Identification of Unusual Wallets on Ethereum Platform*

Mikhail Petrov

National Research University Higher School of Economics
Myasnitskaya 20, Moscow, Russia
mishanayrus@gmail.com

Abstract. In this work we introduce a suspiciousness rating of Ethereum wallets. The rating is based on different characteristics of the wallets and the transactions they were involved in during a week. To achieve this goal atypical vertices of the transactions graph for the Ether cryptocurrency need to be discovered, so, first, we identify typical groups of nodes. Then the nodes which are far different from all these groups are considered to be suspicious.

Keywords: Ethereum platform · cryptocurrency transactions · transactions graph.

1 Introduction

Anomaly detection is an area that has been receiving much attention in recent years. It has a wide variety of applications, including fraud detection, network intrusion detection, medical diagnosis and other fields. Usually research in this area is using attribute-value data as the medium from which anomalies are to be extracted. Some works are focused on anomaly detection in graph-based data. In our paper we are going to combine these approaches for the goal of identification of suspicios wallets in a cryptocurrency community.
[1]

1.1 Cryptocurrencies and Ether

The area of cryptocurrency is quite young. The first appearance of Ether, the transactions graph of which will be analyzed in this article, appeared in 2013. By now the analysis of the Ether community has progressed very little. One can find quite many articles on the analysis and prediction of cryptocurrencies rates, but so far few research has been focused on the analysis of exchange communities for the cryptocurrencies.

It is known that the semantics of transactions of blockchain systems can be captured by a transaction graph, see e.g. [1]. Such a graph generally consists of

the states and the transactions as transitions between the states, together with conditions for the consistency and validity of transactions.

Detailed overview of cryptocurrencies could be found in [4].

1.2 Overview of anomaly detection methods

In [8] a comprehensive survey of recent anomaly detection systems and hybrid intrusion detection systems is provided, and recent technological trends in anomaly detection are also discussed. In [2] existing techniques are grouped into different categories based on the underlying approach adopted by each technique. For each category key assumptions are identified, which are used by the techniques to differentiate between normal and anomalous behavior. In [7] two techniques for graph-based anomaly detection are introduced. The authors suggest a new method for calculating the regularity of a graph, with applications to anomaly detection. Experimental results are provided which use both real-world network intrusion data and artificially-created data. In [5] several information-theoretic measures, namely, entropy, conditional entropy, relative conditional entropy, information gain, and information cost for anomaly detection are used.

1.3 Networks analysis

As it was already mentioned, see [7], some anomaly detection method use graph-based analysis. Besides standard graph analysis library networkx, we are also using statistical properties of social networks, see [3, 6].

1.4 Structure of the remaining text

The rest of the paper is organized as follows. Section 2 describes the data we worked with. Section 3 describes the methods we used to analyze transaction data. Section 4 presents a description of how the points for the strangeness rating were calculated. Section 5 shows the top 5 vertices as a result of calculating the suspiciousness rating with the rating values.

2 The data analyzed

In this paper data on all transactions for a week was downloaded using the open etherium API. Significant characteristics for the community analysis were chosen, and an ether exchange graph was constructed based on them. The vertices of this graph are the participants in the platform and also the edges are the transactions between these participants. Further in the received graph, the characteristics of the vertices were calculated, clusters were found, and associative rules based on them were constructed. As a result, taking into account the data on graph vertices, clustering, and associative rules, the suspiciousness rating of vertices was constructed.

Since in this paper we were focused on identifying suspicious and untypical members in the ether exchange community, only those characteristics that were useful for analyzing and constructing the graph were extracted from the set of parameters.

2.1 Ethereum API

Ethereum API provides information about each block, all transactions and every members of the network. JSON RPC API of the Ethereum platform currently supports four programming languages: C++, Go, Python, and Parity. Also there is an option to access the data via web interface at *https://etherscan.io*, which we used.

For our study all transactions executed during a particular week have been downloaded. That was done using the timestamp field of the blocks and the method eth_getBlockByNumber().

