpisovanie vstupov transakcií je realizované podľa nasledujúceho algoritmu:

1. Pomocou algoritmu `SHA-256` sa z verejného kľúču príjemcu a hashu predchádzajúcej transakcie, s ktorou odosielateľ môže manipulovať vytvorí hash.

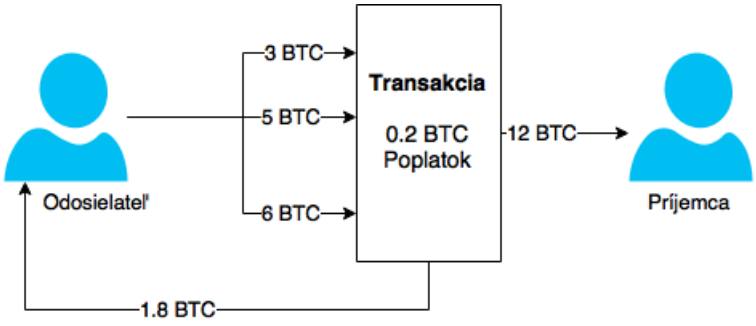
2. Použitím kryptografického algoritmu `ECDSA` sa hash vytvorený v predchádzajúcom kroku podpíše súkromným kľúčom odosielateľa.

3. Výsledná hodnota z predchadzajúceho kroku sa uloží spolu s verejným kľúčom príjemcu do položky `signature_script`.

Príklady transakcií

Nasledujúcich niekoľko odstavcov popisuje dva príklady vykonávania transakcií v sieti Bitcoin. Za pomoci prvého príkladu je znázornený spôsob akým klient pracuje so svojimi `UTXO` a ako sú realizované poplatky za vykonanie transakcie. Druhý príklad predstavuje ilustráciu jednoduchej transakcie vo formáte `raw` pomocou, ktorého je prenášaná informácia o transakcii po sieti Bitcoin.

Obrázok 2.3 ilustruje situáciu, kedy odosielateľ chce poslať príjemcovi 12 BTC. Keďže 12 BTC odosielateľ nevlastní na práve jednom výstupe z predchadzajúcich transakcií, tak na vstup transakcie použije 3 výstupy prechadzajúcich transakcií (celkovo 14 BTC). Pomocou jedneho výstupu definuje že príjemcovi pošle 12 BTC druhým výstupom zabezpečí že 1.8 BTC sa mu vráti ako zostatok po vykonaní uvedenej transakcie. Rozdiel súčtov vstupov a výstupov o hodnote 0.2 BTC bude použitých ako poplatok za vykonanie transakcie.



Obr. 2.3: Príklad transakcie medzi dvoma klientami systému Bitcoin.

Tabuľka 2.3 popisuje príklad transakcie systému Bitcoin kde je použitý práve jeden vstup a jeden výstup. Ako bolo spomenuté vyššie vstup transakcie je previazaný pomocou hashu na výstup predchádzajúcej transakcie. Vstup obsahuje taktiež položku ktorá obsahuje podpis transakcie. Výstup definuje práve jedného príjmcu. Pomocou transakcie sa má presunúť 50 000 satoshi (50 BTC) na účet príjemcu. Príkladom adresy použitej v sieti Bitcoin je adresa 404371705fa9bd789a2fcd52d2c580b65d35549d. Tvar tejto adresy je uvedený v hexadecimálnom tvare.

Vstup	
Predchádzajúca tx	f5d8ee39a430901c91a5917b9f2dc19d6d1a0e9cea205b00 9ca73dd04470b9a6
Index výstupu	0
`signature_script`	04502206e21798a42fae0e854281abd38bacd1aeed3ee3738d9e1446 618c4571d1090db022100e2ac980643b0b82c0e88ffdfec6b64e3e6ba 35e7ba5fdd7d5d6cc8d25c6b241501
Výstup	
Množstvo satoshi	5000000000
`pk_script`	OP_DUP OP_HASH 160404371705fa9bd789a2fcd52d2c580b65d3 5549d OP_EQUALVERIFY OP_CHECKSIG

Tabuľka 2.3: Príklad transakcie.[12]

2.2.2 Typy transakcií a realizácia zaúčtovania

Každy z výstupov transakcie je sprevádzaný so skriptom zapísaným v jazyku `Bitcoin Script`[10]. Skriptovací jazyk `Bitcoin Script` je používaný v sieti Bitcoin pre realizáciu zúčtovania. Za pomoci jazyka `Bitcoin Script` je možné definovať rôzne podmienky pre realizáciu zaúčtovania prichádzajúcej transakcie ako napr. Zaúčtuj ďalší pondelok a iné.

Jazyk `Bitcoin Script` je jednoduchým turingovsky neúplným skriptovacím jazykom a je inšpirovaný skriptovacím jazykom `Forth`. Spracovávanie skriptu je vykonávané rovnako ako v prípade jazyka `Forth`, čiže za pomoci zásobníkového automatu. Samotný skript je následne interpretovaný z ľava do prava. Skript v jazyku `Bitcoin Script` je tvorený reťazcom operačných kódov a operandov. Pomocou operačných kódov je možné definovať rôzne podmienky, realizovať aritmeticke operácie, porovnávať hodnoty operandov a iné.

Samotný proces overovania transakcie pozostavá troch krokov:

1. Hodnoty položiek `signature_script` a `pk_script` sú vložené na vrchol zásobníka.

Krok	Zásobník (vrchol vpravo)	Skript
1.	Prázdny	<sig><pubKey>OP_DUP OP_HASH160 <pubKeyHash>OP_EQUALVERIFY OP_CHECKSIG
2.	<sig><pubKey>	OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG
3.	<sig><pubKey><pubKey>	OP_HASH160, <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG OP_CHECKSIG
4.	<sig><pubKey><pubHashA>	<pubKeyHash>OP_EQUALVERIFY OP_CHECKSIG
5.	<sig><pubKey><pubHashA> <pubKeyHash>	OP_EQUALVERIFY OP_CHECKSIG
6.	<sig><pubKey>	OP_CHECKSIG
7.	true alebo false	

Tabuľka 2.5: Overovanie transakcií typu Pay-to-PubkeyHash

2. Spracovávanie skriptu v jazyku `Bitcoin Script` za pomoci zásobníkového automatu.
3. V prípade, že po vykonaní 2. kroku vrchol zásobníka obsahuje logickú hodnotu `true` (logická pravda) tak potom je transakcia overená v opačom prípade je neoverená.

Na základe obsahu skriptu v jazyku `Bitcoin Script` je možné definovať rôzne scenáre, ako majú byť transakcie spracovávané. Na základe toho rozlišujeme základné typy Bitcoin transakcií `Pay-to-PubkeyHash`, `Pay-to-Script-Hash` a transakcie typu `Coinbase`.

Pay-to-PubkeyHash

Jedná sa o najčastejšie používaný typ transakcie, ktorý je určený pre prenos finančných prostriedkov na jednu alebo viacero Bitcoin adres. Tabuľka 2.4 popisuje obsah položiek `signature_script` (vstup transakcie) a `pk_script` (výstup transakcie) v prípade tohoto typu transakcie.

pk_script:	OP_DUP OP_HASH160 <pubKeyHash>OP_EQUALVERIFY OP_CHECKSIG
signature_script:	<sig><pubKey>

Tabuľka 2.4: Obsah položiek `signature_script` a `pk_script` v prípade transakcie typu Pay-to-PubkeyHash.

Položka `signature_script` obsahuje dvojicu položiek a to podpis odosielateľa (< sig>) a verejný kľúč príjemcu (<pubKey>). Položka `pk_script` obsahuje skript popisujúci spôsob overovania transakcie a položku <pubKeyHash>, ktorá obsahuje Bitcoin adresu príjemcu. Proces overovania môžme popísať pomocou nasledujúcich 7 krokov:

1. Položky `signature_script` a `pk_script` sa spoja.
2. Položky `signature_script` a `pk_script` sú vložené na vrchol zásobnika.

3. Položka na vrchole zásobníka *pubKey* sa zduplikuje.
4. Z hodnoty na vrchole zásobníka sa vypočíta 160 bitový hash. Výsledok sa uloží na vrchol zásobníka.
5. Na vrchol zásobníka sa vloží pole <`pubKeyHash`>.
6. Porovnajú sa hodnoty dvoch vrchných položiek zásobníka.
7. Overí sa podpis a na vrchol zásobniku sa uloží výsledok overovania.

Pay-to-Script-Hash - P2SH adresy

Motiváciou pre vytvorenie `P2SH` adres bol presun zodpovednosti za splnenie podmienok z odosielateľa transakcie na príjemcu. Umožnujú vytvoriť odosielateľovi ľubovoľne zložitú transakciu použitím 20 B hashu.

`Pay-to-Script-Hash` poskytuje možnosť vytvárania ľubovoľne komplikovaných transakcií, narozdiel od `Pay-to-PubkeyHash`, kde je napevno definovaný formát položiek `singnature_script` a `pk_script`. Na základe toho je možné vytvárať transakcie rôznych druhov.

Coinbase

Je to typ transakcie, za pomoci, ktorého je baník odmeňovaný za svoju prácu v prospech siete Bitcoin. Transakcie tohoto typu majú práve jeden vstup. Položka `signature_script` je nahradená položkou `coinbase`, ktorej význam a obsah nie je definovaný. Často v položke `coinbase` nachádzamé informáciu o zložitosti generovania bloku.

U každého výstupu transakcie je v položke `pk_script` obsiahnuté pravidlo pre realizáciu zaúčtovania. Pravidlo pre realizáciu zúčtovania môže byť napríklad v nasledujúcej podobe: `OP_DUP OP_HASH160` <`pubKeyHash`> `OP_EQUALVERIFY OP_CHECKSIG`. Spracovávanie tohoto `Bitcoin Scriptu` je realizované podobne ako v prípade transakcie typu Pay-to-PubkeyHash.

2.2.3 Double-spending problém

Je to problém, ktorého výsledkom je minutie určítého finančného prostriedku viac ako jeden krát. V sieti Bitcoin by mohol tento problém nastať tak, že klient by po odoslaní transakcie vložil ďalší verejný kľúč príjemcu a podpis transakcie by bol realizovaný rovnako ako v prípade štandardnej transakcie. Výsledkom opakovania uvedeného procesu by bolo nekontrolované generovanie nových finančných prostriedkov, ktoré by v konečnom dôsledku spôsobilo infláciu[5]. Systém Bitcoin je voči problému Double-spending ochránený pomocou verejnej účtovnej knihy, ktorej kópiu ma uložený každý z klientov. Popis verejnej účtovnej knihy nájdeme v podkapitole 2.3.

V prípade štandardného bankového systému je ochrana voči problému `Double-spending` riešená pomocou centrálnej autority, ktorá vykonáva dohlad nad zrealizovanými transakciami. V prípade siete Bitcoin každý z účastníkov súhlasí s použitím protokolu pre spracovávanie transakcií. Každý z klientov dokáže overiť dôverihodnosť transakcie pomocou verejného kľúču príjemcu.

2.3 Evidencia vykonaných transakcií - blockchain

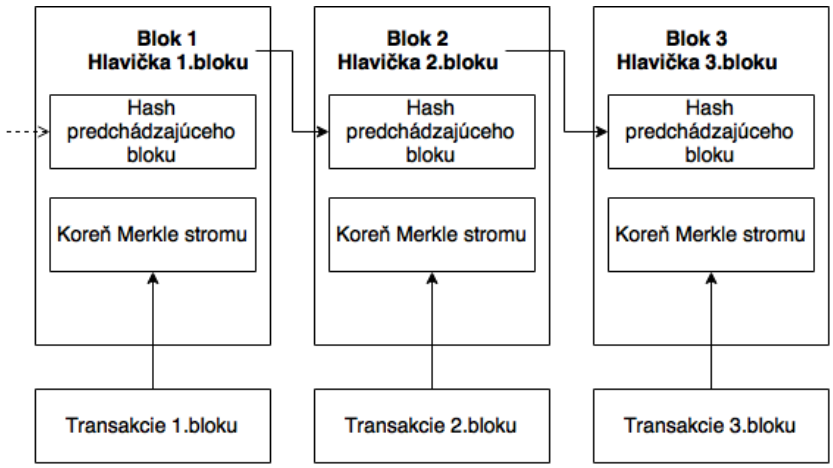
Evidencia transakcií v systéme Bitcoin je realizovaná za pomoci distribuovanej databáze, ktorá je označovaná ako `blockchain` [14]. `Blockchain` predstavuje zdieľanú účtovnu knihu (anglicky *ledger*), v ktorej sú evidované všetky úspešne zrealizované platby.

Každý z klientov disponuje svojou lokálnou kópiou `blockchainu`, ktorého veľkosť je v súčastnosti (20.1.2016) približne 56 GB. Za správu obsahu `blockchainu` zodpovedajú klienti označovaný ako baníci, ktorý za svoju prácu dostávajú odmenu vo forme finančných prostriedkov.

Na základe obsahu takejto databáze užívateľia môžu zistiť stav finančných prostriedkov na akejkoľvek adrese v minulosti. Každá novo vzniknutá transakcia, ktorá je úspešne overená je pridaná do `blockchainu`. V prípade vzniku požiadavku na vykonanie transakcie klienti označovaný ako baníci sa usilujú transakciu zaradiť do jedného z blokov, čo predstavuje položku v `blockchaine`. Proces pridávania nových transakcií do blokov a následného pripojenia vzniknutého bloku do `blockchainu` je označovaný ako ťažba. Proces ťažby je popísaný v podkapitole 2.4.

2.3.1 Bloky

`Blockchain` je implementovaný ako spojovaný zoznam (`linked list`). Položky zoznamu sú označované ako bloky[3]. Previazanosť medzi jednotlivými blokmi je realizovaná pomocou hahshov aktuálneho a predchádzajúceho bloku. Pomocou týchto dvoch hashov je realizovane previazanie jednotlivých blokov `blockchainu`. Bloky v `blockchaine` sú usporiadané podľa času ich vzniku. Každý z blokov potvrdzuje transakcie vykonané v predchadzajúcich blokoch. Pomocou obrázku 2.4 je ilustrovaná schéma `blockchainu`. Prvý blok v `blockchaine` je označovaný ako Genesis blok [6].



Obr. 2.4: Štruktúra spoločnej účtovnej knihy `Blockchain`.

V tabuľke 2.6 je obsiahnutý popis významu jednotlivých položiek v bloku.

Veľkosť[B]	Názov	Význam
4	Magic no	Hodnota tejto položky je vždy 0xD9B4BEF9.
4	Blocksize	Veľkosť bloku.
80	Blockheader	Hlavička bloku. Popísaná nižšie.
1-9B	Transaction counter	Počet transakcií v bloku.
Variabilná	Transactions	Zoznam transakcií v bloku

Tabuľka 2.6: Popis obsahu bloku `blockchainu`.

V každom z blokov je obsiahnutý zoznam transakcií vykonaných za posledných cca. 10 minút. To, že bloky v `blockchaine` vznikajú približne každých 10 minút je zabezpečené za pomoci položky označovanej ako `Bits`, ktorej popis je uvedený v ďalšej časti textu.

V prípade, že by potenciálny útočnik chcel zmeniť obsah jedného z blokov, tak by musel zmeniť obsah ďalších blokov `blockchainu`, a to je vo výsledku výpočetne veľmi náročné. Na základe toho, že každý z klientov má uloženú kópiu účtovnej knihy, tak by takúto aktivitu dokázali klienti detekovať.

2.3.2 Hlavičky bloku v blockchaine.

Následujúca podsekcia je zameraná na popis významu položiek hlavičky bloku. Tabuľka 2.7 obsahuje popis jednotlivých položiek. Najvýznamnejšími položkami sú položky `hashPrevBlock` a `hashMerkleRoot`. Položka `hashPrevBlock` definuje referenciu na predchadzajúci blok a položka `hashMerkleRoot` obsahuje hash `Merkle Tree` pre overovanie pravosti obsahu bloku. Z pohľadu tvorby `blockchainu` ako reťazcu blokov sú kľúčové položky `hashPrevBlock` a hodnota hashu hlavičky bloku.

Veľkosť[B]	Názov	Význam
4	Version	Číslo verzie bloku
32	hashPrevBlock	Hash predchádzajúceho bloku.
32	hashMerkleRoot	Hash všetkých transakcií v bloku.
4	Time	Časová značka bloku.
4	Bits	Cieľ výpočtu hashu Merkle Tree.
4	Nonce	Číselná hodnota, ktorá je inkrementovaná.

Tabuľka 2.7: Popis obsahu hlavičky bloku.

Z pohľadu tvorby nových blokov `blockchainu` sú veľmi dôležité položky `Time`, `Bits` a `Nonce`.

Time

Položka `Time` obsahuje 4B časové razítko. Časové razítko je používané pre overovanie platnosti bloku a zabráňuje modifikácii bloku. Táto položka je zároveň používana pri výpočte hashu hlavičky bloku, kde predstavuje náhodnosť.

Bits

Položka `Bits` je 4B hodnota udávajúca aktuálny cieľ. Hodnota tejto poľožky je zdieľaná v rámci celej siete Bitcoin. Položka `Bits` udáva maximálnu prípustnú hodnotu hashu hlavičky nového bloku. Platí, že čím je hodnota položky `Bits` nižšia, tak je náročnejšie vytvorenie nového bloku. Keďže výpočtová sila baníkov postupom častu

narastá, tak potom hodnota tejto položky musí byť premenlivá. Aktualizácia hodnoty položky `Bits` je vykonávaná po vytvorení každých 2016 blokov, tak aby nové bloky vznikali približne po 10 minútach.

S aktuálnym cieľom súvisí vlastnosť označovaná ako obťiažnosť (anglicky Difficulty). Obtiažnosť určuje náročnosť vytvorenia nového bloku. Hodnota obtiažnosti je daná pomerom maximálneho možného cieľu a hodnoty aktuálneho cieľu.

Nonce
Položka `Nonce` obsahuje 4B číslo a spolu s časovým razítkom uloženým v položke `Time` predstvuje zdroj náhodnosti, ktorý je využitý pri výpočte hashu hlavičky bloku.

Hash bloku[4] je vyočítaný za pomoci dvojnásobného použitia hashovacieho algoritmu SHA-256 nad obsahom hlavičky bloku. Pre výpočet hashu hlavičky bloku sú obzvlášť dôležité hodnoty v položkách `Bits` a `Nonce`. Výpočet hashu bloku môžme popísať nasledujúcim výrazom.

