Discovering and Clustering Hidden Time Patterns in Blockchain Ledger

Anna Epishkina and Sergey Zapechnikov
National Research Nuclear University (Moscow Engineering Physics Institute), Moscow, Russia
AVEpishkina@mephi.ru, SVZapechnikov@mephi.ru

Abstract
Currently, immutable blockchain-based ledgers become important tools for cryptocurrency transactions, auditing, smart contracts, copyright registration and many other applications. In this regard, there is a need to analyze the typical, repetitive actions written to the ledger, for example, to identify suspicious cryptocurrency transactions, a chain of events that led to information security incident, or to predict recurrence of some situation in the future. We propose to use for these purposes the algorithms for T-patterns discovering and to cluster the identified behavioral patterns subsequently. In case of having labeled patterns, clustering might be replaced by classification.

Keywords: **audit trails, blockchain, data mining, classification, clustering**

1 Introduction

The advent of decentralized cryptocurrencies started from Bitcoin (Nakamoto, 2008) has brought a lot of blockchain-based systems and databases. It is created and maintained through network consensus and can be used as a public ledger.

The most widely known blockchain-based decentralized databases are BigchainDB (McConaghy, 2016), BlockCypher, Openchain etc. Almost all of them provide some form of API which can be used for developing distributed application.

Besides blockchain-based databases, applications may also use decentralized file systems such as IPSF (InterPlanetary File System) or BitTorrent-like storage systems.

The further important adventure in the area of decentralized computations are generalized decentralized virtual machines like Ethereum (Wood, 2017). It provides the possibility to support two types of accounts. They are manually managed Externally Owned Accounts (EOAs) and automatically executable Contact Accounts (CAs). The latter is able to execute special programs in byte codes of Ethereum Virtual Machine (EVM) called smart contracts. It is very important that EVM language is Turing-complete, so in principle any possible functionality can be realized through smart contracts. Ethereum platform supports a lot of programming languages such as Solidity, Serpent, Mutan etc.

There are some ideas for more advanced privacy-preserving smart contract platforms, i.e. Hawk (Cosba, 2015) and Enigma (Zyskind, 2016) but no one of them has implementation yet.

The novelty of such platforms like Ethereum is their ability to provide immutable storage for any transactions among the accounts. These transactions may be not only financial but any other involving state changes of EOA of CAs. Besides that, blockchain-based application ensures perfect integrity for all data or references stored in blockchain and high availability of system services. In particular, such platforms have a lot of applications in Cybersecurity including audit trails management and many others.

The rest of the paper is organized as follows. In section 2 we consider related works. In section 3 we suggest a technique to apply T-pattern analysis for discovering hidden behavior patterns in audit logs. In section 4 we discuss how to evaluate distance among such patterns and to cluster them. Section 5 is about the clustering-based anomaly detection in behavior patterns. We give the conclusion and future research directions in section 6.

2 Related Works

There are a lot of applications of blockchain-based techniques in the area of cybersecurity. Some projects such as ShoCard and ChainAnchor (Hardjono, 2016) provide blockchain-based identity management and anonymous permissions. Others like DECENT (Michalko, 2015) are decentralized content distribution systems. One more area for blockchain is distributed Certification Authorities and certificate validation systems (Matsumoto, Reischuk, 2016). There are some other applications such as software authentication and version control, IoT devices authentication, data provenance, secure messaging and so on.

However, the most evident cybersecurity application of blockchain is events' tracing and audit (Cucurull, 2017). Traditional logs can be replaced by immutable ledger storing system's events history. Every audit log should be useful for event incident management. For instance, if log contains security events it should be possible to investigate a chain of events leading to the incident. That is why blockchain transactions should also be auditable.

We suggest a technique for blockchain ledger auditing based on computation of hidden behavior patterns discovery, clustering them and outlier detection. We assume blockchain-based audit log because currently blockchain is the most prominent tool for auditing. However, such or similar technique may quite evidently be used for analysis of traditional read-only service or network events log. In our work, we use Magnusson's T-pattern discovery technique (Magnusson, 2000) followed by a standard technique for agglomerative clustering and cluster-based outlier detection (Han, 2012).

3 Discovery Hidden Patterns in Audit Logs

As it is well known, blockchain ledger growths by new block in regular time intervals. The length of interval may be different from 10 minutes in Bitcoin to near 1 second in BigchainDB. The number of transactions is one block may also be different, in the limit decreasing to 1 transaction per block, reducing from blockchain, to transaction chain.

