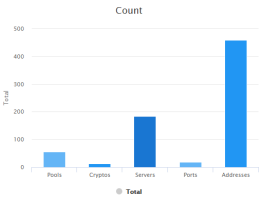
BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY

Excel@FIT 2018
http://excel.fit.vutbr.cz

Katalóg kryptomien

Jakub Kelečéni*



Abstrakt

Ťažba kryptomien je dnes čoraz viac populárna. Ľudia sa snažia zužitkovať dostupný výpočtový výkon s cieľom získania finančnej odmeny. Toto správanie však môže byť v rozpore s politikov internej siete niektorých organizácií (školy, banky, korporácie, . . .). Úlohou sieťových administrátorov je identifikovať zakázanú komunikáciu a následne ju odfiltrovať. Na odfiltrovanie takejto komunikácie je potrebná znalosť určitých dát (IP adresa, port). Pre tento účel bol vytvorený nástroj, katalóg, uchovávajúci údaje o kryptomenách, ťažobných pooloch a serveroch.

Cieľom tejto práce je predstavenie katalógu a popis návrhu (implementácie) jeho rozšírenia. Rozšírenie má doplniť možnosť sledovania a zaznamenávania dostupnosti ťažobných serverov. Výsledkom bude vyššia presnosť a spoľahlivosť informácií uchovaných v katalógu.

Pred samotnou implementáciou rozšírenia bol vypracovaný návrh, ktorému predchádzal teoretický rozbor problematiky. Ten pozostával z pochopenia základných princípov ťažby kryptomien a analýzy komunikácie generovanej počas ťažby. V článku sú spracované tri základné ťažobné protokoly (Getwork, Getblocktemplate, Stratum), ktoré sú v súčasnosti používané na výmenu informácií medzi ťažobným serverom a minerom.

Kľúčové slová: Cryptocurrency — Catalog — Mining — Getwork — Getblocktemplte — Stratum

Priložené materiály: GIT repozitár, webová aplikácia

*xkelec00@stud.fit.vutbr.cz, *Fakulta Informačných technológií, Vysoké Učení technické v Brně*

1. Úvod

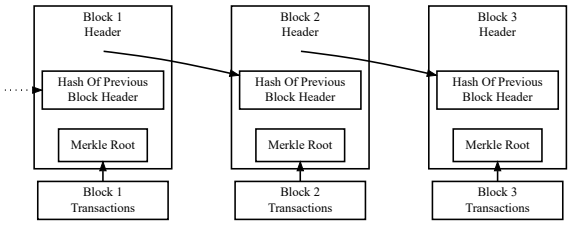
Článok je zameraný na predstavenie katalógu kryptomien a na popis návrhu (implementácie) jeho rozšírenia. Text práce je rozčlenený na samostatné logické celky, ktoré sú stručne špecifikované nižšie.

Korektnému návrhu rozšírenia predchádzala teoretická analýza princípov kryptomien. V článku je zahrnutá kapitola (2), ktorá stručne popisuje fundamentálne pojmy súvisiace s ťažbou kryptomien.

Ďalšia sekcia je venovaná trom základným ťažobným protokolom. Protokoly definujú formát a spôsob komunikácie medzi ťažobným serverom a minerom. Z hľadiska návrhu rozšírenia katalógu možno túto kapitolu (3) považovať za kľúčovú, čomu zodpovedá aj jej rozsah.

Jadro práce tvorí popis spomínaného katalógu (Kap. 4) a jeho rozšírenia (Kap. 5). Cieľom rozšírenia bolo pridanie časovej roviny k uloženým informáciám. To si môžeme predstaviť ako záznam údajov o dostupnosti ťažobného servera pre jednotlivé časové okamihy. Overenie dostupnosti bude vykonávané iniciovaním komunikácie na ťažobné protokoly popísané v Kap. 3. Výsledkom uvedených úprav bude zvýšenie spoľahlivosti a presnosti uložených dát.

Informácie uložené v katalógu vytvárajú databázu, ktorú je možné využiť pri identifikácií sieťovej prevádzky súvisiacej s ťažbou kryptomien. Z tohoto hľadiska je aktuálnosť a uložených dát kľúčová.



Obrázok 1. Schéma štruktúry Blockchainu (prevzaté z [3]).