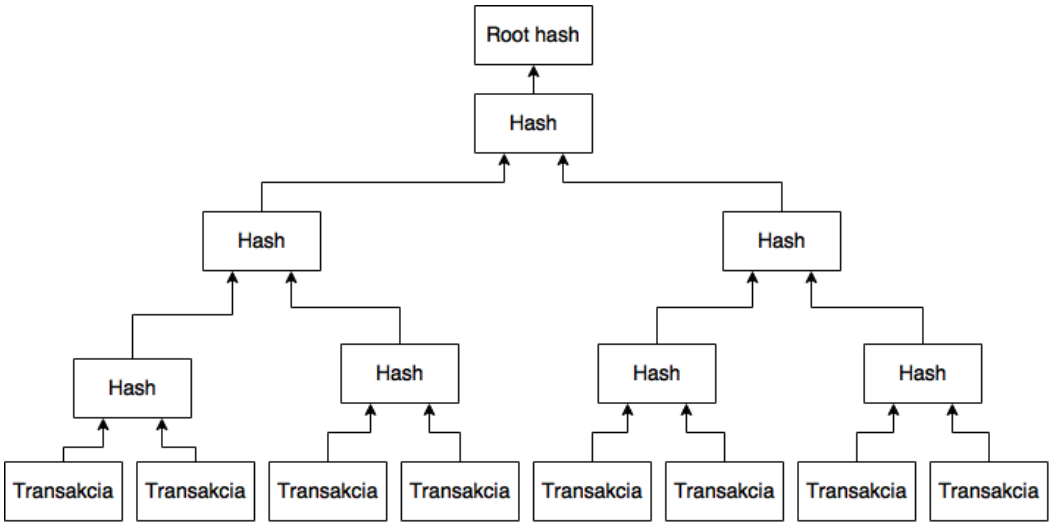
$$hash\_bloku = SHA\text{-}256(SHA\text{-}256(hlavicka\_bloku))$$

2.3.3 Dátová štruktúra Merkle Tree

`Merkle Tree`[21] je stromová dátová štruktúra hashov, ktorá môže byť obecne n-árna. `Merkle Tree` sa používa pre efektívne a bezpečne overovanie obsahu rozsiahlych dátových štruktúr. Koreň stromu obsahuje hash celého stromu. V prípade, že hodny hashov koreňov dvoch `Merkle Tree` sa zhodujú tak môžme prehlásiť, že dáta, ktoré sú zabezpečené za ich pomoci sú identické.

Implementácia `Merkle Tree` je realizovaná tak, že listové uzly stromu reprezentujú dáta, ktoré chceme zabezpečiť a ostatné uzly obsahujú hashe synovských uzlov. V prípade, že počet listových uzlov je nepárny tak potom je posledný listový uzol zduplikovaný.

V sieti Bitcoin je `Merkle Tree` implementovaný ako binárny strom. Pre výpočet hashu synovských uzlov je používaný kryptografický algoritmus `SHA-256` pričom je použité dvojité hashovanie. Spôsob výpočtu `Merkle Tree` používaného v sieti bitcoin je naznačený na obrázku 2.5.



Obr. 2.5: Spôsob výpočtu `Merkle Tree` v sieti Bitcoin.

2.4 Tvorba nových finančných prostriedkov

V prípade štandardného bankového systému finančné prostriedky generujú vlády krajín. Ako už vieme z predchádzajúcej časti textu sieť Bitcoin je decentralizovaná, a tak tu nenachádzame žiadnu centrálnu autoritu, ktorá by takúto činnosť vykonávala. Tvorba nových finančných prostriedkov v systéme Bitcoin je realizovaná za pomoci klientov označovaných ako baníci, ktorý za svoju činnosť v prospech siete dostávajú odmenu, ktorá predstavuje nové finančné prostriedky vstupujúce do platobnej siete. Hlavnou úlohou baníkov je vytváranie nových blokov, ktoré budú pripojené do `blockchainu`. Proces zostavovania nových blokov je často označovaný ako ťažba alebo anglicky *mining*.

Podkapitola čerpá z [7] a [8].

2.4.1 Princíp ťažby

Novo vzniknuté transakcie, ktoré ešte neboli zahranuté v žiadnom bloku `blockchainu` zbierajú baníci a následne sa snažia zhromaždené transakcie zaradiť do nového bloku, ktorý bude pridaný do blockchainu. V prípade, že sa baníkovi podarí zostaviť blok ako prvému v sieti tak je mu udelená odmena vo forme niekoľkých BTC. Aktuálna výška odmeny je 25 BTC za jeden zostavený blok. Výška odmeny sa znižuje s pribúdajúcim časom. Keďže platobná mena má deflačný charakter tak v obehu systému Bitcoin bude maximálne 21 000 000 BTC.

Výška odmeny pre baníkov za zostavené bloky v `blockchaine` je približne každé štyri roky znižovaná. Tabuľka 2.8 znázorňuje spôsob akým sa bude meniť výška odmeny pre baníkov.

Dátum dosiahnutia	**Odmenová éra**	**Výška odmeny BTC/Blok**	**Rok (Predpokládaný)**
1.3.2009	1	50	2009
22.4.2010			2010
28.1.2011			2011
14.12.2011			2012
28.11.2012	2	25	2013
9.10.2013			2014
11.8.2014			2015
29.7.2015			2016
	3	12.5	2017
			2018
			2019
			2020
	4	6.25	2021
			2022
			2023
			2024

Tabuľka 2.8: Výška odmeny pre baníkov v sieti Bitcoin.

Vývojový diagram 2.6 znázorňuje algoritmus na základe, ktorého sa baníci pokúšajú o zostavenie nového bloku, ktorý bude pridaný do `blockchainu`. Jedná o algoritmus za pomoci, ktorého je riešený zložitý matematický problém.

Výpočet hashov hlavičiek blokov je veľmi výpočtovo náročný proces. Náročnosť tohoto algoritmu narastá s výpočtovou silou celej siete Bitcoin. V ranných dobách siete Bitcoin baníci využívali za účelom ťažby štandardné PC, avšak v súčasnej dobe je realizácia ťažby na štandardných PC neefektívna, a tak sa začali využívať špeciálne HW zariadenia ako napríklad programovateľné polia (FPGA) alebo čipy, ktoré sú špeciálne určené na túto činnosť (ASIC).

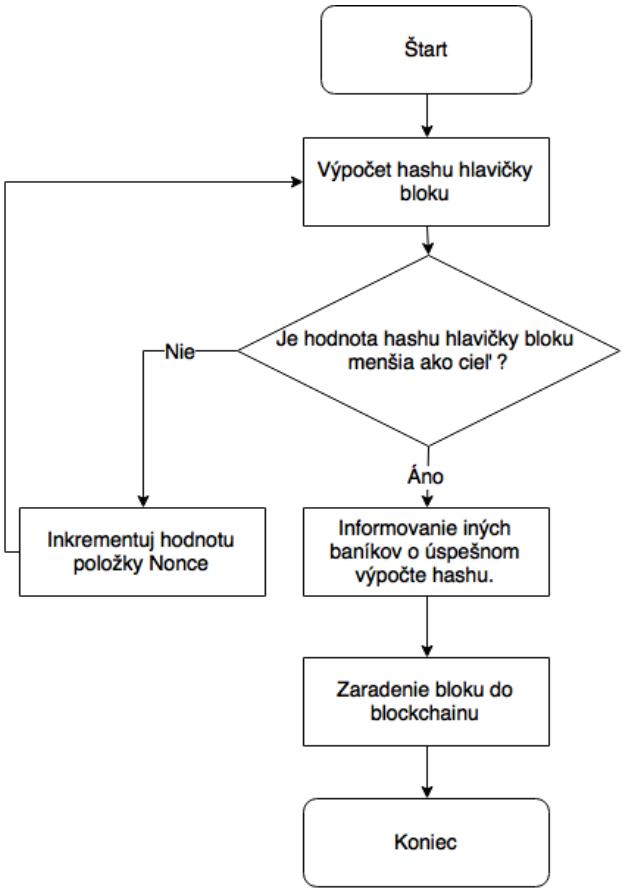
Podľa toho či baníci vykonávaju ťažbu samostatne alebo v skupine s ďalšími baníkmy rozlišujeme dva spôsoby ťažby označované ako sólo ťažba a banícke zoskupenie.

Sólo ťažba

V tomto prípade baník vykonáva ťažbu sám a celá odmena je priradená jemu. Ako už vieme tak náročnosť výpočtu nových hashov blokov sa neustále zvyšuje, a tak samostatná ťažba jedného baníka je často krát neefektívna.

Banícke zoskupenia

Banícke zoskupenie je spôsob ťažby, ktorý je realizovaný za pomoci viacerých baníkov. Každý z baníkov rieši podčasť úlohy samostatne. Následne sú výsledky od baníkov v rámci zoskupenia spojené a je rozhodnuté či bol nájdený nový blok. V prípade úspechu je odmena rozdelená medzi účastníkov zoskupenia [8].



Obr. 2.6: Vývojový diagram tvorby nových prostriedkov v sieti Bitcoin.

2.4.2 Overovanie transakcií

Aby bola transakcia v sieti Bitcoin úspešne vykonaná, tak musí byť overená. Ukazateľom overenia transakcie je počet blokov, ktoré sa nachadzajú v `blockchaine` za blokom, v ktorom je obsiahnutá overovaná transakcia. Hodnota šesť predstavuje prah pre označenie transakcie za úspešne overenú.

Po zaradení transakcie do bloku je nastavená hodnota tohoto čítača na hodnotu jedna. Pridaním každého ďalšieho bloku do `blockchainu` je inkrementovaná hodnota čítača overení transakcie. V prípade, že bola dosiahnutá prahová hodnota (šesť) tak je transakcia overená. V inom prípade sa musí počkať na vznik ďalších blokov, ktoré potrvrdia overovanú transakciu.

2.5 Protokol Bitcoin

Pre komunikáciu s ostatnými klientami v sieti Bitcoin je používaný binárný protokol `Bitcoin Protocol`. Protokol je využívaný za účelom validácie pôvodu finančných prostriedkov, ich presuny a podpisovanie transakcií.

Dáta komunikačného protokolu `Bitcoin Protocol` sú prenášané za pomoci transportného protokolu `TCP`. Produkčná verzia komunikačného protokolu využíva ku svojej činnosti port s číslom 8333. Na sieti testnet (sieť určená pre vývojarov) komunikácia pomocou protokolu `Bitcoin Protocol` prebieha na TCP porte 18333.

Tabuľka 2.9 popisuje typy správ používaných v protokole `Bitcoin Protocol`. V uvedenej tabuľke sa nenachádza popis správ označených ako *deprecated* a metód používaných pre získavanie obsahu `blockchainu`. Popis týchto metód bude uvedený v ďalšej časti textu.

Správa	**Význam**
version	Žiadosť o spojenie s klinetom.
verack	Odpoveď na správu typu version.
getaddr	Žiadosti o získanie zoznamu peerov.
addr	Odpoveď na getaddr.
ping	Overenie živosti klienta.
pong	Odpoveď na správu typu ping.
inv	Oznámenie o znalosti objektu.
filter	Bloomove filtre.
alert	Upozornenie pre všetkých klientov siete.
reject	Zamietnutie prijatej správy

Tabuľka 2.9: Správy protokolu `Bitcoin Protocol` bez nepodporovaných typov správ a správ vzťahujúcich sa k onsahu `blockchainu`.

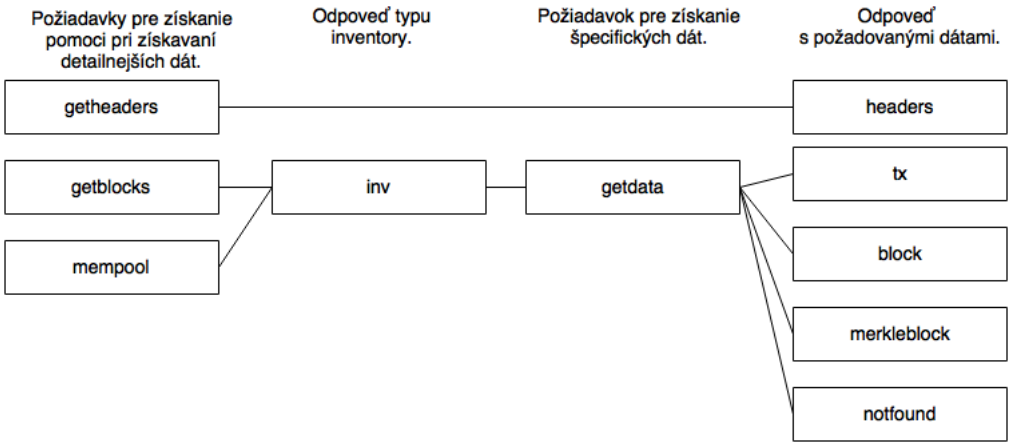
Obrázok 2.7 znázorňuje sieť správ, ktoré sú používané pre získavanie dát vzťahujúcich sa ku transakciám a blokom.

Metóda `getblocks` slúži pre získanie zoznamu idnetifikátorov blokov od určitého bloku. Ku podobnému účelu je určená metóda `getheaders`. Hlavným rozdieľom medzi metódami `getblocks` a `getheaders` je v tom, že odpoveď u metódy `getheaders` obsahuje maximálne 2000 položiek a odpoveď na druhu obsahuje maximálne 500 položiek.

Metóda `mempool` je určená pre získavanie zoznamu transakcií, ktoré ešte neboli spracované.

Odpoveď na uvedené metódy okrem metódy `getheaders` je vo forme správy typu `inv`, ktorá obsahuje zoznam (inventár) identifikátorov jednotlivých dát (transakcií a blokov). Položky v spŕave typu `inv` označujeme ako `inventory` a u každej položky je uvedený typ a identifikátor konkrétneho záznamu (napr. identifikátor bloku, transakcie atď.) vo forme hashu.

Po spracovaní tela správy typu `inv` je možné za pomoci správy typu `getdata` získať kompletné dáta vzťahujúce sa ku blokom a transakciam. Napríklad pomocou správy `block` sú prenášané informácie o obsahu požadovaného bloku v `blockchaine`.



Obr. 2.7: Sieť dátových správ protokolu `Bitcoin Protocol`.

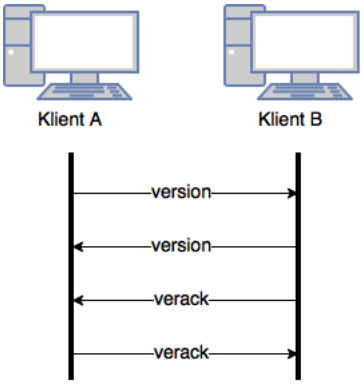
2.5.1 Pripojenie do siete Bitcoin

Potom, ako je klient spustený musí sa pripojiť ku zvyšku P2P siete. Klient po spustení nepozná IP adresu žiadneho aktívneho klienta, a tak musí zistiť IP adresu aktívných klientov v P2P sieti a následne sa k nim pripojiť, aby mohol s P2P sieťou komunikovať.

Pre získavanie IP adries aktývných užívateľov sú používané DNS dotazy na špecifické domény označované ako `DNS seeds`. Správa obsahu DNS seedu je realizovaná nezávysle na sebe za pomoci komunity. Aktualizácia obsahu `DNS seedu` môže byť realizovaná dynamicky alebo staticky. Dynamická správa `DNS seedu` spočíva v replikácií IP adres aktývnych zariadení.

Po získaní IP adresy iného aktívneho užívateľa. Klient inicializuje komunikáciu s klientom za pomoci správy typu `version`. Príjemca správy odpovedá pomocou správy typu `verack`.

Pripojenie ku ďalšiemu peerovi siete je realizované zaslaním `version` správy, ktorá obsahuje nasledovné informácie: verzia protokolu, blok a aktuálný čas. Klient, ku ktorému sa chceme pripojiť odpovedá správou `version`. Následné obaja klienti si vzájomné vymenia správy `verack` čím sa spojenie vytvorý. Obrázok 2.8 znázorňuje sled správ v prípade, že sa klient chce pripojiť ku inému klientovi siete Bitcoin.



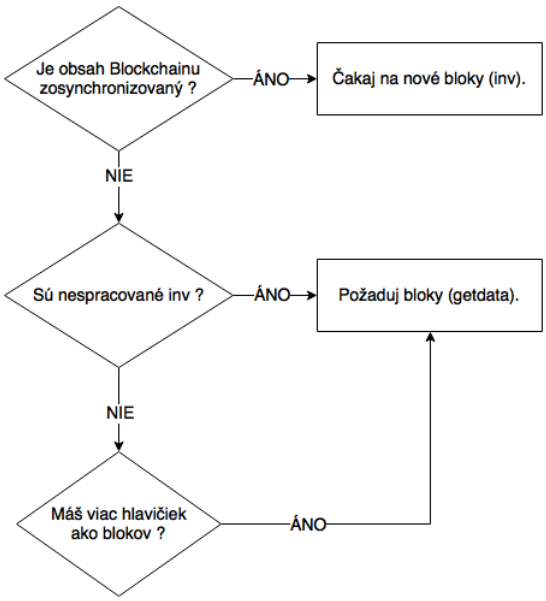
Obr. 2.8: Pripojenie ku klientovi Bitcoin siete.

Získavanie IP adries ďalších susedov je realizované za pomoci zasielania správ typu `getaddr`. Na správy `getaddr` odpovedajú klienti za pomoci správ typu `addr`. Komunikácia pomocou správ `getadr` a `addr` musí byť realizované každých 30 minút. V inom prípade spojenie je považované za neaktívne.

Pre overovanie či je spojenie aktívne je možné použiť kombináciu správ `ping - pong`.

2.5.2 Vytvorenie lokálnej kópie blockchainu.

Predtým ako klient môže validovať doposial nepotvrdené transakcie a novo vytvorené bloky musí stiahnuť a zvalidovať obsah všetkých blokov od bloku označovaného ako `Genesis blok`. Proces realizujúci uvedenú činnosť označujeme ako *Initial Block Download (IBD)*. Táto činnosť sa musí vykonať po každom opätovnomm pripojení klienta do P2P siete len s tým rozdieľom, že sa aktualizujú dáta od posledného uloženého bloku v lokálnej kópii `blockchainu`. Proces IBD môže byť realizovaný ako `Blocks-First` alebo `Headers-First`. Obrázok 2.9 znázorňuje princíp metódy `Headers-First`, ktorá je používaná v aktuálnej verzii oficiálneho Bitcoin klienta (Bitcoin Core verzie 0.9.3).

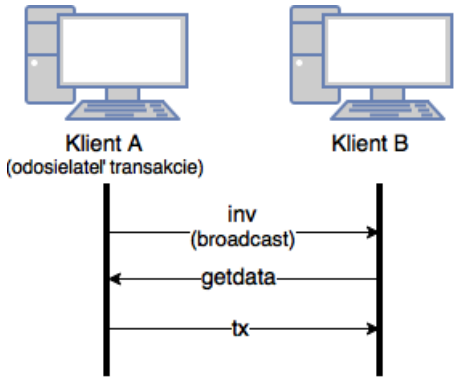


Obr. 2.9: Princíp procesu `IBD` realizovaného metódou `Headers-First` v sieti Bitcoin.

2.5.3 Šírenie informácie o novo vzniknutej transakcií

Ako už bolo spomenuté v predchádzajúcej časti textu práce po vzniku požiadavku na vykonanie transakcie klient informuje všetkých klientov siete o požiadavku pre vykonanie transakcie. Tvorca transakcie odošle takúto informáciu všetkým priamo pripojeným klientom. Príjemcovia takejto správy overia validitu transakcie. V prípade, že transakcia je validná tak informáciu o vzniknutej transakcie pošlu všetkým priamo pripojeným stanicam. Týmto spôsobom za zabezpečí, že v sieti sa rozširuje informácia len o validných transakciách. Plný klienti by si takéto novo vzniknuté transakcie mali uložiť do zoznamu aktuálne nepotvrdených transakcií. Takéto transakcie následne budú zaradené do nových blokov.

Za účelom šírenia informácie o požiadavku pre vykonanie transakcie sú využívané správy typov `inv`, `getdata` a `tx`. Diagram 2.10 znázornuje sled správ zasielaných počas informavia klienta o novo vzniknutej transakcii.