So, ever blockchain-based audit trail can be represented as a sequence of events, each placed into one of the time intervals. One interval can contain strictly one event or a lot of events. Such log may contain huge amount of different events. Thus, it may be very difficult to trace a sequence of events preceding some other event. The technique of T-pattern discovery introduced by M. Magnusson (Magnusson, 2000) can be used to solve the task. Let N be the length of time interval (measured in time slots) and A_1, A_2 be two security events. Let N_{A_1} be the number of blocks where the A_1 event

was written to blockchain ledger (among the total amount of N blocks). Then $P(A_1) = \frac{N_{A_1}}{N}$ is the frequency of A_1 event occurring (this is the evaluation of A_1 probability). So, $P(\overline{A_1}) = 1 - P(A_1)$ is the frequency of the A_1 event non-occurring. Thus $P(\overline{A_1})^t$ is the estimated probability of the event non-occurring during t segmental slots, where $t = t_2 - t_1 + 1$ and $1 - P(\overline{A_1})^t$ is the estimated probability of at least one occurring of A_1 during this interval. Let $f(k, p, n) = C_n^k p^k (1-p)^{n-k}$ be polynomial distribution of occurring *k*-of-*n* events each of which has probability p. Thus, the apriory probability that A_1 event will occur at least N_{AB} times followed by A_2 event in the next t blocks is

$$P = 1 - \sum_{i-0}^{N_{AB}-1} f\left(N_A, i, P(\overline{A_2})^t\right), \quad (1)$$

where

$$f\left(N_A, i, 1 - P(\overline{A_2})^t\right) = C_{N_A}^i (1 - P(\overline{A_2})^t)^i P(\overline{A_2})^{t(N_A - i)}. \quad (2)$$

This probability should be compared with the actual frequency of event occurring to decide if it is above or below the significance level. Let's show, how it can be evaluated. According to (Magnusson, 2000) there are four cases:

- Event A_2 occurs after a series of events A_1 only once, so no pattern exists;
- If there are no N_{AB} cases when event A_2 occurs by some blocks after event A_1, we can take minimum and maximum distance between A_1 and A_2 (in blocks) and evaluate P using (1);
- If this probability is quite significant, this is most likely random coincidence;
- If it is not significant (i.e. less than 0.05), this is likely pattern.

What can be seen as different type of events? It strongly depends on the application. For instance, if we trace financial transactions, it may be a transfer from one certain account to another.

After mining a pair of certain events A_1, A_2 the every discovered pattern $B = A_1 \to A_2$ (where $\to$ is a sign denoting that A_2 is following A_1) can be thought as one event. The second-order patterns can be discovered similarly. In such pattern every compound event B became the first or the second part of wider pattern where the remaining part is also elementary or compound event. Thus the tree of events grows. It can be visualized by means of dendrogram. After all, each pattern can be written as $S = A_{i_1} \to A_{i_2} \to ... \to A_{i_k}$.

4 Clustering Behavior Patterns

For auditing, discovered behavior patterns should be analyzed further. We suggest that some different pattern types may be discovered, moreover, repeating patterns may be not strictly equal one to other, but rather similar. For this purpose, Levenshtein distance (Levenshtein, 1966) may be used as a measure of similarity between the patterns. Let S_1 and S_2 be two event strings with lengths M and N accordingly. The Levenshtein distance is defined as follows

$$D(i,j)=\begin{cases}0, if\ i=0,\ j=0\\ i, if\ i>0,\ j=0\\ j, if\ i=0,\ j>0\\ \min\{D(i,j-1), D(i-1,j)+1, D(i-1,j-1)\}+m(S_1[i], S_2[j]), if\ i>0,\ j>0\end{cases} \quad (3)$$

where $m(a,b)=0, if\ a=b$ and $m(a,b)=1, if\ a\neq b$.

Using Wagner – Fischer's algorithm (Wagner, 1974) optimal distances among event strings can be evaluated. Thus, after that we have a matrix of distances between the discovered event strings.

After that, any type of agglomerative hierarchical clustering may be used. We recommend to use Ward's method (Ward, 1963) because of its monotonicity and tension property. As it is well known, the best visual technique for hierarchical clustering is dendrogram, so the optimal number of clusters may be found easily as maximum inter-cluster distance.

For example, Lance – Williams algorithm () may be used. It starts from n one-element clusters where n is the number of different patterns. On each step two clusters U,V with minimal distance are united into one cluster W. Let Z be any other cluster that is not merged on this step. The Ward's distance is defined as

$$R(W,Z)=\frac{|Z||W|}{|Z|+|W|}D^2\left(\sum_{w\in W}\frac{w}{|W|},\sum_{z\in Z}\frac{z}{|Z|}\right) \quad (4)$$

where $D(u,v)$ is Levenstein's distance between patterns, $|Z|,|W|$ are cardinalities of Z and W clusters.

5 Anomaly Detection in Behavior Patterns

As the rule, the main purpose of audit is discovering a kind of non-typical behavior. Therefore, outlier detection is the main technique for solving such task. Clustering-based outlier detection using distance to the closest cluster is more convenient choice for our case. For each pattern S we can assign it an outlier score according to the distance between the pattern and the center of a cluster that is closest to the pattern (Tao, 2006). For this purpose, the centers of all discovered clusters should be evaluated. In our case, the center of each cluster is the mean value of Levenshtein distance of all cluster's patterns from the null pattern. Suppose that closest center to S is C_S, the distance between them is $D(S,C_s)$ and the average distance between C_S and the patterns assigned to cluster is L_{C_S}. The ratio $\frac{D(S,C_s)}{L_{C_S}}$ will be a measure how long is the pattern S, stands out from the average distance. The larger this ratio, the relatively farther away pattern S is from the cluster's center, so it is more likely that S is an outlier.