2. Princípy kryptomien

Termín kryptomena možno chápať ako digitálne aktívum (majetok, prostriedok), prostredníctvom ktorého je možné vykonávať nejaké transakcie. Štandardne je postavená na určitom kryptografickom probléme, vďaka čomu poskytuje vysokú mieru bezpečnosti toku transakcií [1].

Dnes existuje relatívne veľké množstvo kryptomien, kde každá mena môže mať inú hodnotu. Medzi najpopulárnejšie môžeme zaradiť meny ako *Bitcoin*, *Ethereum*, *Monero*, *Ripple*, *DASH*, ...

Každá kryptomena implementuje tzv. *Blockchain*. Ide o dátovú štruktúru, ktorej úlohou je udržovať záznamy o všetkých vykonaných transakciách. Pre zjednodušenie je možné si ho predstaviť ako účtovnú knihu danej meny. Každý užívateľ siete kryptomeny disponuje úplnou, aktuálnou verziou tejto štruktúry. V prípade modifikácie *Blockchain-u* (pridanie transakcie) je táto zmena distribuovaná všetkým užívateľom. Dáta sú v štruktúre organizované do blokov, ktoré reprezentujú skupiny transakcií. Bloky sú usporiadané v lineárnom zozname, pričom každý blok uchováva informáciu o hodnote hash predchádzajúceho bloku. Samotná štruktúra transakcií vo vnútri bloku má podobu dátovej štruktúry *Merkle Tree* [2]. Vizualizácia *Blockchain-u* je viditeľná na Obr. 1.

Získavanie jednotiek meny je realizované tzv. ťažbou (*mining*). Je založená na rovnakom kryptografickom probléme ako príslušná kryptomena. Väčšinou je ťažba definovaná ako proces vytvárania dôkazu o vykonanej práci, ktorý sa vykonáva pri ukladaní bloku *B* do *Blockchain-u*. V sieti je generovaný hash `H` a miner sa snaží nájsť hodnotu *nonce* `N` takú, že platí $\texttt{H(N+blck\_hdr)} \leq \texttt{H}$. V momente splnenia tejto podmienky je daný blok vyťažený. Miner následne získava odmenu a vykoná sa aktualizácia *Blockchain-u*. Písmeno `H` v predchádzajúcom zápise, vyjadruje hašovaciu funkciu, ktorá závisí na použitej kryptomene. Názov `blck_hdr` zase označuje hlavičku bloku.

Prístup k ťažbe je možné rozdeliť na dva základné spôsoby: *solo mining* a *pooled mining*. Pri prístupe *solo mining* je hľadanie riešenia vykonávané iba jedným užívateľom a prípadná odmena za vyťaženie bloku náleží iba jemu. V súčasnosti sa tento prístup už takmer nepoužíva. Modernejší spôsob ťažby je *pooled mining*, kde zdroje minerov vytvárajú skupinu (*pool*). Hľadanie riešenia je potom rozdelené medzi jednotlivých užívateľov. Rovnako aj prípadná odmena za nájdenie vyhovujúceho riešenia. Z dôvodu súčasnej (veľkej) náročnosti riešenia a s tým súvisiacej výkonnosti siete ide o efektívny prístup k ťažbe a je dnes najpoužívanejší. Preto je táto práca zameraná práve na princíp *pooled mining*.

3. Ťažobné protokoly

Každá kryptomena špecifikuje vlastný protokol a sieť, prostredníctvom ktorej prebieha komunikácia medzi serverom a minermi. V kontexte toho článku chápeme server ako poskytovateľa *pool-u*. Formát komunikácie špecifikujú ťažobné protokoly – *Getwork*, *Getblocktemplate*, *Stratum*. Tieto protokoly tvoria základný kameň súčasných sietí kryptomien. Stručný popis jednotlivých protokolov je uvedený v nasledujúcej časti.

Najstarším protokolom navrhnutým pre *pooled mining* ťažbu je protokol *Getwork*. Pozostáva z jednej JSON-RPC[1] metódy, pomocou ktorej je vykonávaná požiadavka na prácu zo strany minera. Server následne zašle odpoveď obsahujúcu hlavičku bloku. Logika ťažby spočíva iba v inkrementáci hodnoty *nonce*. Po nájdení vyhovujúceho riešenia je toto odoslané serveru [4]. Ťažbu protokolom *Getwork* možno definovať Algoritmom 1. Obmedzenie protokolu *Getwork* spočíva