Obr. 2.10: Šírenie informácie o novo vzniknutej transakcii v sieti Bitcoin.

2.5.4 Informovanie o vzniku nového bloku v blockchaine.

V prípade, že baník zostavý nový blok tak za pomoci broadacstoveho vysielania rozšíri informáciu o novom bloku všetkým peerom siete pomocou jednej z nasledujúcich dvoch metód.

Unsolicted Block Push
: Baník priamo rozošle svojim plným peerom informáciu o zostavenom bloku pomocou správy `block`.

Standard Block Relay
: Baník fungujúci ako štandardný relay uzol zašle správu `inv` každému peerovi s inventárom vzťahujúcim sa k novému bloku.

2.6 Alternatívne kryptomeny

Úspech kryptomeny Bitcoin viedol ku vzniku ďalších kryptomien. Podľa štatistiky [13] trhových hodnôt kryptomien je ku dňu 3.12.2015 najhodnotejšiou kryptomenou mena Bitcoin.

Prvá trojica najhodnotnejších kryptomen je tvorená menami Bitcoin, Ripple a Litecoin. Spoločným znakom uvedených kryptomien je ich decentralizácia (neexistencia centrálnej autority) a využitie krpytografických algoritmov pre zabezpečenie vykonávania platobných transakcií.

Nasledujúcich niekoľko odstavcov sa venuje stručnému popisu ďalších populárných kryptomien Ripple [17] a Litecoin [16]. V popise je kladený dôraz na popis základných princípov a rozdieľov od kryptomeny Bitcoin.

Ripple

Jedná sa o open-source platobný systém založený spoločnosťou CoinBase. Základnou jednotkou finančných prostriedkov systému je XRP. V systéme Ripple bude v obehu maximálne 100 000 000 XRP. Narozdiel od siete Bitcoin v sieti Ripple nie je obsah `blockchainu` tvorený za pomoci ťažby ale pomocou iteratívneho procesu. Rýchlosť overenia transakcie v systéme zabere niekoľko sekund čo je omnoho menej ako v systéme Bitcoin kde uvedený proces je vykonávaný niekoľko minút. Sieť Bitcoin zaznamenáva len presuny finančných prostriedkov v sieti narozdiel od toho systém Ripple dokáže zaznamenávať presuny akéhokoľvek druhu informácie z čoho vyplíva, že systém dokáže bez väčších problémov spolupracovať s inými kryptomenami.

Litecoin

Litecoin je platobný systém pracujúci s finančnými prostriedkami v jednotkách označovaných Litecoin (LTC). Podobne ako v prípade systému Bitcoin sa tu stretávame s procesom ťažby. Systém Litecoin bude pracovať s maximálne 84 000 000 LTC. Narozdiel od systému Bitcoin je v procese ťažby používaný algoritmus Scrypt. Bloky v sieti Litecoin sú vytvárané každých 2.5 minút. Uvedená vlastnosť implikuje vyššiu rýchlosť vykonávania transakcií ako v siete Bitcoin kde bloky sú vytvárané každých 10 minút.

2.7 Zhrnutie

Kapitola sa venovala popisu princípov a algoritmov použitých v platobnej sieti Bitcoin.

Úvod kapitoly bol venovaný popisu základných princípov siete Bitcoin. Ďalšie podkapitoly sa postupne venovali jednotlivým detailom implementácie siete Bitcoin.

Najskôr sme sa venovali popisu spôsobu adresácie užívateľov v sieti Bitcoin. Následne boli popísané spôsoby implementácie Bitcoin peňaženiek, za pomoci, ktorých je realizované komunikácia so sieťou Bitcoin. Za pomoci Bitcoin peňaženiek sú platobnej sieti zasielané požiadavky pre realizáciu platobných transakcií. Postupne kapitola prešla k popisu realizácie transakcií v sieti Bitcoin. Boli definované tri základné typy transakcií a proces zaúčtovania transakcie za pomoci jazyka `Bitcoin Script`. V ďalšej časti textu sme sa venovali popisu toho, ako sú transakcie evidované vo verejnej účtovnej knihe označovanej ako `blockchain`. Ďalšia časť kapitoly bola venovaná popisu komunikačného protokolu `Bitcoin Protocol` za pomoci ktorého klienti P2P siete Bitcoin komunikujú. V závere kapitoly sme sa venovali stručnému popisu alternatívnych kryptomen. Pričom dôraz bol kladený na popis hlavných rozdieľov oproti mene Bitcoin.

Kapitola 3

Návrh a implementácia nástroja forenznej analýzy v sieti Bitcoin

Túto kapitolu môžme rozdeliť do celkovo štyroch podkapitol.

V prvej podkapitole je predstavená motivácia pre implementáciu nástrojov podporujúcich forenznú činnosť v prostredí siete Bitcoin (prípadne iných kryptomen).V ďalšej podkapitole si popíšeme požiadavky kladené na forenzné nástroje pracujúce s kryptomenou Bitcoin. Následne sa kapitola venuje popisu súčasného stavu v prostredí nástrojov pre podporu forenznej činnosti v sieti Bitcoin. Dôraz je kladený na popis existujúcich riešení a ich vzájomné porovnávanie. Pozornosť v tejto podkapitole je venovaná popisu možnosti, ako zlepšiť existujúce nástroje. Posledná podkapitola obsahuje zhrnutie kapitoly.

V kapitole sa čerpá zo [15].

3.1 Motivácia pre implementáciu nástrojov

Ako už bolo v texte spomenuté, platobná sieť Bitcoin a krpytomeny vo všeobecnosti sú často využívané na základe svojich vlastností pre realizáciu platobných transakcií osobami, ktoré sa dopúšťajú trestnej činnosti.

Siete kryptomen sú často využívané pre realizáciu platieb na čiernych trhoch a pranie špinavých peňazí. Hlavným dôvodom prečo je takáto nekalá aktivita koncentrovaná do takýchto sieti je fakt, že chovanie na platobných sieťach je anonymné alebo minimálne pseudoanonymné. Potom je veľmi problematické vystopovať skutočné osoby, ktoré sa podieľali na platobných transakciách.

Ďalším dôvodom používania sieti kryptomen je absencia centrálnej autority (vlády, banky a ďalšie), ktorá by kontrolovala chod platobnej siete. Na základe toho sa tu nestretávame so zmrazovaním účtov. Dôsledkom toho je aktívny vývoj v oblasti softvérových nástrojov pre podporu forenznej činnosti v sieti Bitcoin.

3.2 Aktuálny stav

Drvivá väčšina nástrojov pre podporu forenznej činnosti v sieti Bitcoin je implementovaná vo forme webovej aplikácie.

Medzi zaujímavé nástroje patria: Bitcoin Block Explorer [1], Wallet Explorer [2], BitIodine

[1] Zdroj: https://blockchain.info/

[2] Zdroj: https://www.walletexplorer.com/

project [3].

Za najzaujímavejší z týchto nástrojov považujem nástroj Wallet Explorer, ktorý adresy automaticky zoskupuje do skupín, ktoré definujú adresy obsiahnuté v Bitcoin peňaženke. Pozitívnou vlastnosťou Wallet Exploreru je to, že nástroj peňaženkám priraďuje verejný identifikátor. Na profilových stránkach jednotlivých adries (peňaženiek) je možné sledovať zmeny stavov finančných prostriedkov na adrese (adresách). Nespornou výhodou Wallet Exploreru je rýchlosť. Za nedostatok nástroja Wallet Explorer považujem absenciu vizualizácie transakcií a to, že nástroj neprirađuje informácie o identitách jednotlivým adresám ale len celým peňaženkám.

Nástroj Bitiodine project vznikol ako výsledok diplomovej práce Michele Spagnuolo. Aplikácia umožňuje identifikovať adresy, ktoré tvoria Bitcoin peňaženku. Pomocou aplikácie je možné následné vyhľadávať transakcie medzi konkrétnymi adresami (peňaženkami) ale aj kombináciami adries a peňaženiek. Ako nedostatok nástroja Bitiodine považujem veľmi vysokú reakčnú dobu na dotazy. V prípade dotazov vzťahujúcich sa k peňaženkám s viac ako 1000 adries je odpoveď získaná za niekoľko minút a v častých prípadoch aplikácia úplne zamrzne. Pre prácu s malými peňaženkami je však aplikácia relatívne efektívna. Pre získavanie podrobných informácií o transakciách je potreba použiť ďalší nástroj.

Nástroj Bitcoin Block Explorer rozširuje funkcionalitu štandardného `blockchain` prehliadača o funkcie pomocou, ktorých je možné vizualizovať transakcie, zobrazovať identity osôb v sieti a iné. Nástroj sa pokúša prideľovať ku jednotlivým adresám identity (pseudoidentity). Informácie o identitách (pseudoidentitách) nástroj získava za pomoci parsovania obsahu internetových stránok, na ktorých sú uvedené informácie o adresách. Do databáze známych adries je možné pridávať nové záznamy taktiež ručne. Ako nedostatok nástroja Bitcoin Block Explorer považujem to, že nástroj nepracuje s peňaženkami ale len jednotlivými adresami, a tak nie je možné realizovať komplexnejšiu analýzu nad obsahom `blockchainu`. V prípade grafickej vizualizácie transakcií vidím nedostatok v tom, že z grafu nie je možné sa priamo prekliknúť na profil adresy a zároveň v grafe chýba informácia o identitách (pseudoidentitách) vzťahujúcich sa k adresám.

Ako problém popisovaných nástrojov vidím to, že žiaden z nástrojov neintegruje funkcie pre podporu forenznej činnosti v sieti Bitcoin do jedného komplexného nástroja. Napríklad projekt BitIodine je síce možné použiť pre dohľadanie adries, ktoré pravdepodobne patria jednému účastníkovi, avšak pre získavanie potrebných informácií o transakciách, identitách (pseudoidentitách) je nutné použiť iný nástroj.

Kľúčovou úlohou vyvíjaného nástroja bude komplexná podpora realizácie forenznej činnosti v platobnej sieti kryptomeny Bitcoin s ohľadom na možné rozšírenie podpory pre ďalšie populárne kryptomeny. Vyvíjaná aplikácia bude spĺňovať všetky funkčné požiadavky definované diagramom prípadov užitia na obrázku 3.1.

3.3 Návrh nástroja

Podkapitola sa zaoberá návrhom nástroja pre podporu realizácie forenznej činnosti v sieti Bitcoin. V prvom kroku návrhu aplikácie bola realizovaná analýza kľúčových požiadavkov kladených na implementovaný nástroj.

Základným požiadavkom kladeným na nástroje pre podporu realizácie forenznej činnosti v sieti Bitcoin je možnosť prehľadávania obsahu verejnej účtovnej knihy. V tom je zahrnuté

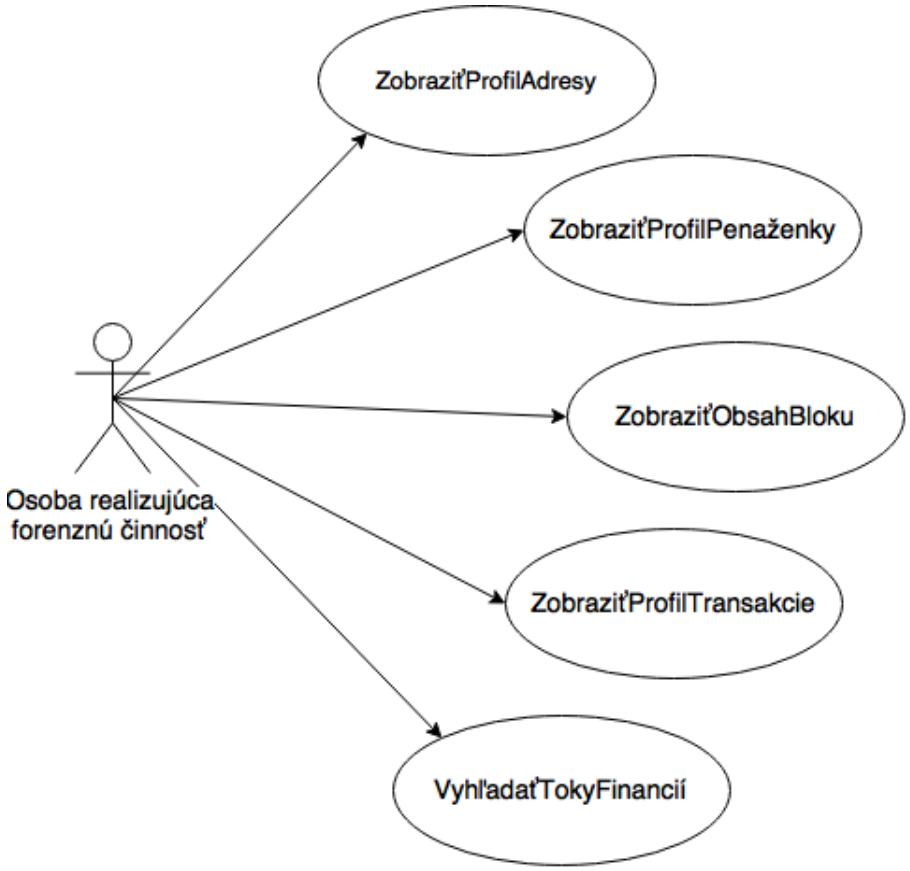
[3] Zdroj: https://bitiodine.net/

vyhľadávanie transakcií realizovaných užívateľmi, sledovanie výšky zostatkov finančných prostriedkov, možnosť vystopovania tokov finančných prostriedkov v platobnej sieti atď.

Dôležitým požiadavkom kladeným na systém pre podporu forenznej činnosti v sieti Bitcoin je identifikácia zoznamu adries, s ktorými klient disponuje. Takýto zoznam adries definuje peňaženku užívateľa siete. Následne je vhodné, aby aplikácia umožňovala zobrazovať ich profil.

Ďalším dôležitým požiadavkom kladeným na nástroje pre podporu forenznej činnosti v sieti Bitcoin je deanonymizácia aktivity na platobnej sieti. Táto aktivita spočíva v priradení identít a pseudoidentit skutočných osôb užívateľom pracujúcim so systémom. Získané informácie o identitách budú následne zobrazované jak vo výpisoch transakcií, tak profilov adries a Bitcoin peňaženiek.

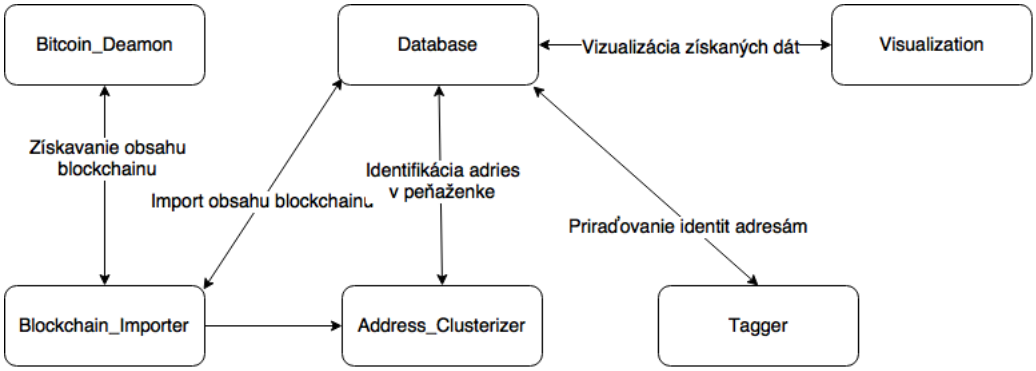
Obrázok 3.1 obsahuje diagram prípadov použitia vzťahujúcich sa ku aplikácií pre podporu forenznej činnosti v sieti Bitcoin a iných kryptomenách.



Obr. 3.1: Diagram prípadov použitia.

Logická schéma aplikácie

Nástroj pre podporu bude implementovaný vo forme webovej aplikácie. Aplikácia bola navrhnutá ako sada celkovo šiestich logických komponent. Každá z komponent vykonáva určité činnosti, za ktorých vykonanie je zodpovedná. Logická schéma aplikácie je reprezentovaná za pomoci diagramu 3.2. Obdĺžniky v diagrame reprezentujú komponenty a orientované hrany medzi nimi znázorňujú spôsob komunikácie medzi komponentami.



Obr. 3.2: Diagram komponent navrhovaného systému.

Význam komponent navrhovanej aplikácie je nasledovný:

Bitcoin_Deamon
Táto komponenta nebude mnou implementovaná. Komponenta reprezentuje oficiálneho Bitcoin klienta, ktorý bude komunikovať so zvyškom platobnej siete za účelom získania obsahu nových blokov `blockchainu`. Klient bude spustený v režime `deamon`[4] s podporou ovládania pomocou Bitcoin JSON-RPC API. Následne za pomoci uvedeného API bude komponenta `Blockchain_Importer` získavať obsah jednotlivých blokov `blockchainu`.

Database
Komponenta predstavuje databázu, do ktorej sú ukladané všetky potrebné informácie pre chod systému. V tejto databáze bude uložený obsah `blockchainu`, predovšetkým základné informácie o blokoch a transakciách v nich obsiahnutých. Ďalej bude databáza obsahovať informácie o adresách. Medzi tieto informácie patrí: priradenie adresy do Bitcoin peňaženky, identita atď.

Blockchain_Importer
Jedná sa o komponentu, ktorá je zodpovedná za import nových blokov `blockchainu` do databáze (komponenta `Database`), s ktorou bude vyvíjaný systém pracovať. Počas chodu komponenty bude komponenta zadávať pokyny bloku `Address_Clusterizer` pre zistenie príslušnosti adries do Bitcoin peňaženiek.

Address_Clusterizer
Komponenta bude realizovať identifikáciu adries, ktoré sú pravdepodobne spravované za pomoci jednej Bitcoin peňaženky. Zistené informácie bude komponenta predávať bloku `Database`, ktorý zabezpečí ich uloženie pre ďalšiu prácu.

Tagger
Komponenta sa bude snažiť za pomoci verejne dostupných zdrojov priradiť Bitcoin adresám minimálne pseudoidentitu osoby, ktorá danú adresu vlastní. Získané identity bude následne predávať komponente `Database`, ktorá identity zaznamená pre ďalšiu prácu.

Visualization
Predstavuje blok, ktorý realizuje vizualizovanie zaznamenaných dát (obsah blokov,

[4] Aplikácia bežiaca na pozadí

profily adries, obsahy peňaženiek atď.) v prehľadnej podobe, pričom umožňuje pohodlné filtrovanie výsledkov. Tento blok predstavuje vstupný bod, pomocou ktorého osoba realizujúca forenznú činnosť komunikuje so systémom.