6 Conclusion

Finally, we have the following technique for discovering and clustering hidden time patterns in blockchain ledger.

1. Repeating event sequences S are discovered in blockchain ledger using Magnusson's technique of T-pattern discovery. Pattern can be discovered if it repeats no less twice. The database D of event sequences is created.

2. For each pair of sequences Levenstein's distance is evaluated using Wagner-Fischer's algorithm. In particular, Levenstein's distance between each sequence and null sequence should be defined. The distance between all sequences is append to the database D.

3. Event sequences are combined in clusters using Lans – Williams agglomerate hierarchical algorithm and Ward's distance between clusters. The optimal number of clusters is evaluated using maximal inter-cluster distance. The membership of sequences in clusters is appended to the database D.

4. For each cluster, its center and the average distance between the center and the sequences assigned to cluster is evaluated and appended to the database D. Outliers (anomaly sequences) are discovered as objects with high ratio of the distance between the object and the nearest center of a cluster and the average distance between the sequences and the center of a cluster. These outliers are most likely abnormal event sequences that may be traces of attacks, user's errors and so on.

The main advantage of the suggested technique is that it does not require any additional parameters and may be fully automated. The main drawback is that any sequence to be clustered or estimated as anomaly must repeat at least twice. So, absolutely new anomalous sequences of events could not be discovered by the technique. Thus, further research may be associated with prediction of completely new sequences of anomalous events.

Acknowledgement

This work was supported by Competitiveness Growth Program of the Federal Autonomous Educational Institution of Higher Education National Research Nuclear University MEPhI (Moscow Engineering Physics Institute).

References

Cosba A. et al. (2015). *Hawk: The Blockchain Model of Cryptography and Privacy-Preserving Smart Contracts.* URL: http://eprint.iacr.org/2015/675

Cucurull J., Puiggali J. (2017). *Distributed immutabilization of secure logs.* URL: https://www.scytl.com/wp-content/uploads/2017/01/Distributed-Immutabilization-of-Secure-Logs_Scytl.pdf

Han J., Kamber M., Pei J. (2012). *Data mining: Concepts and techniques.* Morgan Kaufmann Pubs. pp. 740.

Hardjono T., Smith N., Pentland A. (2016). *Anonymous Identities for Permissioned Blockchains.* URL: http://connection.mit.edu/wp-content/uploads/sites/29/2014/12/Anonymous-Identities-for-Permissioned-Blockchains2.pdf

Lance G., Williams W. (1967). *A general theory of classificatory sorting strategies.* 1. Hierarchical systems, The Computer Journal, Vol. 9, No. 4, pp. 373–380

Levenshtein, V. I. (1966). *Binary codes capable of correcting deletions, insertions, and reversals.* Soviet Physics Doklady. Vol. 10 (8), pp. 707–710. (in Russian)

Magnusson M. (2000). *Discovering hidden time patterns in behavior: T-patterns and their detection.* Behavior Research Methods, Instruments, & Computers. 2000, Vol. 32, Issue 1, pp. 93–110.

Matsumoto S., Reischuk R. (2016). IKP: *Turning a PKI Around with Blockchains.* URL: http://eprint.iacr.org/2016/1018

McConaghy T. et al. (2016). *BigchainDB: A Scalable Blockchain Database.* URL: https://www.bigchaindb.com/whitepaper/bigchaindb-whitepaper.pdf

Michalko M., Sevcik J. (2015). *DECENT whitepaper*. URL: http://www.the-blockchain.com/docs/Decentralized%20Open-Source%20Content%20Distribution%20(DECENT)%20whitepaper.pdf

Nakamoto S. (2008). *Bitcoin: A Peer-to-Peer Electronic Cash System*. URL: https://bitcoin.org/bitcoin.pdf

Tao Y., Xiao X., Zhou S. (2006). *Mining distance-based outliers from large databases in any metric space.* In Proc. 2006 ACM SIGKDD Int. Conf. Knowledge Discovery in Databases (KDD'06), pp. 394–403, Philadelphia, PA, Aug. 2006.

Wagner R., Fischer M. (1974) *The string-to-string correction problem.* Journal of the As~,ociation for Computing Machinery, Vol. 21, No. I, January 1974, pp. 168-173.

Ward, J. H. (1963). *Hierarchical Grouping to Optimize an Objective Function*. Journal of the American Statistical Association, Vol. 58, pp. 236–244.

Wood G. (2017). *Ethereum: A secure decentralized generalized transaction ledger.* URL: http://gavwood.com/paper.pdf.

Zyskind G., Nathan O., Pentland A. (2016). *Enigma: Decentralized Computation Platform with Guaranteed Privacy*. URL: http://www.enigma.co/enigma_full.pdf