Algorithm 1 Algoritmus ťaženia kryptomeny protokolom Getwork

1: send *getwork* method without parameters
2: receive and decode data: $originalBlock\_header, target$
3: $nonce \leftarrow 0$
4: $block\_header \leftarrow originalBlock\_header + nonce$
5: $result \leftarrow hash(hash(\text{block\_header}))$
6: **while** $result \geq target$ **do**
7: $\quad nonce \leftarrow nonce + 1$
8: $\quad block\_header \leftarrow originalBlock\_header + nonce$
9: $\quad result \leftarrow hash(hash(\text{block\_header}))$
10: **end while**
11: submit share using "method": "getwork"and "params": "*result*"

v nemožnosti generovania práce lokálne, bez potreby

[1] Popis: `http://www.jsonrpc.org/`

kontaktovania servera. Pri výkone súčasného hardvéru tu vzniká obmedzenie v podobe generovania veľkého množstva komunikácie. V základnej podobe je *Getwork* schopný obslúžiť výkonnosť maximálne 4.2 GHash/s.

S efektívnejším riešením prišiel protokol *Getblocktemplate*. Rovnako ako *Getwork* využíva volania JSON-RPC nad ktorým implementuje dve základné metódy. Prvá má názov `getblocktemplate`. Po volaní je klientovi dodaná šablóna bloku, podľa ktorej je schopný generovať prácu lokálne. Toto riešenie vo veľkej miere odstraňuje nedostatky predchádzajúceho protokolu a dovoľuje obslúžiť výkonnosť súčasného hardvéru. Generovanie lokálnej práce môžeme popísať Algoritmom 2, ktorý spracúva dáta z odpovede na požiadavku metódy `getblocktemplate`.

Táto obsahuje zoznam transakcií, verziu, referenciu na predchádzajúci blok, aktuálny čas, obtiažnosť apod. Na strane minera je vypočítaný nový koreň stromu, tzv. *MerkleRoot*. Ten zahŕňa transakcie, ktoré si miner môže zvoliť podľa svojich preferencií. Následne je zostavená hlavička nového bloku a dochádza k samotnému hľadaniu *nonce* (podobne ako pri protokole *Getwork*). V momente nájdenia vhodného riešenia prichádza na rad volanie druhej metódy `submitblock`. Touto miner potvrdzuje nájdené riešenie. Obsah správy pozostáva z konkatenácie hlavičky bloku, počtom transakcií a ich identifikátormi.

Tento prístup prináša niekoľko výhod v porovnaní s predchádzajúcim riešením. Určite môžeme hovoriť o decentralizácií, pretože ťažba už nie je riadená výlučne na strane poolu. S tým súvisí už spomínaná obsluha výkonného hardvéru, ktorý nie je protokolom limitovaný. Taktiež prináša výrazné zníženie zaťaženia siete a *pool* serverov. Samotný návrh protokolu počítal s možnými rozšíreniami a tak poskytuje lepšiu škálovateľnosť v prípade budúcich modifikácií [5].

Dnes najrozšírenejším protokolom je *Stratum*. Ten je postavený na podobnej myšlienke ako *Getblocktemplate*, teda poskytuje možnosť generovania práce lokálne. Na rozdiel od neho však nepracuje nad protokolom HTTP, ktoré nie je príliš efektívne v obsluhe častých dotazov. Operuje priamo nad sieťovou vrstvou TCP, kde je nadviazané spojenie v rámci ktorého prebieha výmena JSON-RPC správ verzie 2.0. Minerovi poskytuje menšie množstvo informácií ako protokol *Getblocktemplate*. Vymieňajú sa iba nevyhnutné dáta, z ktorých je možné zostaviť blok [6] (coinbase[2] transakcie, časti *MerkleTree* pre vytvorenie

Algorithm 2 Pseudokód generovania share a hlavičky bloku v protokole Getblocktemplate