3.4 Použité technológie

Obsah `blockchainu` v súčasnej dobe (január 2016) zaberá približne 56 GB diskového priestoru a jeho veľkosť s pribudajúcim časom narastá. V dôsledku toho bolo potrebné vybrať databázový systém, ktorý dokáže efektívne pracovať s veľkým objemom dát s dôrazom kladeným na rýchlosť vyhľadávania, pretože operácia vyhľadávania bude najčastejšie realizovaná. S ohľadom na možnosť rozšírenia podpory pre prácu s ďalšími kryptomenami bol vybraný NoSQL databáza MongoDB[5]. Tri kľúčové vlastnosti, na ktorých si databáza MongoDB zakláda sú: vysoký výkon, vysoká dostupnosť a automatická horizontálna škálovateľnosť databáze. V databázovom systéme MongoDB nenachádzame pevnú štruktúru ukladaných dát (schéma dát), a tak je možné za chodu aplikácie meniť. Na základe toho bude možné v budúcnosti bez väčších problémov použiť vytvorenú databázu pre ukladanie dát vzťahujúcich sa k ďalším kryptomenám. V databáze MongoDB sú dáta reprezentované za pomoci JSON dokumentov, ktoré sú interne ukladané databázovým systémom vo formáte BSON.[6]

Pre samotnú aplikáciu bolo potrebné zvoliť vhodnú implementačnú platformu. Implementovaný nástroj forenznej analýzy predstavuje webovú aplikáciu. Serverová časť implementovanej aplikácie je napísaná v skriptovacom jazyku PHP vo verzii 5.6. Pre efektívnejší vývoj aplikácie v jazyku PHP bol vybraný framework Laravel[7], ktorý rieši veľa problémov spojených s tvorbou webových aplikácií, ako je napríklad ošetrenie vstupov, práca s formulármi, stránkovanie a poskytuje veľa ďalších podporných prostriedkov pre vývoj moderných webových aplikácií. Aplikácie vytvárané pomocou frameworku Laravel sú štruktúrované podľa návrhového vzoru MVC (Model-View-Controller). Externé knižnice používané implementovanou aplikáciou sú spravované pomocou balíčkovacieho systému Composer[8]. Za pomoci uvedeného balíčkovacieho systému boli do projektu pridané ďalšie externé PHP knižnice. Jedná sa o knižnicu unirest[9], ktorá bola použitá pre implementáciu JSON-RPC klienta, ktorý komunikuje s Bitcoin klientom.

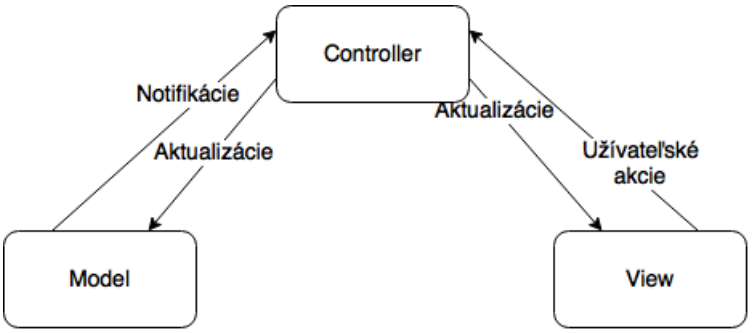
Klientská časť aplikácie bola realizovaná za pomoci CSS/JS frameworku Bootstrap[10] a javascriptového frameworku jQuery[11]. Javascriptová knižnica D3.js[12] bola použitá pre implementáciu vizualizácie tokov finančných prostriedkov v sieti Bitcoin.

Implementačné rysy použitej architektúry MVC

Keďže aplikácia je implementovaná na základe návrhového vzoru MVC tak tu nachádzame striktné rozdelenie implementácie na tri logické časti a to modely (M) - entity realizujúce prácu s dátovým modelom, zobrazovače (V) - realizujú vizualizáciu dát z modelov a radiče (C), v ktorých je zapuzdrená aplikačná logika. Výhodou takéhoto rozdelenia implementácie

[5] Viac informácií na: https://www.mongodb.com
[6] Formát BSON rozširuje formát JSON o podporu ďalších dátových typov.
[7] Viac informácií na: https://laravel.com/
[8] Viac informácií na: https://getcomposer.org
[9] Viac informácií na: http://unirest.io/php.html
[10] Viac informácií na: http://getbootstrap.com
[11] Viac informácií na https://jquery.com
[12] Viac informácií na: https://d3js.org

umožňuje ľahšie realizovať úpravy jednotlivých častí, ktoré sa naviac takmer nepremietajú do iných častí. Diagram na obrázku 3.3 ilustruje architektúru MVC. Dôležitou vlastnosťou architektúry MVC, ktorú je vidieť na uvedenom diagrame je nezávislosť modelov od iných častí aplikácie.



Obr. 3.3: Diagram architektúry MVC.

Komponenty `Blockchain_Importer`, `Address_Clusterizer` a `Tagger` implementovanej aplikácie sú realizované v rámci modelov. Komponenta `Visualisation` je realizovaná za pomoci implementácie radičov a zobrazovačov, ktoré pracujú s dátami uloženými v modeloch.

3.5 Implementácia dátového modelu v prostredí MongoDB

Nasledujúca podkapitola sa venuje popisu implementácie dátového modelu v prostredí databázového systému MongoDB.

Keďže v NoSQL databázach nenachádzame schému dát, tak je štruktúra jednotlivých dokumentov ukladaných v MongoDB kolekciách popísaná pomocou príkladov v JSON dokumentoch. Hodnoty v týchto príkladov definujú dátové typy hodnot uložených v odpovedajúcich položkách dokumentov.

3.5.1 Reprezentácia obsahu blockchainu v MongoDB

Podkapitola sa venuje popisu toho, ako je obsah `blockchainu` ukladaný v databáze implemtovaného nástroja forenznej analýzy.

Bloky

Informácie o blokoch tvoriacich `blockchain` sú ukladané do MongoDB kolekcie pomenovanej ako `blocks`. Jeden dokument v kolekcií popisuje práve jeden blok `blockchainu`.

Schéma jednotlivých dokumentov v kolekcii `blocks` je naznačená nasledovne:

```
{
    "hash"              : String,
    "previousblockhash" : String,
    "nextblockhash"     : String,
    "size"              : Number,
    "height:"           : Number,
    "time"              : Number,
    "sum_of_inputs"     : Double,
    "sum_of_outputs"    : Double,
```

```
    "sum_of_fees"       : Double,
    "transactions"      : Number
}
```

Popis jednotlivých položiek dokumentu je nasledujúci. Položka `hash` slúži ako identifikátor bloku v `blockchaine`. `Hash` predstavuje unikátny identifikátor bloku v `blockchaine`, a preto je u tejto položky nastavený index typu `unique`, za pomoci ktorého je zabezpečené, že do kolekcie nebude pridaný ďalší blok s rovnakým hashom a nevzniknú tak duplicitné záznamy v kolekcii. Položka `previousblockhash` obsahuje identifikátor predchádzajúceho bloku v `blockchaine`. Položka `nextblockhash` obsahuje hash nasledujúceho bloku v `blockchaine`. Za pomoci položky `height` je uchovávaná pozícia bloku v rámci `blockchainu`. Čas vzniku bloku je uchovávaný v položke `time` a je zadávaný vo forme UNIX timestampu. Ku každému z blokov sú ďalej ukladané sumárne charakteristiky a to `transactions` - počet transakcií obsiahnutých v bloku, `sum_of_inputs` - celkové množstvo finančných prostriedkov na vstupoch transakcií v rámci daného bloku, `sum_of_outputs` - celkové množstvo finančných prostriedkov na výstupoch transakcií v rámci daného bloku a charakteristika `sum_of_fees`, ktorá uchováva informáciu o celkovej výške poplatkov za transakcie zahrnuté v bloku.

Transakcie

Informácie o transakciách realizovaných v platobnej sieti Bitcoin sú ukladané do MongoDB kolekcie s názvom `transactions`. Jeden dokument v kolekcii popisuje práve jednu transakciu v sieti.

Schéma dokumentov v kolekcii `transactions` je naznačená nasledovne:

```
{
    "txid"          : String,
    "blockhash"     : String,
    "time"          : Number,
    "inputsOutputs" : Array,
    "blocktime"     : Number,
    "sumOfInputs"   : Double,
    "sumOfOutputs"  : Double,
    "sumOfFees"     : Double,
    "uniqueInputs"  : Number
}
```

Popis jednotlivých položiek dokumentov kolekcie `transactions` je nasledujúci. Položka `txid` obsahuje jednoznačný identifikátor transakcie v rámci siete Bitcoin. Hodnota tejto položky v rámci kolekcie musí byť unikátna, a tak je tejto položke nastavený index typu `unique`, za pomoci čoho je zabránené duplicitnému zaznamenaniu informácií o transakcii. Za pomoci pomoci položky `blockhash` je realizované previazanie transakcie s blokom. Položke `blockhash` bol nastavený databázový index, aby bolo dosiahnuté čo najrýchlejšie vyhľadávanie transakcií podľa príslušnosti transakcií do bloku. Položka `time` obsahuje informáciu o čase vzniku transakcie a je ukladaná v podobe UNIX timestampu. Čas, kedy transakcia bola zaradená do bloku je uchovávaný v položke `blocktime`. V prípade dokumentov popisujúcich transakcie sú sumárne charakteristiky transakcie uchovávané v nasledujúcich položkách `sumOfInputs` - súčet vstupov transakcie, `sumOfOutputs` - súčet výstupov transakcie, `sumOfFees` - poplatok za vykonanie transakcie a položku `uniqueInputs`, ktorá udáva počet unikátnych adries, ktoré boli použité na vstupoch transakcie. Položka inputsOutputs obsahuje pole dokumentov popisujúcich vstupy a výstupy transakcie.

Štruktúru dokumentov popisujúcich vstupy transakcie, ktoré sú obsiahnuté v položke `inputsOutputs` je možné naznačiť nasledovne:

```
{
    "type"      : 1,
    "txid"      : String,
    "vout"      : Number,
    "value"     : Double,
    "addresses" : Array
}
```

Položka `type` obsahuje príznak rozhodujúci o tom, či sa jedná o vstup alebo výstup transakcie. V prípade vstupu je nastavená na hodnotu „1". Položka `addresses` obsahuje pole adries vstupujúcich do realizácie transakcie za pomoci daného vstupu. Množstvo finančných prostriedkov, ktoré vstupovalo do transakcie na uvedenom vstupe je uchované v položke `value`. Pomocou kombinácie položiek `txid` a `vout` je identifikovaný výstup inej transakcie, ktorý sa vzťahuje ku uvedenému vstupu.

Štruktúru dokumentov popisujúcich výstupy transakcie, ktoré sú obsiahnuté v položke `inputsOutputs` je možné naznačiť nasledovne:

```
{
    "type"      : 2,
    "addresses" : Array,
    "value"     : Double,
    "n"         : Number,
    "spent"     : true,
    "spentTxid" : String,
    "spentTs"   : Number
}
```

Položka `type` obsahuje príznak rozhodujúci o tom, či sa jedná o vstup alebo výstup transakcie, a tak je nastavená na hodnotu „2". Množstvo finančných prostriedkov, ktoré sa presúvali na výstup je uchovávané v položke `value`. Adresy, na ktore smeroval konkrétny výstup transakcie sú uložené v položke `addresses`. Položka `n` identifikuje číslo výstupu transakcie. Položka `spent` obsahuje informáciu o tom, či výstup transakcie bol použitý pre realizáciu ďalšej transakcie. V prípade, že výstup transakcie bol minutý, tak dokument popisujúci výstup transakcie obsahuje položky `spentTxid` - identifikácia transakcie, v ktorej boli minuté finančné prostriedky a položku `spentTs`, ktorá uchováva čas, kedy boli finančné prostriedky minuté.

Nad položkou `addresses`, ktorá sa nachádza jak vo vstupoch tak výstupoch transakcie bol definovaný databázový index, za pomoci ktorého je zabezpečené rýchle vyhľadávanie transakcií, ktoré sa vzťahujú k určitým adresám. Takýto index napríklad zrýchlil operáciu, počas ktorej sa určuje stav finančných prostriedkov na adrese (adresách) užívateľa.

3.5.2 Práca s adresami a peňaženkami v MongoDB

Pre prácu s Bitcoin adresami je používaná kolekcia `addresses`. Adresy môžu byť zoskupované do peňaženiek. Peňaženky sú reprezentované za pomoci dokumentov v kolekcii `clusters`.

Štruktúra dokumentov v kolekcii `clusters` je jednoduchá, pretože obsahujú len dokumenty, v ktorých je obsiahnutý generovaný identifikátor dokumentu v MongoDB kolekciách označovaný ako _id. V budúcnosti je sem možné doplniť ďalšie položky, ktoré by popisovali peňaženku. Napríklad sa môže jednať o informácie o type peňaženky (zmenáreň, hazard atď.).

Do kolekcie `addresses` sú zaznamenávane informácie o adresách, ktoré boli zaradené do Bitcoin peňaženiek a tie, u ktorých boli získané informácie o identite užívateľa pracujúceho s danou adresou.

Schéma dokumentov uložených v kolekcii `addresses` je naznačená nasledovne:

```
{
    "address" : String,
    "cluster" : ObjectId,
    "balance" : Double,
    "tags"    : Array
}
```

Položka `address` obsahuje Bitcoin adresu, ku ktorej sa viaže záznam v kolekcii. Hodnota položky `address` musí byť v každom dokumente kolekcie `addresses` jedinečná, a tak je na tejto položke definovaný databázový index typu `unique`, ktorým sa zamedzí vytvoreniu duplicitných záznamov. Dokument v kolekcii môže obsahovať položku `cluster` obsahujúcu identifikátor peňaženky, do ktorej adresa patrí.

V položke `balance` je uchovávaný aktuálny stav finančných prostriedkov na adrese. Môže sa zdať, že je informácia redundantná, ale v tomto prípade táto redundancia dát má zmysel, pretože za pomoci predpočítaných stavov účtov sa urýchľuje výpočet stavou na adresách v rámci peňaženky atď. Predpočítaný stav prostriedkov na adrese v prípade veľkých peňaženiek má obrovské opodstatnenie, pretože je práca so zobrazovaním profilov adres/peňaženiek veľmi urýchlená. Nie je potreba vykonávať opakované agregačné výpočty nad obrovskou kolekciou adries. Viac informácií ohľadom použitia predpočítaných čiastkach na adresách bude uvedených v ďalších častiach textu.

Položka `tags` obsahuje pole dokumentov, ktoré popisujú získané informácie o identite užívateľa siete.

Štruktúru jednotlivých dokumentov v poly tags je možné naznačiť nasledovne:

```
{
    "tag"        : String,
    "url"        : String,
    "source"     : Number,
    "sourceType" : Number
}
```

Položka `tag` obsahuje získaný verejný identifikátor užívateľa. Napríklad sa môže jednať o prezývku na diskusnom fóre, emailovú adresu alebo iný identifikátor. V položke `url` je uchovávaná adresa webovej stránky, z ktorej bol získaný identifikátor. Môže to byť profil na diskusnom fóre, kontaktná stránka e-shopu alebo iný druh webovej stránky. Položky `source` a `sourceType` sú použité pre rozlíšenie jednotlivých zdrojov odkiaľ bol stiahnutý identifikátor.

3.5.3 Realizácia komunikácie medzi DB a systémom

Za účelom zabezpečenia komunikácie medzi DB a nástrojom forenznej analýzy boli implementované triedy `MongoConnection` a trieda `BaseMongoModel`. Za pomoci uvedených tried boli implementované ďalšie triedy, ktoré zapuzdrujú prácu s dátami uloženými v MongoDB kolekciách. Skupinu takýchto tried označujeme ako databázové `modely`.

Trieda `MongoConnection` reprezentuje pripojenie do databáze MongoDB. Trieda bola implementovaná na základe návrhového vzoru singleton[13]. V súbore pomenovanom ako .env

[13] Za behu aplikácie môže vzniknúť maximálne jeden objekt triedy.

sú uložené parametre pre pripojenie sa ku databáze MongoDB. Uvedený súbor je mimo verzovací systém a nie je možné k nemu pristúpiť z vonku aplikácie.

Trieda `BaseMongoModel` predstavuje abstraktnú triedu, za pomoci ktorej sú definované triedy reprezentujúce objekty v jednotlivých kolekciách DB systému MongoDB. Dôležitou metódou triedy `BaseMongoModel` je metóda `collection`, za pomoci ktorej je realizovaný výber MongoDB kolekcie, s ktorou sa bude pracovať. Za použitia dedičnosti od triedy `BaseMongoModel` boli implementované triedy: `BlockModel` - zapuzdruje prácu s blokmi v `blockchaine`, `TransactionModel` - implementuje prácu s transakciami realizovanými v sieti, `AddressModel` - zapuzdruje prácu s adresami siete Bitcoin a trieda `ClusterModel` realizujúca prácu so zhlukmi adries (adresy v peňaženkách užívateľov siete Bitcoin).

Implementácia databázových model bola kľúčová pre implementáciu ďalších časti nástroja forenznej analýzy v sieti Bitcoin, pretože sú nimi aktívne využívané.

3.6 Implementácia bloku Blockchain_Importer

Nasledujúca kapitola je zameraná na popis implementácie bloku `Blockchain_Importer`, ktorého úlohou je import nových blokov z `blockchainu` do databáze nástroja forenznej analýzy.

Implementovaná komponenta `Blockchain_Importer` ku svojej činnosti využíva komponentu `Bitcoin_Deamon`, ktorá reprezentuje oficiálneho Bitcoin klienta spusteného v režime deamon s podporou ovládania za pomoci JSON-RPC API. Komponenta `Bitcoin_Deamon` za svojho behu synchronizuje obsah lokálnej kópie `blockchainu` so zvyškom siete.

Komponenta `Blockchain_Importer` komunikuje s komponentou `Bitcoin_Deamon` pomocou JSON-RPC API, ktorého podpora je integrovaná v oficiálnom Bitcoin kliente.

Komponenta `Blockchain_Importer` ku svojej činnosti využíva nasledujúce RPC metódy poskytované Bitcoin klientom:

GetInfo
: Metóda poskytujúca rôzne informácie o spustenom Bitcoin klientovi a stave Bitcoin siete. Pomocou metódy `GetInfo` je získavaná informácia o počte blokov, ktoré sú uložené v lokálnej kópii `blockchainu` Bitcoin klienta.

GetBlockHash
: Za pomoci metódy `GetBlockHash` je získavaný hash bloku podľa jeho poradového čísla v `blockchaine`.

GetBlock
: Metóda `GetBlock` je určená pre získanie informácií o bloku so zadaným hashom.

GetRawTransaction
: Za pomoci metódy `GetRawTransaction` sú získavané informácie o transakcii so zadaným hashom.

Implementácia komponenty `Blockchain_Importer` je sústredená do metódy `handle` triedy `DownloadBlocks`, ktorá predstavuje automatickú úlohu spúšťanú každú hodinu. Frekvenciu spúšťania automatickej úlohy je možné meniť v triede `Kernel` z menného priestoru `App\Console`. V triede je taktiež možné nastaviť maximálny počet blokov `blockchainu` stiahnutých počas jedného spustenia automatickej úlohy. V prípade prvotného spustenia

nástroja forenznej analýzy je potrebné plánovanie automatickej úlohy pozastaviť a zvýšiť maximálny limit na počet stiahnutých blokov za jeden beh. Pomocou týchto krokov je možné stiahnuť kompletnú kópiu `blockchainu` za niekoľko málo behov.

Algoritmus podľa, ktorého je implementovaná komponenta `Blockchain_Importer` je možné popísať pomocou troch krokov:

1. Na začiatku algoritmu je získaný hash bloku, ktorého obsah ma byť importovaný. V prípade, že neexistuje blok, ktorý nebol importovaný tak algoritmus končí. Hlash bloku, ktorý má byť stiahnutý je určený najskôr na základe hodnoty položky `nextblockhash` posledného bloku v kolekcii `blocks`. V prípade, že hodnota položky je prázdna, tak je využitá RPC metóda `GetInfo`, pomocou ktorej je získaná informácia o počte existujúcich blokov v lokálnej kópií `blockchainu` Bitcoin klienta. V prípade, že v sieti Bitcoin existuje viac blokov, ako je importovaných v databáze nástroja forenznej analýzy, tak algoritmus končí. V inom prípade je získaný hash nasledujúceho bloku za pomoci metódy `GetBlockHash`. Ako vstupný parameter uvedenej metódy je počet importovaných blokov v databáze nástroja forenznej analýzy zvýšený o hodnotu jedna. Získaný hash je uložený do položky `nextblockhash` u posledne importovaného bloku v kolekcii `blocks`.

2. Po úspešnom získaní hashu bloku, ktorý ma byť stiahnutý je zavolaná RPC metóda `GetBlock` pre získanie dát popisujúcich blok v `blockchaine`. Následne sú do databáze forenzného nástroja importované transakcie zahrnuté v danom bloku. Dáta popisujúce jednotlivé transakcie sú získavané pomoci RPC metódy `GetRawTransaction`. Počas importu jednotlivých transakcií sú dopočítavané sumárne charakteristiky popisujúce transakciu (súčet vstupov transakcie, súčet výstupov transakcie, výška poplatku, počet unikátnych adries na vstupe transakcie atď.) a charakteristiky bloku (súčet vstupov všetkých transakcií v bloku, súčet výstupov všetkých transakcií v bloku a celková výška poplatkov). V prípade, že na vstupoch, alebo výstupoch transakcie sú uvedené adresy obsiahnuté v kolekcii `addresses` tak je navýšená, alebo znížená hodnota položky `balance` na základe toho či sa jednalo o vstup, alebo výstup transakcie. Keď je importovaná transakcia s viac ako dvoma unikátnymi adresami na vstupoch, tak je zadávaný pokyn komponenta `Address_Clusterizer` pre zaradenie adries na vstupoch do jednej Bitcoin peňaženky. Činnosť komponenty `Address_Clusterizer` je popísaná v ďalšej časti textu. Po importovaní všetkých transakcií v rámci bloku sú uložené do kolekcie `blocks` informácie popisujúce blok v `blockchaine`.

3. V prípade, že existuje ďalší blok za posledným importovaným blokom `blockchainu` a nie je dosiahnutý limit na maximálny počet stiahnutých blokov za jeden beh algoritmu, tak je realizovaný import obsahu ďalšieho bloku podľa druhého kroku.

3.7 Implementácia bloku Address_Clusterizer

Podkapitola sa venuje popisu implementácie komponety `Clusterizer`, ktorej úlohou je zoskupovať adresy patriace do jednej Bitcoin peňaženky. Komponenta bude realizovať tzv. zhlukovú analýzu, ktorej výsledkom budú zhluky (skupiny) adries, ktoré sú pravdepodobne spravované jednou osobou. Takýto zoznam adries je súčasťou práve jednej Bitcoin peňaženky.

Text podkapitoly je členený do dvoch častí. Prvá polovica sa venuje popisu problému identifikácie adries patriacich do jednej Bitcoin peňaženky. Implementácia komponenty

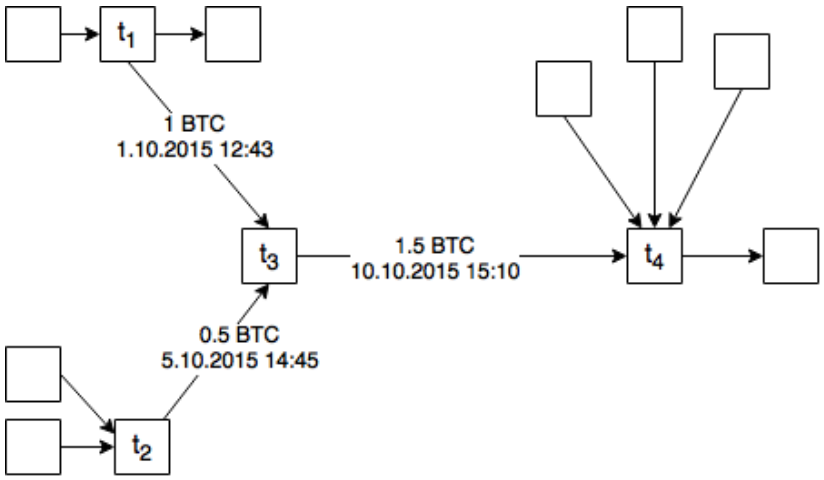
`Clusterizer` ie umiestnená v druhej časti podkapitoly.

3.7.1 Analýza problému

V predchádzajúcich častiach textu bolo objasnené, že užívateľ siete Bitcoin môže disponovať prakticky nekonečným množstvom Bitcoin adries, pomocou ktorých realizuje platobné transakcie. V prípade, že by užívateľ používal neustále nové adresy, tak by za pomoci takejto aktivity mohol skrývať svoju aktivitu na sieti Bitcoin. Na základe toho je jednou z kľúčových funkcií nástroja forenznej analýzy získanie zoznamu adries, ktoré sú spravované jednou osobou.

Spôsob riešenia problému identifikácie Bitcoin peňaženiek je popísaný za pomoci prevodu grafu transakcií na graf užívateľov siete Bitcoin. Tok finančných prostriedkov v sieti Bitcoin môžme reprezentovať za pomoci acyklického orientovaného grafu (označovaný ako DAG) transakcií. Uzly grafu predstavujú transakcie a každá hrana medzi zdrojom a cieľom reprezentuje výstup predchádzajúcej transakcie, ktorý bol použitý ako vstup pre ďalšiu transakciu. Hrany sú ohodnotené množstvom presúvaných BTC a časovou známkou, kedy bol presun vykonaný.

Konštrukciu takéhoto grafu z obsahu `blockchainu` je možné realizovať priamo s použitím štandardných algoritmov. Obrázok 3.4 ilustruje graf transakcií vytvorený za pomoci uvedeného spôsobu.

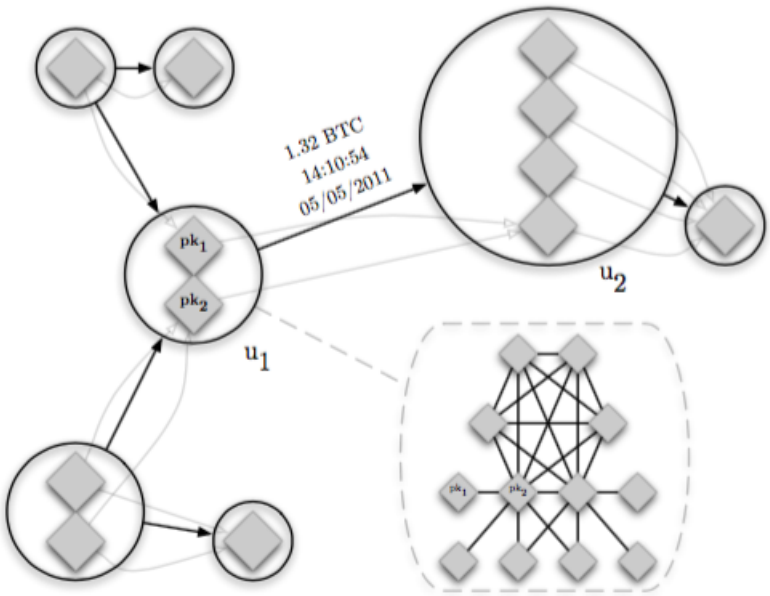


Obr. 3.4: Graf transakcií v sieti Bitcoin.

Ako je možné vidieť na obrázku 3.4 s každou pribúdajúcou transakciou veľkosť grafu transakcii narastá, a tak sa graf stáva čoraz viac neprehľadným. Graf transakcií je vhodný pre sledovanie toho, ako sa finančné prostriedky presúvali medzi jednotlivými adresami. Avšak z uvedeného grafu nie je možné určiť to, ako sa prostriedky presúvali medzi jednotlivými peňaženkami siete Bitcoin.

Graf transakcií je možné transformovať do takého tvaru, kde uzly reprezentujúce transakcie sú nahradené identifikátormi užívateľov siete Bitcoin a hrany značia presuny finančných prostriedkov medzi užívateľmi siete. Takto vzniknutý graf môžme označiť ako graf užívateľov siete Bitcoin.

Obrázok 3.5 znázorňuje štruktúru grafu užívateľov siete. Kosoštvorce reprezentujú adresy užívateľov siete. Kruhy v grafe znázorňujú užívateľov siete (Bitcoin peňaženky). Hrany medzi jednotlivými kruhmi reprezentujú tok finančných prostriedkov medzi užívateľmi siete.



Obr. 3.5: Graf užívateľov siete Bitcoin. Prevzaté z [15].

Počas prevodu grafu transakcií na graf užívateľov siete sú vstupy transakcie nahradené kruhom, ktorý reprezentuje užívateteľa siete Bitcoin, pretože pokyn pre realizáciu transakcie zadával užívateľ sám, a preto je možné predpokladať, že adresy na vstupoch transakcie sú spravované práve jednou osobou. Výstupy transakcie predstavuju hrany vo vznikajúcom grafe. Priradenie adries na výstupoch transakcie do Bitcoin peňaženiek je komplikované z toho dôvodu, že časť výstupov transakcie je adresovaných adresami, na ktoré sú zasielané zvyšky nepoužitých finančných prostriedkov v transakciách. Často sú tieto výstupy adresované pomocou tzv. `shadow adres`.

`Shadow adresy` sú užívateľom novo vygenerované adresy za účelom adresácie výstupov, na ktoré budu preposlané zvyšky nevyužitých vstupných prostriedkov transakcie. Najčastejšie sa stretávame s transakciami s práve dvoma výstupmi, kde jeden z nich adresuje iného užívateľa a druhý je adresovaný pomocou `shadow` adresy.

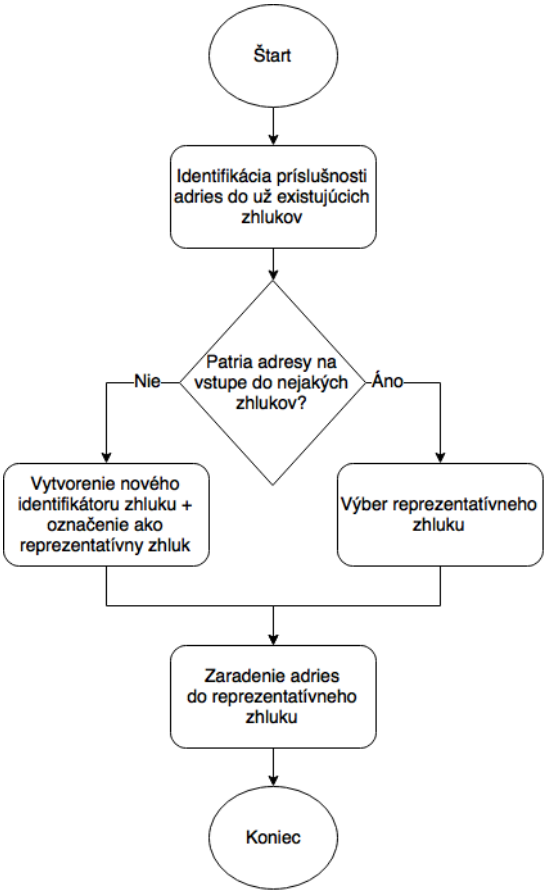
V prípade, že by sme mohli bezpečne určiť, ktoré adresy z výstupov transakcie predstavujú `shadow adresy`, tak by sme tieto adresy mohli zaradiť do rovnakého zhluku ako adresy, ktoré sú na vstupoch transakcie. Identifikovať `shadow adresy` by bolo možné tak, že by bolo kontrolované či adresy na jednotlivých výstupoch transakcie už boli v minulosti použité. V prípade, že by adresa na výstupe ešte nebola použitá, tak by sme ju mohli označiť ako `shadow` a zaradiť ju bezpečne do zhluku. Avšak s týmto princípom bojuje implementácia `blockchainu` tak, že transakcie v bloku nie sú zoradené podľa ich vzniku, a tak nie je možné týmto spôsobom identifikovať `shadow adresy`.

Za pomoci konštrukcie grafu užívateľov dokážeme identifikovať presuny finančných prostriedkov medzi jednotlivými užívateľmi siete Bitcoin. Graf je oproti grafu transakcii neporovaneteľne jednoduchší a poskytuje tak lepší pohľad nad tokmi prostriedkov v sieti. Tvorba (simulácia) výpočtu grafu užívateľov je kľúčová v procese vyhľadávania adries, ktoré sú spravované jednotlivými užívateľmi siete Bitcoin.

3.7.2 Implementácia

Na základe vykonanej analýzy som sa rozhodol, že komponenta `Address_Clusterizer` bude zaraďovať do jedného zhluku adresy, ktoré boli použité na vstupoch jednotlivých platobných transakcií v platobnej sieti. Môže sa stať, že adresy na vstupoch transakcie patria do už existujúcich zhlukov adries. V takomto prípade sú existujúce zhluky spojené do jedného, ktorý reprezentuje celú Bitcoin peňaženku. Adresy z výstupov transakcií nebudú zaraďované do zhlukov adries, pretože som nenašiel žiadne riešenie problému identifikácie `shadow` adries, ktoré by spoľahlivo fungovalo.

Implementácia komponenty je zapuzdrená v metóde `createCluster` triedy `ClusterModel`. Metóda `createCluster` preberá ako vstupný parameter pole Bitcoin adries, ktoré sú na vstupoch platobnej transakcie, a tak majú byť zaradené do jedného zhluku adries. Pokyny pre zaradenie adries do zhlukov (Bitcoin peňaženiek) zadáva komponente `Addresss_Clusterizer` komponenta `Blockchain_Impoter` počas importu nových transakcií z `blockchainu`, ktoré majú vstupy adresované pomocou minimálne dvoch unikátnych Bitcoin adries. Za pomoci toho sa zamedzí vzniku nadmernému množstvu zhlukov s práve jednou adresou, ktoré nie sú nejak zaujímavé. Zhluky adries sú počas importu nových dát `blockchainu` do databáze nástroja forenznej analýzy vytvárané inkrementálne.



Obr. 3.6: Štruktúra transakcií v sieti Bitcoin pre popis implementovaného algoritmu.

Vývojový diagram na obrázku 3.6 popisuje činnosť algoritmu implementovaného v metóde `createCluster`.

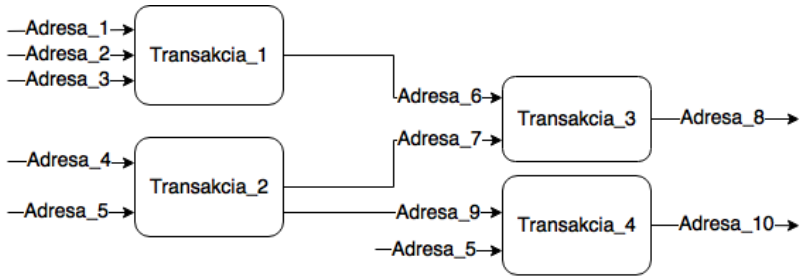
Činnosť implementovaného algoritmu môžme dekomponovať do troch krokov:

1. V prvom kroku je realizované mapovanie vstupných adries na už existujúce zhluky. Výsledkom tohoto kroku je pole s indexmi `clusters` - zoznam zhlukov, do ktorých patria vstupné adresy a `notInClusters` - zoznam adries, ktoré nepatria do žiadneho z existujúcich zhlukov.

2. Následne prebieha voľba zhluku, ktorý bude reprezentovať novo vzniknutý zhluk. V prípade, že žiadna z adries na vstupe nepatrí do už existujúceho zhluku, tak potom vznikne úplne nový zhluk adries. Inak je potrebné vybrať zo získaného zoznamu zhlukov reprezentatívny zhluk. Zhluk s najväčším počtom adries je vybraný ako reprezentatívny zhluk.

3. V tomto kroku sú adresy v už existujúcich zhlukov zaradené do reprezentatívneho zhluku. Adresy, ktoré ešte neboli zaradené do žiadneho zhluku sú zaradené do reprezentatívneho zhluku a zároveň je u týchto adries uložený stav na účte. Ukladanie stavu účtu umožňuje rýchle spočítanie celkového stavu finančných prostriedkov v rámci celej Bitcoin peňaženky, a tak nie je potreba počítať komplikované a časovo náročné agregácie nad miliónmi záznamov v MongoDB kolekcii `transactions`.

Obrázok 3.7 reprezentuje sadu štyroch Bitcoin transakcií, za pomoci ktorých bude ilustrovaná činnosť implementovaného algoritmu.



Obr. 3.7: Štruktúra transakcií v sieti Bitcoin pre popis implementovaného algoritmu.

Počas spracovávania transakcií uvedených na obrázku komponenta `Blokckchain_Importer` zadá komponente `Address_Clusterizer` štyri príkazy pre identifikáciu príslušnosti adries do Bitcoin peňaženiek.

1. Po spracovaní vstupov transakcie Transakcia_1 vznikne zhluk $A = \{Adresa\_1, Adresa\_2, Adresa\_3\}$.

2. Spracovaním vstupov transakcie Transakcia_2 vznikne zhluk $B = \{Adresa\_4, Adresa\_5\}$.

3. Po spracovaní vstupov transakcie Transakcia_3 vznikne zhluk $C = \{Adresa\_6, Adresa\_7\}$

4. Počas spracovávanie vstupov transakcie Transakcia_4 je zistené, že adresa Adresa_5 bude zaradená do zhluku B, a tak adresa Adresa_9 bude pridaná do zhluku B. Obsah zhluku B tak bude $\{Adresa\_4, Adresa\_5, Adresa\_9\}$

3.8 Implementácia bloku Tagger

V nasledujúcej podkapitole sa budeme venovať popisu implementácie komponenty `Tagger`. Úlohou komponenty je priraďovanie verejne dostupných identifikátorov Bitcoin adresám, za účelom deanonymizácie aktivity na sieti Bitcoin.

Text podkapitoly je rozdelený do dvoch častí. Prvá časť sa venuje analýze toho akým spôsobom je možné získavať informácie o identite vlastníkov Bitcoin adries za pomoci verejných zdrojov. Druhá časť sa venuje popisu samotnej implementácie komponenty `Tagger`.

3.8.1 Analýza problému

V sieti Bitcoin sa nenachádza žiaden centrálny register (registre), vďaka ktorému je možné dohľadať identitu osôb pracujúcich so sieťou Bitcoin. Ponúka sa nám možnosť vytvoriť vlastný adresár, ktorý by obsahoval asociácie adries užívateľov siete Bitcoin a informácií získaných z externých zdrojov.

Mnoho organizácii a služieb, ktoré využívajú služby siete Bitcoin (napríklad online obchody, Bitcoin zmenárne, diskusné fóra, internetové kasína atď.) obsahujú informácie o identitách (pseudoidentitách) užívateľov, ktorý s takýmito systémami pracovali. Identifikátory osôb môžu byť v rôznej podobe: vo forme emailu, prezývky na diskusnom fóre, kompletná adresa zákazníka, číslo bankového účtu, IP adresa zariadenia, z ktorej bola realizovaná transakcia a iné. Pokiaľ sú informácie o identitách voľne dostupné na webových stránkach. tak sa ponúka možnosť implementácie automatizovaných skriptov. Úlohou týchto skriptov je vytváranie asociácií medzi existujúcimi adresami siete a zistenými identifikátormi. V prípade, že sa jedná o prezývku na diskusnom fóre alebo o inú formu nekonkrétnej identifikácie vlastníka adresy, tak bezpečnostná zložka môže právnou cestou získať detailnejšie informácie o osobe, ktorá vystupuje pod danou pseudoidentitou.