receive and store block template data: *coinbasetxn*, *target*, *transactions*, *version*, *prevBlk_hash*, *time*
coinbase_tx ← *coinbasetxn*[0 : 41]
ctx_lnt ← *coinbasetxn*[42]
tx_data ← *coinbasetxn*[43 : *ctx_lnt*]
▷ read data from coinbasetxn
coinbase ← *coinbase_tx* + (*ctx_lnt*{+length(*extra_tx*)} + *tx_data*{+*extra_tx*+length(*extra_tx*)}
▷ parameters in {} are optional
tx_list ← []
merklehashes ← [] ▷ define new lists
for all *tx* ∈ *transactions* **do**
    tx_list += *coinbase* + *tx*
end for
for all *tx* ∈ *tx_list* **do**
    merklehashes += (hash(hash(*tx*))
end for
while length(*merklehashes*) > 1 **do**
    if length(*merklehashes*) % 2 **then**
        merklehashes += *merklehashes*.prev()
    end if
    for *i* ← 0 to length(*merklehashes*) **do**
        new_hash ← []
        new_hash +=hash(hash(*merklehashes*[*i*] + *merklehashes*[*i* + 1]))
        i ← *i* + 2
    end for
    merklehashes ← *new_hash*
end while
merkleroot ← *merklehashes*.first()
block_hdr ← *version* + *prevBlk_hash* + *merkleroot* + *time* + *target* + *nonce*
▷ create block header; *nonce* is 0
start mining ▷ find nonce

koreňa, obtiažnosť, ...). Algoritmus vytvárania bloku je jednoduchší ako v prípade protokolu *Getblocktemplate*, viď Algoritmus 3.

Hlavnou výhodou je efektívnejšie využitie prenosového pásma a jednoduchá implementácia. Za obmedzenie môžeme považovať nemožnosť voliť konkrétne transakcie. Vo väčšine prípadov je to však nepodstatné, pretože minera zaujíma maximálny finančný zisk.

[2]Popis: https://www.cryptocompare.com/coins/guides/what%2Dis%2Da%2Dcoinbase%2Dor%2Dgeneration%2Dtransaction/

Algorithm 3 Pseudokód generovania share a hlavičky bloku v protokole Stratum

1: receive $mining.notify$, $mining.set\_difficulty$ and store values
2: randomly generate $extranonce2$ unique for job_id : $extranonce2 < 2^{extranonce2\_size \cdot 8}$
3: $coinbase \leftarrow coinb1 + extranonce1 + extranonce2 + coinb2$
4: $coinbase\_hash \leftarrow$ hash(hash(toBinaryHexString($coinbase$)))
5: $merkle\_root \leftarrow coinbase\_hash$
6: **for all** $i \in merkle\_branch$ **do**
7: $merkle\_root \leftarrow$ hash(hash($merkle\_root$ + toBinaryHexString(i)))
8: **end for**
9: $merkle\_root \leftarrow$ reverseByteOrder($merkle\_root$)
10: $block\_header \leftarrow version + prev\_hash + merkle\_root + time + target + nonce$ ▷ $nonce$ is 0
11: start mining ▷ find nonce

4. Katalóg

Katalóg kryptomien vznikol v rámci projektu Tarzan, ktorý sa zaoberá vývojom integrovanej platformy pre spracovanie digitálnych dát z bezpečnostných incidentov [7][8]. Jeho úlohou je uchovanie informácií súvisiacich s ťažbou kryptomien. Nástroj je určený ako zdroj dát pre sieťových administrátorov, orgány činné v trestnom konaní, bezpečnostné agentúry apod.

Existujúce riešenie je implementované ako webová aplikácia[3] v programovacom jazyku *PHP*, využívajúca framework *Laravel*[4]. Tá je previazaná s *MySQL* databázou, v ktorej sú uchované záznamy o kryptomenách, pooloch, serveroch, portoch a adresách. Databázová schéma popisujúca vzťahy medzi jednotlivými entitami je viditeľná na obrázku 2 (vrátane modifikácií). Katalóg je naplnený množstvom dát, pokrývajúcich veľkú časť ťažobných poolov a kryptomien. Dáta boli volené s ohľadom na najväčšiu distribúciu hash rate v rámci siete kryptomeny.

Nástroj poskytuje administrátorské rozhranie, ktoré je sprístupnené po prihlásení do systému. Užívateľ je potom schopný pridávať nové a upravovať existujúce záznamy (kryptomeny, pooly, servery, porty) katalógu a tiež zobraziť detail záznamov.

Tabuľka adries je systémom napĺňaná automaticky. Katalóg získava z doménovej adresy serveru (FQDN) zoznam IP adries prostredníctvom *PHP* funkcie `gethostbynamel`. Pre danú IP adresu však nemusí platiť, že na nej „počúva“ ťažobný server. Z toho dôvodu bola navrhnutá a implementovaná modifikácia popísaná v Kapitole 5.