Pokiaľ by nástroj forenznej analýzy dokázal získavať informácie o identitách (pseudoidentitách) osôb pracujúcich s platobnou sieťou Bitcoin, bolo by možné aktivitu na sieti aspoň čiastočne deanonymizovať. Následne by osoba realizujúca forenznú analýzu mohla priradiť identitu jednotlivým účastníkom platobných transakcií a efektívne tak sledovať toky finančných prostriedkov medzi subjektami tvoriacich platobnú sieť.

3.8.2 Implementácia

Po analýze som sa rozhodol, že komponenta `Tagger` bude implementovaná ako automatická úloha, ktorá bude sťahovať informácie o identitách/pseudoidentitách Bitcoin adres z verejne dostupných zdrojov (webových stránok) v pravidelnom časovom intervale. Informácie o identitách/pseudoidentitách sú sťahované z webovej stránky `https://blockchain.info/tags`, ktorá agreguje takéto informácie z viacerých zdrojov (užívateľmi zadané, získané za pomoci parsovania webových stránok diskusných fór atď.).

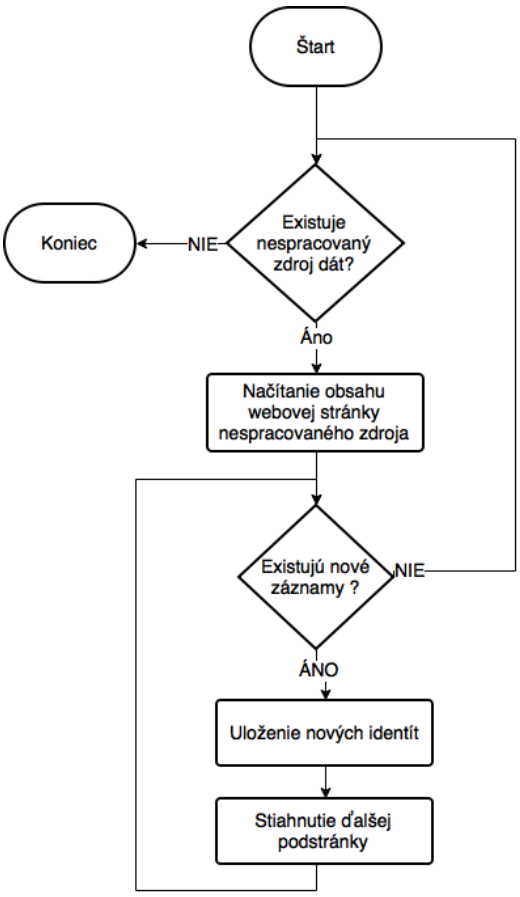
Na webovej stránke sú informácie o identitách/pseudoidentitách Bitcoin adries rozdelené do celkovo štyroch skupín a to *Submitted links* - užívateľmi zadávané informácie o adresách, *Signed messages* - užívatelia, ktorý potvrdili svoju identitu za pomoci digitálneho podpisu, *BitcoinTalk Profiles* - informácie získané z profilov na diskusnom fóre Bitcoin Forum[14] a *Bitcoin-OTC Profiles* - dáta získané spracovaním obsahu na webovej stránke Bitcoin trhoviska #bitcoin-otc[15]. Ku všetkým adresám na podstránkach je uvedený popisok adresy

[14] Viac informácií na: https://bitcointalk.org
[15] Viac informácií na: https://bitcoin-otc.com

(prezývka na diskusnom fóre, názov webu atď.). K niektorým adresám je tu navyše uvedený webový odkaz, na ktorom bola spomenutá daná adresa.

Implementácia samotnej komponenty `Tagger` sa sústreďuje do triedy `DownloadTags`. Táto trieda implementuje automatickú úlohu, ktorá je spustená každú hodinu (časový interval je možné zmeniť v triede `Kernel` z menového priestoru `App\Console`). Vývojový diagram na obrázku 3.8 popisuje algoritmus implementovaný v triede `DownloadTags`, ktorého úlohou je sťahovanie informácií o identitách/pseudoidentitách užívateľov siete Bitcoin z verejne dostupných zdrojov. Jednotlivé identity/pseudoidentity sú ukladané do MongoDB kolekcie `addresses`, ktorá bola popísaná vyššie. Počas ukladania informácií o identite užívateľa siete je u užívateľa v MongoDB kolekcii nastavená položka `balance` na aktuálnu hodnotu so stavom účtu. Za pomoci tohoto je zrýchlene zobrazovanie dát na profilových stránkach adries a peňaženiek zároveň sa tak vyhneme realizácií agregačných funkcií nad veľkými kolekciami dát. V prípade importu nových transakcií s adresami v kolekcii adries je hodnota položky `balance` navyšovaná/znižovaná podľa toho či sa jednalo o adresu na vstupe alebo výstupe transakcie.

V budúcnosti je možné rozšíriť implementáciu o podporu získavania identit/pseudoidentit z ďalších zdrojov dát.



Obr. 3.8: Vývojový diagram algoritmu pre získavanie informácií o identitách/pseudoidentitách užívateľov siete Bitcoin.

3.9 Implementácia bloku Visualization

Podkapitola sa zameriava na popis implementácie bloku `Visualization`, ktorý zodpovedá za zobrazovanie získaných dát nástrojom forenznej analýzy pomocou webovej aplikácie. Pomocou implementovanej aplikácie je možné zobrazovať dáta vzťahujúce sa k blokom, transakciam a Bitcoin adresám. Komponenta zároveň umožňuje vyhľadávanie v obsahu `blockchainu`.

Komponenta `Visualization` bola implementovaná pomocou niekoľkých radičov a zobrazovačov. Radiče odchytávajú požiadavky užívateľov systému, následne na základe nich získavajú a spracovávajú dáta z požadovaných dátových modelov. Získané a spracované dáta sú následné zobrazované pomocou zobrazovačov.

V rámci implementácie bolo vytvorených niekoľko radičov a zobrazovačov. Medzi implementované radiče patria nasledovné:

Controller
: Abstraktný radič, od ktorého dedia všetky implementované radiče. V rámci tohoto radiča bola implementovaná spoločná funkcionalita pre všetky radiče.

SearchController
: Radič, ktorý odchytáva a následne obsluhuje požiadavky na vyhľadávanie v obsahu `blockchainu`. `blockchainu`.

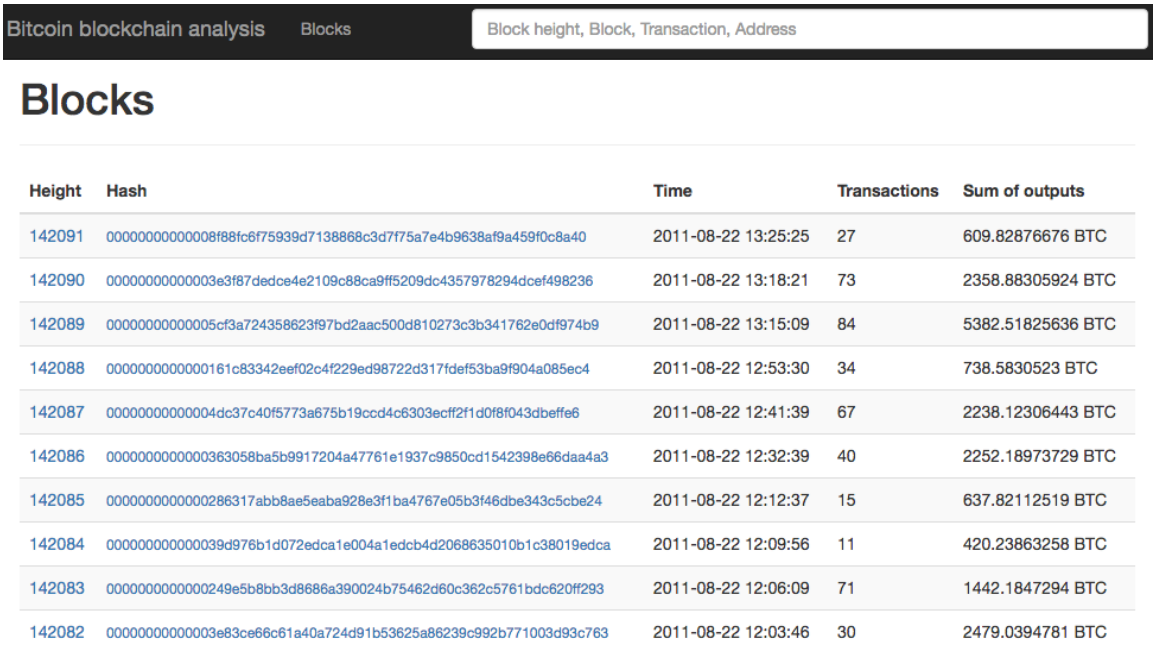
BlockController
: Je radič, ktorého úlohou je obsluha požiadavkov vzťahujúcich sa ku zobrazovaniu informácií o obsahu v `blockchaine`. Patrí sem zobrazovanie výpisu blokov obsiahnutých v `blockchaine` a zobrazovanie detailných stránok popisujúcich obsah jednotlivých blokov `blockchainu`.

TransactionController
: Radič, ktorého úlohou je obsluha požiadavkov vzťahujúcich sa ku transakciám. Jedná sa o zobrazovanie štruktúry transakcie, profilovej stránky transakcie a vizualizáciu transakcií.

AddressController
: Je radič, ktorého úlohou je zobrazovanie informácií vzťahujúcich sa k Bitcoin adresám. Za pomoci tohoto radiča sú zobrazované profilové stránky Bitcoin adries peňaženiek.

Sekcia *Blocks*

Na obrázku 3.9 je zobrazená hlavná obrazovka nástroja forenznej analýzy v sieti Bitcoin. Hlavná stránka webovej aplikácie zobrazuje základne informácie o importovaných blokoch vo forme tabuľky. Vrchná časť stránky je tvorená hlavným menu, ktorá obsahuje odkaz na jedinú sekciu *Blocks* a textové pole pre zadávanie požiadavkov pre vyhľadávanie v blockchaine.

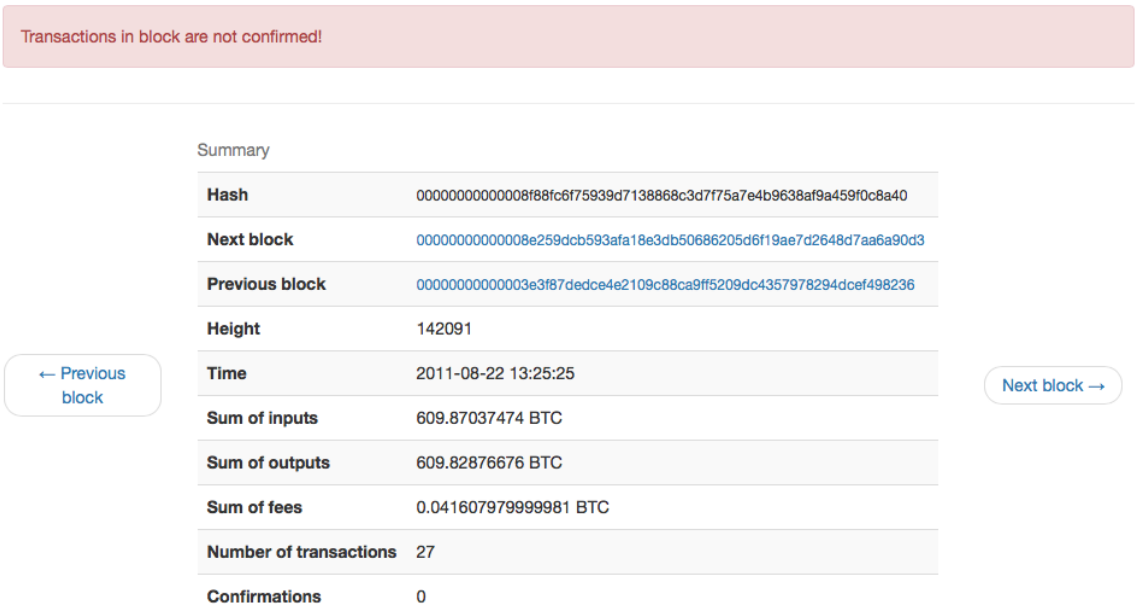


Obr. 3.9: Hlavná obrazovka nástroja forenznej analýzy.

Sekcia *Blocks* obsahuje tabuľkový výpis informácií o blokoch uložených v databáze nástroja forenznej analýzy. Sú tu obsiahnuté základné informácie o jednotlivých blokoch. Po kliknutí na položku *hash* alebo *height* vybraného bloku je užívateľ presmerovaný na stránku popisujúcu obsah bloku v `blockchaine`.

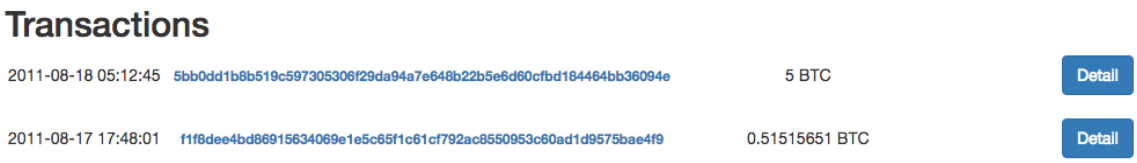
Profil bloku `blockchainu`

Podoba vrchnej časti profilovej stránky bloku je uvedená na obrázku ??. Úplne na vrchu stránky sa nachádza informácia o stave overenia transakcií obsiahnutých v bloku. V strede sa nachádza tabuľka obsahujúca informácie popisujúce obsah bloku, sumárne charakteristiky atď. Po stranách sú umiestnené ovládacie prvky, pomocou ktorých je možné pohodlne prechádzať medzi profilovými stránkami jednotlivých blokov.



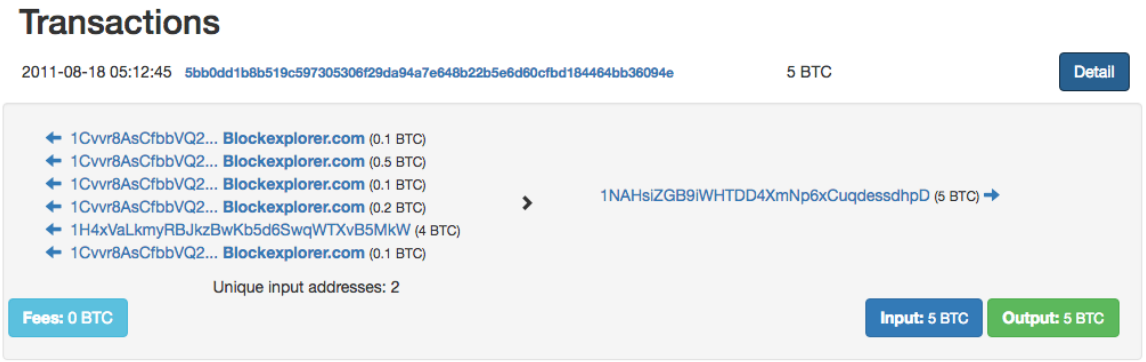
Obr. 3.10: Vrchná časť profilovej stránky bloku.

Ďalšia časť profilovej stránky je tvorená zoznamom transakcií, ktoré sú obsiahnuté v bloku. Znázornené je to na snímku 3.11. Zoznam obsahuje základné informácie vzťahujúce sa ku transakcii. Po kliknutí na identifikátor transakcie sa užívateľ môže prekliknuť na profilovú stránku platobnej transakcie.



Obr. 3.11: Profilová stránka bloku.

Po kliknutí na tlačítko „Detail" je zobrazená štruktúra platobnej transakcie tak, ako je možné vidieť na snímku 3.12. Informácie o štruktúre transakcie sú načítavané pomocou AJAXu čím je dosiahnuté rychlé vykresľovanie zoznamu transakcií na profilových stránkach blokov, pretože sú tú vykresľované len základné informácie.



Obr. 3.12: Profilová stránka bloku.

V ľavej časti je umiestnený popis vstupov transakcie. Po kliknutí na ikonu šípky, ktorá je umietnenená pri vstupoch, je užívateľ presmerovaný na profilovú stránku transakcie, ktorej výstup bol použitý na realizáciu transakcie. Pravá časť detailu štruktúry transakcie obsahuje popis výstupov transkacie. V prípade, že výstup transakcie bol použitý na realizáciu ďalšej transakcie, tak je pri ňom zobrazená ikona šípky. Po kliknutí na túto ikonu je užívateľ presmerovaný na profilovú stránku transakcie v rámci, ktorej boli použité finančné prostriedky z výstupu transakcie.

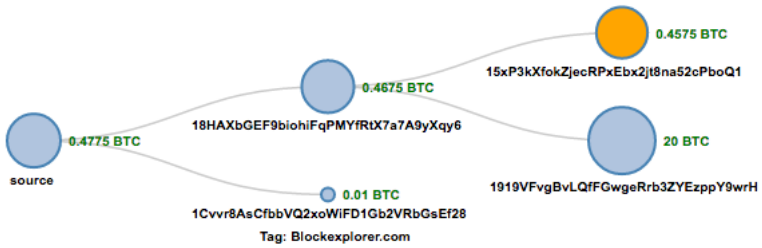
Všetky vstupy a výstupy transakcie sú popísané pomocou Bitcoin adries a množstvom finančných prostriedkov, ktoré sa k nim vzťahovali. V prípade, že u niektorej z adries na vstupe alebo výstupe je známa identita, tak je u daného vstupu/výstupu zobrazená informácia o identite vedľa adresy. V prípade, že užívateľ klikne na Bitcoin adresu, tak je presmerovaný na jej profilovú stránku.

V spodnej časti transakcie sú zobrazené informácie o výške poplatku za realizáciu transakcie, množstve prostriedkov vstupujúcich do transakcie a celkovom množstve prostriedkov na výstupoch transakcie.

Profilová stránka transakcie

Profilová stránka transakcie obsahuje informácie o transakcie vo forme tabuľky a štruktúre transakcie v rovnakej podobe ako po kliknutí na tlačítko „Detail" pri transakcii v prípade výpisu transakcií na profilovej stránke bloku `blockchainu`.

Navyše tu nachádzame odkaz s popiskom „Transaction graph". Po kliknutí na tento odkaz je zobrazená grafická vizualizácia transakcie tak, ako je vidieť na snímku3.13.



Obr. 3.13: Grafická vizualizácia transakcie.

V rámci vizualizácie transakcie je užívateľovi zobrazená legenda popisujúca graf. Jednotlivé uzly grafu predstavujú výstupy transakcií. Uzly sú popísané pomocou Bitcoin adries,

identitou osôb vzťahujúcich sa k jednotlivým adresám v prípade, že sú známe identity a množstvom finančných prostriedkov, ktoré boli smerované na jednotlivé výstupy. Výstupy transakcií, ktoré boli použité na realizáciu ďalších platobných transakcií sú v grafe označené oranžovou farbou a je možné na ne kliknúť, a tak zobraziť výstupy ďalších transakcií, ktoré z nich vznikli.

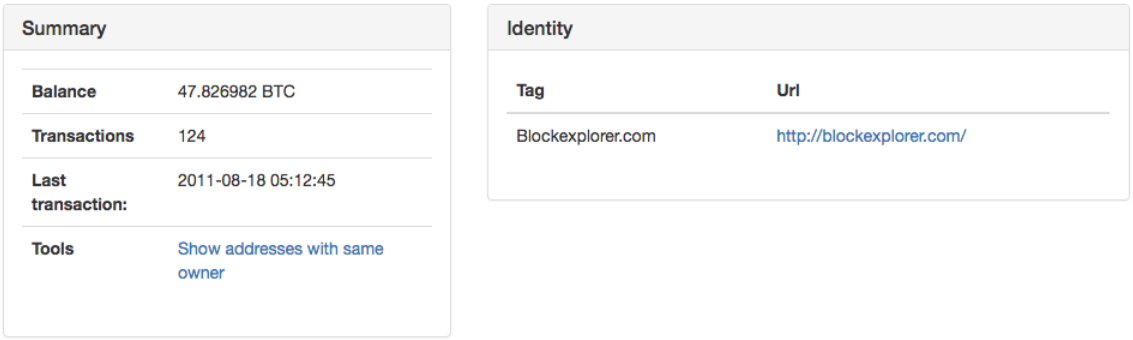
Veľkosť uzlov v grafe je závislá na množstve finančných prostriedkov, ktoré boli smerované na jednotlivé výstupy transakcie. Na jednotlivé Bitcoin adresy v grafe je možné kliknúť a následne tak zobraziť profilovú stránku danej adresy. V prípade, že je pri uzle grafu zobrazená identita, tak je možné kliknúť na jej popisok a byť tak presmerovaný na webovú stránku, kde informácia o identite adresy bola spomenutá.

Pomocou takéhoto grafu je možné sledovať toky finančných prostriedkov v rámci siete Bitcoin.

Profilová stránka adresy

Profilová stránka adresy slúži pre zobrazenie informácií vzťahujúcich sa ku adrese. Vrchná časť profilovej stránky adresy je zobrazená na snímku 3.14. V panele „Summary" sú zobrazované informácie o množstve finančných prostriedkov na adrese, informácie vzťahujúce sa ku transakciám (počet transakcií a čas poslednej transakcie s danou adresou) a sekcia nástrojov. Panel „Identity"je určený pre zobrazovanie informácií o identite užívateľa v prípade, že je známa.

Na profilovej stránke adresy nachádzame odkaz s popiskom „*Show addresses with same owner*". Po kliknutí na uvedený odkaz sa zobrazí profilová stránka peňaženky, do ktorej adresa patrí. Na profilovej stránke adresy je ďalej zobrazovaný výpis transakcií vzťahujúcich sa ku zadanej adrese v rovnakej podobe ako v prípade profilovej stránky bloku.

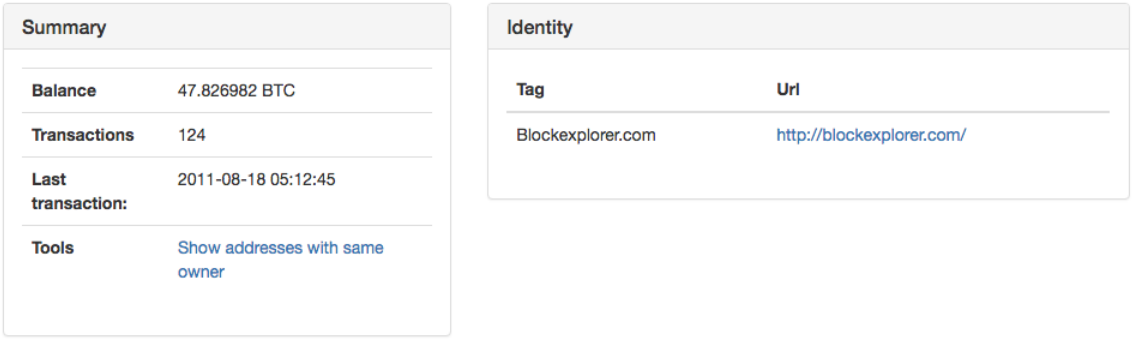


Obr. 3.14: Hlavná časť profilu Bitcoin transakcie.

Profilová stránka peňaženky

Profilová stránka slúži pre zobrazenie informácií vzťahujúcich sa ku Bitcoin peňaženke. Nachádzame tu informácie o celkovom množstve neminutých finančných prostriedkov v rámci Bitcoin peňaženky a identitách adries v Bitcoin peňaženke tak, ako je možné vidieť na snímkoch 3.15 a 3.16.

Obr. 3.15: Sumárne charakteristiky Bitcoin peňaženky.



Obr. 3.16: Informácie o identitách v rámci Bitcoin peňaženky.

Vyhľadávanie v `blockchaine`

Aplikácia umožňuje vyhľadávať v obsahu `blockchainu` podľa výšky bloku, hashu bloku, hashu transakcie alebo podľa adresy užívateľa. V prípade, že bol nájdený požadovaný záznam, tak je užívateľ presmerovaný na profilovú stránku hľadaného záznamu, inak je na hlavnej obrazovke aplikácie zobrazená informácia o neúspechu vyhľadávania.

Odchytávanie požiadavkov na vyhľadávanie je implementované v metóde `search` radiča `SearchController`. Keďže databáza `blockchainu` je rozsiahla, tak bolo potrebné vymyslieť vhodný spôsob vyhľadávanie dát. Na začiatku metódy je získaná vstupná hodnota podľa, ktorej sa má vyhľadávať. Na základe hodnoty vstupu je rozhodované, ako sa bude realizovať vyhľadávanie s dôrazom na to, aby sa prehľadávala čo najmenšia databázová kolekcia.

- Keď parameter je v podobe prirodzeného čísla, tak sa vyhľadáva v zozname blokov podľa výšky bloku. V prípade úspešného vyhľadávania je užívateľ presmerovaný na profilovú stránku bloku so zadanou výškou bloku.
- V prípade, že parameter pre vyhľadávanie obsahuje 64 znakov, tak sa vyhľadáva najskôr v zozname blokov následne v prípade neúspechu v zozname transakcií. Keď sa záznam podarilo nájsť, tak je užívateľ presmerovaný na profilovú stránku bloku alebo transakcie.
- Keď dĺžka vstupného parametru je 26-35 znakov (dĺžka Bitcoin adresy) a prvý znak je jeden z $\{1, 2, 3\}$ tak sa vyhľadáva podľa Bitcoin adresy. V prípade úspechu je užívateľ presmerovaný na profilovú stránku Bitcoin adresy.

Ďalej je na tejto stránke umiestnený zoznam všetkých adries, ktoré boli identifikované, že sú spravované pomocou jednej Bitcoin peňaženky. Adresy sú usporiadané podľa výšky zostatkov finančných prostriedkov na jednotlivých adresách.

3.10 Testovanie implementovaného nástroja

Nasledujúca kapitola je venovaná testovaniu implementovaného nástroja forenznej analýzy v sieti Bitcoin.

Funkčnosť implementovaného nástroja bola testovaná pravidelne počas celého vývoja. Testovanie funkčnej stránky aplikácie prebiehalo vždy po implementácii novej časti aplikácie.

Finálne otestovanie aplikácie bolo realizované v dvoch krokoch.

V prvom kroku boli porovnávané získane informácie o transakciách pomocou implementovanej aplikácie a referenčného nástroja Bitcoin Block Explorer. Toto testovanie zahrňovalo porovnávanie informácií o štruktúre transakcií (vstupy, výstupy, množstvo prenášaných finančných prostriedkov). Zároveň v rámci tohoto kroku prebiehalo testovanie grafickej vizualizácie platobných transakcií, pretože je podobne realizované v rámci nástroja Bitcoin Block Explorer. Pretože databáza identít spravovaná nástrojom Bitcoin Block Explorer predstavuje zdroj informácií o identitách užívateľov siete Bitcoin, tak bolo potrebné otestovať správne nahrávanie týchto informácií do databáze implementovaného nástroja.

V druhom kroku bola testovaná identifikácia adries patriacich do jednej Bitcoin peňaženky.

3.10.1 Porovnávanie získaných dát s referenčným nástrojom

Pre otestovanie správnosti aplikácie v prípade zobrazovania informácií vzťahujúcich sa ku štruktúre transakcií v `blockchaine` boli vybrané dve transakcie.

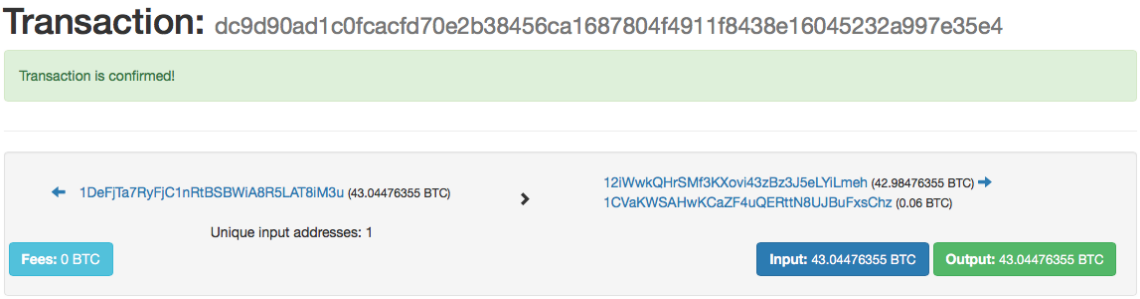
Prvá transkacia reprezentovala transakciu s neznámou identitou účastníkov. V tomto prípade sa overovalo získavanie a následne zobrazovanie štandardných informácií o transakciách.

U druhej transakcie bola známa identita jedného z účastníkov siete. Pomocou takejto transakcie bolo testované správne získavanie a následne zobrazovanie informácií o identite užívateľov siete podieľajúcich sa na realizácií platobných transakcií.

Transakcia s neznámou identitou účastníkov

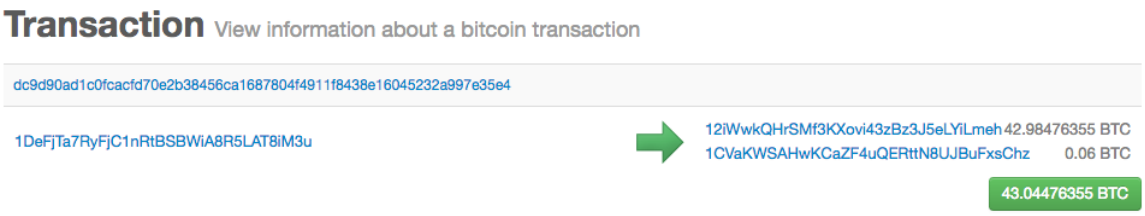
Ako prvá bola vybraná transakcia bez získaných informácií o identite užívateľov, ktorý sa podieľali na realizácií transakcie. Transakcia bola náhodne vybraná v rámci databáze implementovaného nástroja, tak aby splňovala uvedenú podmienku.

Implementovaný nástroj forenznej analýzy interpretoval informáciu o štruktúre vybranej transakcie, tak ako je uvedené na snímku 3.17. Na snímku je vidieť, že nástroj zobrazil zoznamy oadresovaných vstupov a výstupov transakcie. Na žiadnom zo vstupov/výstupov nie je uvedená informácia o identite užívateľa. Na jednotlivých vstupoch a výstupoch sú zobrazené informácie o množstve finančných prostriedkov. Ďalej nástroj zobrazil ovládacie prvky, pomocou ktorých je možný prechod na profilové stránky nadväzujúcich transakcií. Informácia o štruktúre transakcie je doplnená o sumárne charakteristiky transakcie: súčet vstupov, súčet výstupov a veľkosť poplatku.



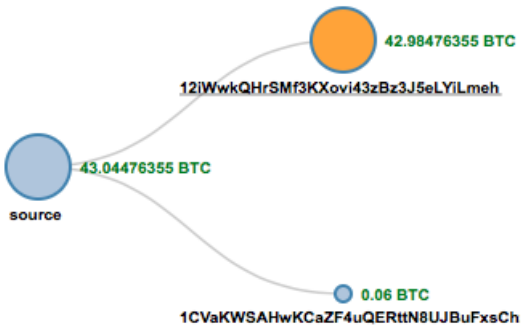
Obr. 3.17: Implementovaný nástroj - štruktúra transakcie s neznámou identitou účastníkov transakcie.

Nástroj Bitcoin Block Explorer zobrazil informácie o štruktúre uvedenej transakcie, tak ako je uvedené na obrázku 3.18. Z obrázku je viditeľné, že štruktúra základných informácií o transakcii je rovnaká (vstupy, výstupy), avšak nie sú tu obsiahnuté detailnejšie informácie (množstvo prostriedkov na vstupoch, kompletné sumárne charakteristiky transakcie, navigačné prvky pre prechod na profilové stránky nadväzujúcich transakcií).



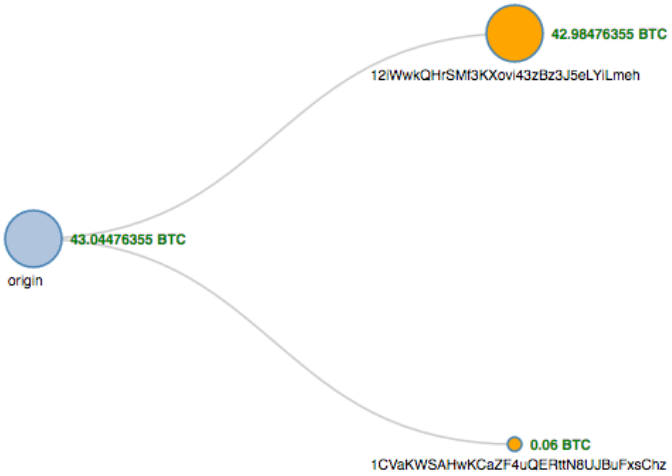
Obr. 3.18: Nástroj Bitcoin Block Explorer - štruktúra transakcie s neznámou identitou účastníkov transakcie.

Následne prebiehalo overovanie funkčnosti grafickej vizualizácie platobnej transakcie. Na obrázku 3.19 je graficky vizualizovaná platobná transakcia pomocou implementovaného nástroja. Keďže transakcia ma práve dva výstupy, tak k uzlu označovanému ako source (počiatok) sú pripojené ďalšie dva uzly reprezentujúce jednotlivé výstupy danej transakcie. Oranžový uzol značí výstup, ktorý bol použitý pre realizáciu ďalšej transakcií. Na tento uzol je možné kliknúť a tým zobraziť ako sa presúvali finančné prostriedky ďalej v sieti. Jednotlivé uzly sú označené adresami a množstvom finančných prostriedkov, ktoré sa na ne presúvali. Na adresy je možné kliknúť a zobraziť ich profily.



Obr. 3.19: Implementovaný nástroj - grafická vizualizácia transakcie.

Nástroj Bitcoin Block Explorer graficky vizualizoval transakciu, tak ako je uvedené na obrázku 3.20. Je možné vidieť, že informácie sú prakticky rovnaké. V nástroji Bitcoin Block Explorer akurát nie je možné prejsť na profily jednotlivých adries priamo z grafu.

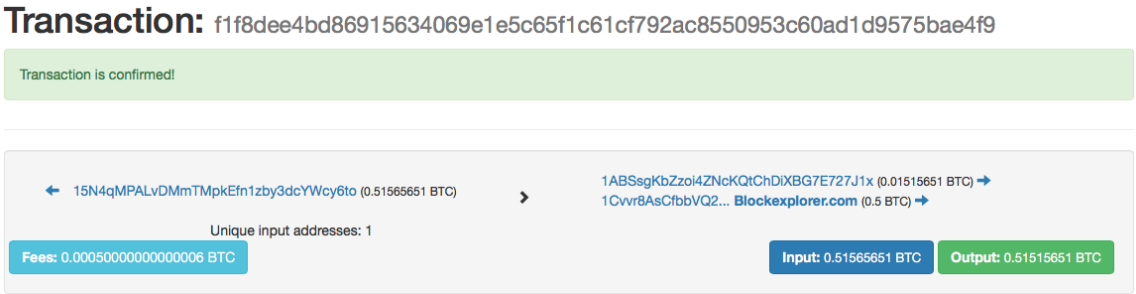


Obr. 3.20: Bitcoin Block Explorer - grafická vizualizácia transakcie.

Transakcia so známou identitou účastníka

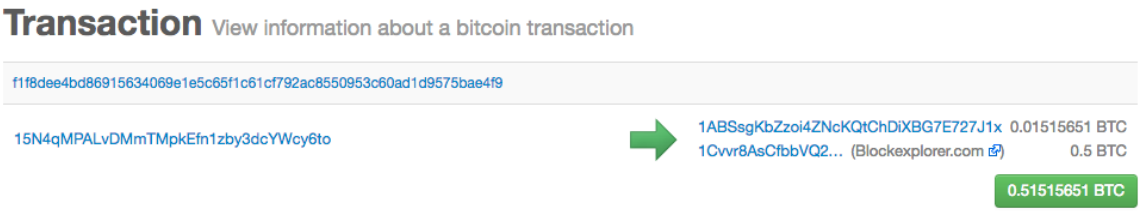
V nasledujúcej časti testovania bola vybraná transakcia, u ktorej ma byť zobrazovaná informácia o identite jedného z užívateľov. V rámci tejto časti je testované zobrazovanie informácie o identite adresy.

Implementovaný nástroj zobrazil informácie o štruktúre transakcie, tak ako je uvedené na obrázku 3.21. Nástroj na rozdiel od predchádzajúcej transakcie zobrazil u jedného z výstupov informáciu o identite osoby vzťahujúcej sa k danej adrese.



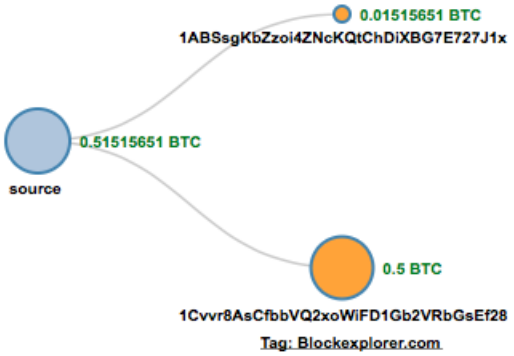
Obr. 3.21: Implementovaný nástroj - štruktúra transakcie so známou identitou účastníka.

Bitcoin Block Explorer zobrazil informáciu o štruktúre transakcie podobne ako u transakcie bez známej identity účastníkov. Akurát u jedného z výstupov transakcie nástroj zobrazil informáciu o identite vzťahujúcej sa k adrese na danom výstupe, ako je možné vidieť na obrázku 3.22



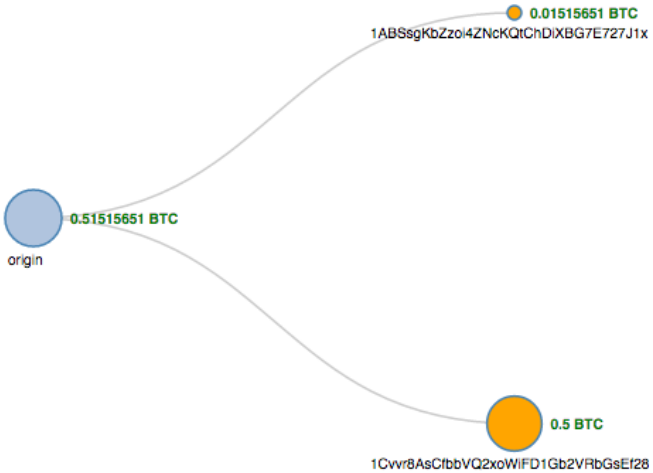
Obr. 3.22: Bitcoin Block Explorer - štruktúra transakcie so známou identitou účastníka.