[3] http://smashed.fit.vutbr.cz
[4] Popis: https://laravel.com/

5. Modifikácie

Vzhľadom k tomu, že nástroj aktuálne neposkytuje spôsob, ktorým by overoval skutočnú dostupnosť ťažobných informácií, na uložených serveroch, bol vypracovaný návrh rozšírenia doplňujúci túto funkcionalitu. Cieľom je implementovať aktívne testovanie každej adresy a portu, ktorého výsledkom bude informácia o stave/dostupnosti ťažobného protokolu. Informácia o stave bude aktualizovaná v pravidelných intervaloch (napr. každé 3 hodiny), s tým že bude zaznamenaný stav po každej aktualizácii.

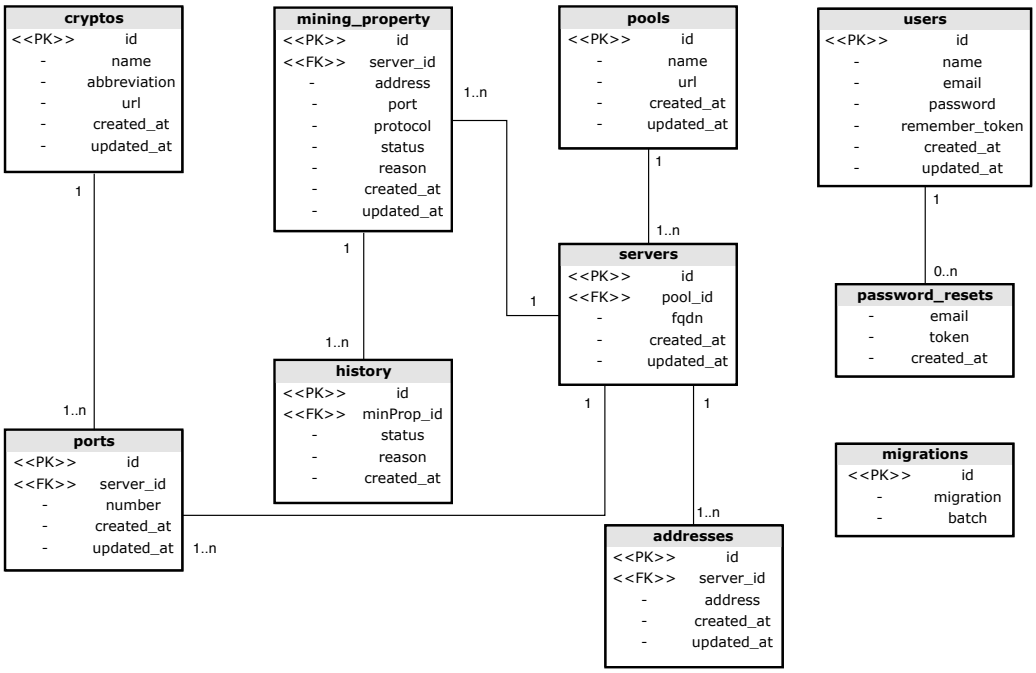
Základom implementácie rozšírenia bolo pridanie dvoch tabuliek (*MiningProperties* a *History*). Tabuľka *MiningProperties* uchováva informácie o dostupnosti služby pre dvojicu <ip adresa, port>, ktoré sú v relácií ku rovnakému serveru. Službu môžeme definovať ako spustenú inštanciu ťažobného protokolu (*Stratum*, *Getblocktemplate*, *Getwork*). Do tabuľky *History* sú ukladané údaje v momente obnovenia záznamov v *MiningProperties*. Uchováva teda informácie o histórií stavov. Na základe tejto histórie potom bude možné doplniť prípadnú funkcionalitu pre mazanie adries, portov, serverov, atď.

Do aplikácie bol pridaný nový kontrolér, ktorý reaguje na požiadavky vrstvy *view* (model MVC[5]) a navracia potrebné dáta. Na webovú stránku bola doplnená nová karta (Mining Properties: Index), ktorá zobrazuje informácie o dostupnosti služby. Prihlásenému užívateľovi navyše sprístupňuje ovládací prvok pre obnovenie dát. Taktiež je možný detailný náhľad konkrétneho záznamu (po kliknutí na ID), ktorý zobrazuje všetky údaje z databáze a informácie o histórií stavov tohoto záznamu.

Pre indexy všetkých kariet (*crypto*, *pool*, *server*, *address*, *port*, *miningProp*) bola definovaná nová cesta `/json`, ktorá sprístupňuje záznamy databáze vo formáte JSON.