Následne prebiehalo overovanie funkčnosti grafickej vizualizácie platobnej transakcie. Implementovaný nástroj vizualizoval graf transakcie podobne ako v prípade bez známej identity účastníkov. Akurát u uzlov označených adresou so známou identitou bola zobrazená informácia o ich identite, tak ako je možné vidieť na obrázku 3.23.



Obr. 3.23: Implementovaný nástroj - grafická vizualizácia transakcie so známym účastníkom siete.

Nástroj Bitcoin Block Explorer zobrazil graf transakcie, tak ako je možné vidieť na obrázku 3.24 bez informácií o identite o jedného z účastníkov.



Obr. 3.24: Bitcoin Block Explorer - grafická vizualizácia transakcie so známim účastníkom siete.

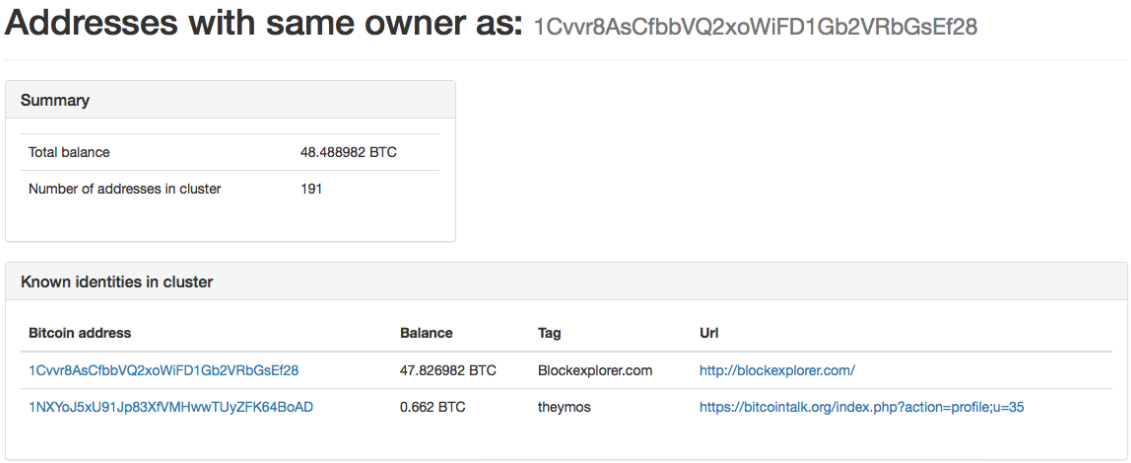
Na základe vykonaných testov pomocou dvojice transakcií som overil funkčnosť implementovanej aplikácie, keďže hlavné časti výsledkov oboch porovnávaných nástrojov boli rovnaké. Identity užívateľov vzťahujúcich sa k Bitcoin adresám sú podľa testov správne ukladané do databázy nástroja forenznej analýzy. Rozdiely medzi zobrazovanými informáciami o transakciách medzi nástrojmi boli akurát v prípade vylepšení, ktoré som implmentoval v rámci nového nástroja forenznej analýzy.

3.10.2 Testovanie idnetifikácie adries patriacich do jednej peňaženky

Testovanie idntifikácie príslušnosti adries do jednej Bitcoin peňaženky prebiehalo počas vývoja aplikácie tak, že bol proces identifikujúci adresy v Bitcoin peňaženkách spustený ručne nad množinou, ktorá obsahovala množiny adries (reprezentovali vstupy na jednotlivých transakciách). V prípade, že výsledok procesu boli množiny s očakávanou veľkosťou a prvkami ako zhluky, ktoré by mali byť identifikovaný, tak proces fungoval správne.

Počas finálneho otestovania aplikácie boli navyše realizované ručne testy. V priebehu týchto testov bola vybraná Bitcoin adresa, ktorá je zaradená do Bitcoin peňaženky. Následne sa zobrazila profilová stránka peňaženky, v ktorej je zahrnutá táto vybraná adresa. Potom bolo realizovaných niekoľko overení či adresy, ktoré sa vyskytovali spoločne na vstupoch transakcií patria do jednej Bitcoin peňaženky.

Príkladom môže byť Bitcoin adresa 1Cvvr8AsCfbbVQ2xoWiFD1Gb2VRbGsEf28. Pre skrátenie ďalších zápisov budeme túto adresu zapisovať v tvare 1Cvvr... Vrchná časť profilovej stránky tejto peňaženky je zobrazená na obrázku 3.25. Na obrázku je vidieť, že do identifikovanej peňaženky patrí 191 unikátnych adries a u dvoch adries z peňaženky je známa identita užívateľov.



Obr. 3.25: Vrchná časť profilovej stránky Bitcoin peňaženky pre adresu 1Cvvr....

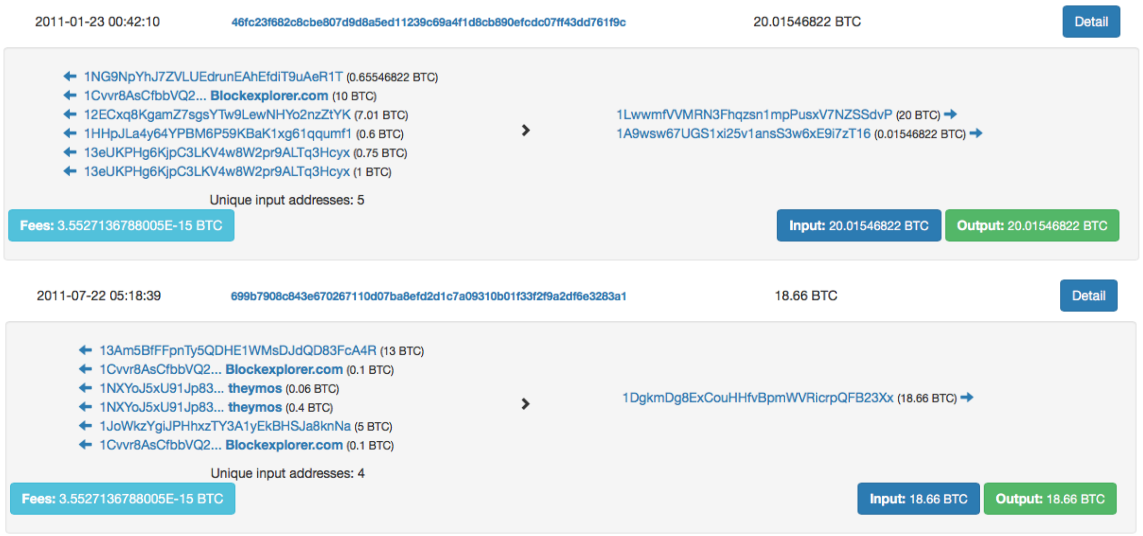
Na obrázku 3.26 je zobrazená časť výpisu adries patriacich do rovnakej peňaženky ako spomínaná adresa.

Addresses

Address in cluster	Balance
1Cvvr8AsCfbbVQ2xoWiFD1Gb2VRbGsEf28	47.826982 BTC
1NXYoJ5xU91Jp83XfVMHwwTUyZFK64BoAD	0.662 BTC
1A8UhGLH2MTncvUAF77CXnQs3tuKNpMePQ	0 BTC
1HHpJLa4y64YPBM6P59KBaK1xg61qqumf1	0 BTC
12ECxq8KgamZ7sgsYTw9LewNHYo2nzZtYK	0 BTC
15jD3qXdKW2Bxk7TXDoenCaLyovPGR6ZUo	0 BTC
19eeDPqJyiYcBo77zp5JvuQMa718CTNtLZ	0 BTC

Obr. 3.26: Časť zoznamu adries patriacich do rovnakej peňaženky ako adresa 1Cvvr....

Transakcie na obrázku 3.27 majú na vstupe spolu s adresou 1Cvvr... niekoľko ďalších adries. Všetky adresy na vstupoch uvedených transakcií by mali patriť do jednej Bitcoin peňaženky. U niekoľkých adresách použitých na vstupov transakcií (12ECxq..., 1HHpJL... a 1NXY...) je možné vidieť, že sú adresy uvedené v zozname adries patriacich do rovnakej peňaženky ako adresa 1Cvvr... Ďalšie adresy niesú priamo vidieť vo výpise v dôsledku stránkovania a zoraďovania výsledkov vo výstupnom zozname adries patriacich do jednej peňaženky.



Obr. 3.27: Príklad transakcií, na ktorých vstupoch je obsiahnutá adresa 1Cvvr....

Na základe vykonaných testov proces identifikujúci adresy patriace do jednotlivých Bitcoin peňaženiek funguje správne. V budúcnosti by bolo vhodné funkčnosť uvedeného procesu testovať za pomoci automatického testovacieho scenára, ktorý by celý proces výrazne uľahčil.

3.11 Zhrnutie

Kapitola sa zoberala návrhom, implementáciou a testovaním nástroja pre podporu realizácie forenznej činnosti v sieti Bitcoin. V prvotnej fázy bola vykonaná analýza aktuálneho stavu v oblasti nástrojov forenznej analýzy v sieti Bitcoin, kde som sa zameral predovšetkým na porovnanie rôznych existujúcich nástrojov. Následne som špecifikoval požiadavky kladené na nástroje pre podporu forenznej činnosti v sieti Bitcoin a na základe toho vytvoril diagram prípadov použitia. Za pomoci tohoto diagramu som navrhol a implementoval systém pre podporu forenznej analýzy.

Systém bol navrhnutý ako sada šiestich komponent označovaných ako: `Bitcoin_Deamon`, `Database`, `Visualization`, `Blockchain_Importer`, `Address_Clusterizer` a `Tagger`. Blok `Database` predstavuje mongoDB databázu, s ktorou pracuje celý systém. Komponenta `Bitcoin_Deamon` reprezentuje oficiálneho Bitcoin klienta spusteného s podporou Bitcoin JSON-RPC API. Pomocou tohoto bloku je získavaný obsah `blockchainu`. Impport novo vzniknutých blokov v `blockchaine` do bloku `Database` je zabezpečený za pomoci komponenty `Blockchain_Importer`. Identifikácia zoznamov adres, ktoré tvoria jednotlivé Bitcoin peňaženky je realizovaná za pomoci komponenty `Address_Clusterizer`. Komponenta `Tagger` bola implementovaná za účelom deanonymizácie činnosti na sieti Bitcoin. Komponenta `Visualization`, ktorá je zodpovedná za zobrazovanie získaných dát je implementovaná vo forme webovej aplikácie.

V závere bol popísaný spôsob akým bol implementovaný nástroj testovaný. Implementovaný nástroj prešiel úspešne všetkými testami , a tak ho môžme považovať za funkčný.

Kapitola 4

Záver

V úvode práce je predstavená najpoužívanejšia kryptomena súčasnosti označovaná ako Bitcoin. Hlavnými dôvodmi nárastu popularity kryptomeny Bitcoin sú: absencia centrálnej autority, ktorá by riadila činnosť siete a anonymita užívateľov podieľajúcich sa na platobných transakciách. Aj na základe týchto vlastností je sieť Bitcoin vo veľkej miere využívaná pre platby na čiernych trhoch, pranie špinavých peňazí a iné účely, pri ktorých je porušovaný zákon, a tak narastajú požiadavky na implementáciu nástrojov forenznej analýzy v sieti Bitcoin.

V prvej kapitole boli popísané princípy použité v sieti Bitcoin: adresácia užívateľských staníc, peňaženky, realizácia transakcií a ich evidencia v `blockchaine` atď. Záver kapitoly sa venoval porovnaniu kryptomeny Bitcoin s ďalšími kryptomenami.

Jadro práce sa venovalo návrhu a implementácii vlastného nástroja forenznej analýzy v sieti Bitcoin. Bol tu spracovaný popis aktuálneho stavu v oblasti tvorby forenzných nástrojov. Porovnávalo sa niekoľko existujúcich nástrojov pričom boli vytknuté nedostatky a možné zlepšenia existujúcich nástrojov. Následne boli za účelom deanonymizácie siete diskutované možnosti ako realizovať forenznú analýzu. Vlastný forenzný nástroj je implementovaný vo forme webovej aplikácie, ktorá pozostáva z komponent `Blockchain_Importer` - import obsahu `blockchainu`, `Address_Clusterizer` - identifikácia adries patriacich do jednej Bitcoin peňaženky, `Tagger` - získavanie informácií o identite užívateľov siete a z komponenty `Visualization`, ktorá zodpovedá za zobrazovanie získaných dát. Implementovaná aplikácia bola v priebehu celého vývoja testovaná. Po implementácii celej aplikácie boli realizované testy kompletnej aplikácie. Finálne testovanie prebehlo bez problémov, a tak je možné mnou vytvorený nástroj považovať za funkčný.

Hlavným cieľom práce bolo predstaviť problematiku fungovania platobnej siete Bitcoin a následne implementovať vlastný nástroj pre podporu realizácie forenznej činnosti v danej sieti. Tieto cieľe boli v rámci práce splnené.

V budúcnosti by bolo ideálne aplikáciu rozšíriť o ďalšiu funkcionalitu. Medzi takúto funkcionalitu patrí: získavanie detailných informácií vzťahujúcich sa k adresám (napr informácií o type adresy: hazard, zmenáreň atď.), zobrazovanie vývoju zostatkov na adresách a peňaženkách klientov siete. Na základe toho, že nástroj pracuje s databázovým systémom MongoDB a aplikácia je implementovaná pomocou návrhového vzoru MVC, tak je možné rozšíriť aplikáciu o podporu pre ďalšie kryptomeny. Úpravy by boli realizované v rámci modelovej vrstvy, ktorá je nezávislá od zvyšku aplikácie. Pre zabezpečenie rýchlosti aplikácie v prípade podpory ďalších krpytomen by bolo vhodné dáta k jednotlivým kryptomenám ukladať do samostatných MongoDB kolekcií. Na základe toho by mala byť implementovaná medzivrstva rozhodujúca o tom, ktoré kolekcie sa vzťahujú ku danej kryptomene.

Literatúra

[1] Bitcoin: Bitcoin Developer Guide. https://bitcoin.org/en/developer-guide.

[2] Bitcoin Wiki: Base58Check encoding. https://en.bitcoin.it/wiki/Base58Check_encoding.

[3] Bitcoin Wiki: Block. https://en.bitcoin.it/wiki/Block.

[4] Bitcoin Wiki: Block hashing algorithm. https://en.bitcoin.it/wiki/Block_hashing_algorithm.

[5] Bitcoin Wiki: Double-spending. https://en.bitcoin.it/wiki/Double-spending.

[6] Bitcoin Wiki: Genesis block. https://en.bitcoin.it/wiki/Genesis_block.

[7] Bitcoin Wiki: Mining. https://en.bitcoin.it/wiki/Mining.

[8] Bitcoin Wiki: Pooled mining. https://en.bitcoin.it/wiki/Pooled_mining.

[9] Bitcoin Wiki: RIPEMD-160. https://en.bitcoin.it/wiki/RIPEMD-160.

[10] Bitcoin Wiki: Script. https://en.bitcoin.it/wiki/Script.

[11] Bitcoin Wiki: Technical background of version 1 Bitcoin addresses. https://en.bitcoin.it/wiki/Technical_background_of_version_1_Bitcoin_addresses.

[12] Bitcoin Wiki: Transaction. https://en.bitcoin.it/wiki/Transaction.

[13] CoinMarketCap: Crypto-Currency Market Capitalizations. http://coinmarketcap.com.

[14] Developer Guide: Bitcoin. https://bitcoin.org/en/developer-guide#block-chain.

[15] F.Reid, M. Harrigan: An Analysis of Anonymity in the Bitcoin System. http://arxiv.org/abs/1107.4524.

[16] Litecoin: Litecoin - Open source P2P digital currency. https://litecoin.org.

[17] Ripple: Technology. https://ripple.com/technology/.

[18] S. Nakamoto: Bitcoin: A Peer-to-Peer Electronic Cash System. https://bitcoin.org/bitcoin.pdf.

[19] Wikipedia: Dark web. https://en.wikipedia.org/wiki/Dark_web.

[20] Wikipedia: Elliptic Curve Digital Signature Algorithm.
https://en.wikipedia.org/wiki/Elliptic_Curve_Digital_Signature_Algorithm.

[21] Wikipedia: Merkle tree. https://en.wikipedia.org/wiki/Merkle_tree.

Prílohy

Zoznam príloh

Príloha A

Zoznam použitých skratiek

- `P2P` - Peer-to-peer
- `UTXO` - Unspent Transaction Output
- `TCP` - Transmission Control Protocol/Internet Protocol
- `PHP` - PHP: Hypertext Preprocessor
- `JSON` - JavaScript Object Notation
- `URL` - Uniform Resource Locator
- `MVC` - Model-View-Controller
- `DB` - databáza
- `API` - Application programming interface
- `JSON-RPC` - JSON-Remote procedure call protocol
- `XML` - Extensible Markup Language
- `AJAX` - Asynchronous JavaScript + XML

Príloha B

Obsah CD

- `text` - Zdrojové TeX súbory textovej časti práce.
- `src` - Zdrojové kódy implementovanej aplikácie.
- `predvadzacie_video.mov` - Predvádzacie video implementovanej aplikácie.

Príloha C

Schéma konfiguračného súboru implementovanej aplikácie

Následujúca príloha je tvorená popisom obsahu konfiguračného súboru prostredia implementovanej aplikácie. Súbor musí byť pomenovaný ako `.env` a musí byť umiestnený v koreňovom adresáry implementovanej aplikácie. Konfiguračný súbor nemôže byť súčasťou verzovacieho systému, pretože sa v ňom nachádzajú prístupové údaje a zároveň obsah súboru nie je verejne dostupný.

Konfiguračný súbor `.env` ma nasledujúcu podobu:

```
APP_ENV=local
APP_KEY=key
BASE_URL=http://bt.local
BITCOINDRPC_URL=http://localhost:8332
BITCOINDRPC_USERNAME=rpc_meno
BITCOINDRPC_PASSWORD=rpc_heslo
MONGO_DSN=mongodb://localhost:27017
```

Parameter `APP_ENV` určuje typ behového prostredia aplikácie nasledovne: `local` - vývojové prostredie (zobrazujú sa chybové hlášky PHP atď.) a `production` - aplikácia beží ako v prípade nasadenia na ostrý server. Pomocou parametru `APP_KEY` je definovaná hodnota kľúča pre realizáciu šifrovania. Parameter `APP_DEBUG` rozhoduje či sa budú zobrazovať detailné informácie o vzniknutých chybách v aplikácií. Parameter `BASE_URL` obsahuje základnú URL implementovanej aplikácie. Parameter `BITCOINDRPC_URL` obsahuje URL adresu JSON-RPC API Bitcoin klienta. Pomocou parametrov `BITCOINDRPC_USERNAME` a `BITCOINDRPC_PASSWORD` sú nastavované prihlasovacie údaje pre JSON-RPC API Bitcoin klienta. Parameter `MONGO_DSN` obsahuje reťazec popisujúci pripojenie do MongoDB databáze. Môže obsahovať navyše ešte prihlasovacie meno a heslo ku databáze MongoDB.