Údaje o dostupnosti služby je možné obnoviť kliknutím na tlačidlo *refresh data* v indexe Mining Properties. Následne je volaný kontrolér, ktorý iniciuje novú úlohu (job)[6]. Táto je spustená na pozadí vďaka čomu nie je narušená interakcia užívateľa aplikácie. Vzhľadom ku množstvu uložených dát môže aktualizácia trvať niekoľko minút. Aktívne dotazovanie na ťažobné informácie je implementované v samotnej úlohe. Úloha je okrem požiadavky užívateľa

[5] Popis: https://doc.nette.org/cs/2.4/presenters
[6] Popis: https://laravel.com/docs/5.6/queues



Obrázok 2. Databázová schéma rozšíreného katalógu.

spúšťaná v pravidelných intervaloch (každé 3 hodiny) za použitia plánovača[7] frameworku *Laravel*.

Tabuľky *Servers*, *Ports*, *Addresses* sú načítané pomocou dotazu na databázu za použitia klauzule `join`. Pokračuje sa iterovaním nad získanou množinou dát, kde je pre každý záznam vykonaný dotaz na ťažobný protokol. Existujúce záznamy sú následne aktualizované, prípadne vložené nové. Vykoná sa tiež záznam stavu do tabuľky *History*.

Vzhľadom k rozdielom medzi jednotlivými protokolmi je vyžadovaná špecifická implementácia pre každý protokol. Overenie protokolu Stratum prebieha nadviazaním TCP spojenia so serverom a postupným zasielaním JSON-RPC volaní na metódy protokolu používané pre iniciovanie ťažby, prihlásenie atď. Základnou metódou je *mining.subscribe*, ktorej správa má formát: `{"jsonrpc": "2.0", "id": 1, "method": "mining.subscribe", "params": []}`. V prípade prijatia odpovede od servera je zrejmé že, server na tejto adrese/porte „počúva“. Prijatú správu je však potrebné zanalyzovať a na základe jej obsahu určiť, či sa jedná o aktívny server. V prípade negatívneho výsledku sa pokračuje odoslaním žiadosti pre inú metódu. Komplikáciou tohoto prístupu je fakt, že každá kryptomena môže používať protokol Stratum v modifikovanej verzii. To znamená rozdielny názov metód, parametrov apod. Z toho dôvodu bolo do katalógu implementované „trojstavové“ riešenie, kde stav *UP* znamená aktívny server, stav *DOWN* server, ktorý neodpovedá na dotazy (alebo je nedostupný) a stav listen vyjadrujúci, že server počúva na danej adrese a porte, avšak získanú odpoveď sa nepodarilo klasifikovať a je teda pravdepodobné, že tu môže byť aktívna ťažobná služba.

Keďže protokol Getblocktemplate (Getwork) pracuje nad HTTP, pri implementácií bude použitá knižnica *cURL*. Uvedená knižnica dovoľuje vytvárať HTTP požiadavky a spracovávať odpovede od servera. Obsah správy tvorí JSON-RPC dotaz s metódou `getblocktemplate` (`getwork`). Podobne ako v predchádzajúcom prípade prebehne analýza odpovede, na základe ktorej bude klasifikovaný stav dotazovaného servera. Ak server na zaslaný dotaz neodpovie, možno usúdiť, že sa na ňom nenachádza spustená implementácia ťažobného protokolu. V aktuálnom riešení nie je zahrnutá implementácia dotazovania na tento protokol. Stav jednotlivých záznamov je implicitne nastavený na *DOWN*.

6. Experimenty

V tejto časti je popísané testovanie a výsledky implementácie rozšírenia. Testovanie prebiehalo na vzorke dát, aktuálne uložených v katalógu t.j. 1102 záznamov tabuľky *MiningProperties*. Jeden cyklus obnovenia dát katalógu trval približne 15 minút. Distribúcia stavov pre protokol Stratum je nasledovná:

- *UP* – 652
- *LISTEN* – 210
- *DOWN* – 240

[7] Popis: https://laravel.com/docs/5.6/scheduling

Z výsledkov je zrejmé, že viac ako polovica dvojíc <adresa, port> je aktívna. Takmer štvrtina adries je nedostupná a o podobnom počte adries nie je možné s určitosťou rozhodnúť, či očakávajú ťažobnú komunikáciu. Adresy môžu medzi jednotlivými stavmi v čase oscilovať. Dôvodom môže byť migrácia serverov/adries v datacentrách alebo ich dynamická konfigurácia. Vďaka rozšíreniu je možné dohľadanie stavu jednotlivých adries v čase, čo je veľkým prínosom z hľadiska forenznej analýzy a použitia tohoto nástroja bezpečnostnými zložkami, sieťovými administrátormi apod.

Okrem stavu servera je vo výstupe testu zahrnutá aj položka *reason*, ktorá nesie informáciu o testovaných metódach protokolu s príslušným návratovým kódom (0 – DOWN, 1 – UP, 2 – LISTEN) a chybovou hláškou. Testovanie končí v momente, keď je dosiahnutý návratový kód 1, alebo po otestovaní všetkých metód. Do aktuálneho stavu *mining property* sa zapíše vždy „najlepší“ zistený stav.

Prístup do administrátorského režimu katalógu je možný po prihlásení sa užívateľským menom *excel2018* a heslom *excel2018_pass*.

7. Záver

Úvodná časť práce je venovaná teoretickému popisu všeobecných princípov kryptomien a ich ťažby. Ďalej je spracovaná analýza základných ťažobných protokolov, ktorá vysvetľuje ich funkcionalitu, použitie a výhody/nevýhody.

Praktická časť práce je zameraná na popis existujúceho systému (katalógu) a tiež na návrh a implementáciu modifikácií tohoto katalógu. Cieľom modifikácií bolo zvýšiť spoľahlivosť uchovávaných dát pridaním časovej roviny ku uloženým informáciám. Korektnému návrhu popísaných rozšírení predchádzalo štúdium a pochopenie problematiky kryptomien, ťažby a používaných protokolov.

Do súčasného riešenia bola úspešne doplnená funkcionalita, ktorá pravidelne vykonáva aktívne testovanie dostupnosti uložených záznamov. Výstupom je aktuálny stav dostupnosti ťažobného servera pre špecifickú adresu/port. Katalóg bol tiež doplnený o záznam histórie stavov, ktorý poskytuje prehľad dostupnosti ťažobných protokolov v časovej rovine. V aktuálnej implementácií je zahrnutá klasifikácia dostupnosti pre protokol Stratum. Rovnaká funkcionalita pre protokoly Getblocktemplate a Getwrok bude predmetom pokračovania práce.

Hlavným prínosom pridanej funkcionality je zlepšenie celkovej použiteľnosti riešenia v oblasti forenznej analýzy sieťovej prevádzky.

V budúcnosti môže byť katalóg doplnený o ďalšie funkcie ako napríklad automatické generovanie ACL pre sieťové prvky, záznam veľkosti hashrate jednotlivých poolov atď.

Poďakovanie

Rád by som poďakoval svojmu vedúcemu Ing. Vladimírovi Veselému, Ph.D. za jeho pomoc a vedenie pri vypracovaní tejto práce.

Literatúra

[1] Usman W. Chohan. *Cryptocurrencies: A Brief Thematic Review*. Paper, Aug 2017. https://ssrn.com/abstract=3024330.

[2] Martin Očenáš. *Analytické zpracování blockchainu kryptoměn*. Master's thesis, Vysoké učení technické v Brně, Fakulta informačních technologií, 2017. http://www.fit.vutbr.cz/study/DP/DP.php?id=19354.

[3] Bitcoin Project. Bitcoin developer guide. online, 2017. https://bitcoin.org/en/developer-guide#block-chain-overview.

[4] bitcoinWiki. Getwork mining protocol. online, 2015. https://en.bitcoin.it/wiki/Getwork.

[5] bitcoinWiki. Getblocktemplate mining protocol. online, 2015. https://en.bitcoin.it/wiki/Getblocktemplate.

[6] Slush Pool. Stratum těžební protokol. online, 2017. https://slushpool.com/help/manual/stratum-protocol.

[7] FIT VUT. Integrovaná platforma pro zpracování digitálních dat z bezpečnostních incidentů. online, 2017. http://www.fit.vutbr.cz/units/UIFS/grants/index.php.cs?id=1063.

[8] Vladimír Veselý. Forenzní analýza bitcoinu (kryptomeny). online, 2017. http://www.fit.vutbr.cz/units/UIFS/grants/index.php.cs?file=%2Fproj%2F1063%2FSeminar-06-2017%2FTARZAN-krypto.pdf&id=1063.