Totally information about 3,382,252 transactions were collected.

2.2 Format of Ethereum transaction data

All transactions were selected from each block and then the following four parameters were saved for each transaction:

- address of the sender;
- address of the receiver;
- date and time of the transaction;
- the amount of the internal currency (wei) that is transferred.

These characteristics of transactions allow us to construct the graph of an ether exchange. The vertices in this graph are the addresses of wallets which are either senders or receivers of these transactions, the edges correspond to the transactions themselves.

In this paper we analyze only total amount sent from node A to node B and ignore details about the number of transactions and distribution of the amount between them.

In addition to the total volume transmitted between a pair of wallets, the frequency of interactions between participants (the number of transactions sent) could be considered too.

2.3 Transactions graph

The resulting set of transaction on ether exchange was processed and presented as a graph. The first step to getting the graph was renaming vertices, since addresses do not make any sense. In order to save memory and more clarity, they were numbered with the help of natural numbers, and the dictionary with addresses mapping in the vertex numbers was saved in the file.

Next, we summed the weights of the edges between identical pairs of sender and receiver, recorded the graph as an adjacency list with edge weights and

the total number of transactions between vertices. As a result, we obtained an undirected graph with positive and negative edge weights. The graph has 1,577,010 vertices and 4,963,980 edges. To analyze this graph we used networkx, a common library of the Python language. All further operations for calculating the various characteristics of the graph have been done with the help of this library.

3 Analysis

Our anomaly detection analysis consists of the following four parts:

1. **Graph connectivity analysis.** During this step we put off wallets for which too few information is available, so no meaningful analysis could be conducted for them.
2. **Analysis of the vertex characteristics.** Here we compute degree, centrality and containing k-core for each vertex of the transaction graph.
3. **Cluster analysis.** The clustering is performed based on the vectors obtained in the previous section.
4. **Association rules** are used to find patterns that can be traced in this graph.

After each part all the vertices are ranged according to their suspiciousness, and then these ratings are merged into one.

3.1 Connectivity analysis

The next step was to analyze the connectivity components of the obtained graph. The graph has 35,628 connectivity components. These components can be divided into three groups:

- The main component of a large community. In this component there are 1,474,024 vertices. Most likely, it is a graph of the interaction of people of some large service or exchange, but the exact meaning is not yet clear. This component has the biggest interest for analysis and will continue to be the first in priority.
- The second type is a small groups of ten to a thousand people. Most likely, these groups are small companies in which payment occurs in the cryptocurrency, or small communities that pay cryptocurrency for some services. This group also deserves attention for analysis, but to a lesser extent.
- The third class includes groups of up to ten people. These are some local exchanges of money, betting between people. This group has no interest for analysis, since it does not contain any meaningful information.

As was mentioned above, the first group has the greatest interest for analyzing, therefore it will be analyzed in the future. The remaining connectivity components are stored in separate files and their analysis is possible in future work.

3.2 Vertex Characteristics

Since the main purpose of the work is to distinguish atypical and suspicious vertices, then we must determine the typical aspects of the obtained community. Taking into account that the received connectivity component can have multiple chains of elements with a degree equal to two, therefore the graph has many vertices that are just a link in the chain of transmission of the cryptocurrency, it was decided to allocate the k-cores of the graph of this component. A k-core of a graph G is a maximal subgraph of G in which all vertices have degree at least k. To do this, we run a search of the value of k-cores from 1 to 50 and used the method *k_core* for each value. If the k-core was not found for the value of n, then it will not be found for n + 1, since any k+1-core is also a k-core. Therefore, if such a situation arises in searching the k-cores, then the search can be stopped.

The algorithm stopped at k equal to 15, which means that the k-kernels from 1 to 14 were found. Since the sizes of k-cores beginning with k equal to 8 are already sufficiently small and the 3-core is quite large, 4, 5, 6, and 7 core were chosen as an optimal from the point of view of time for analyzing the graph. Based on the selected k-cores we can calculate new characteristics of the vertices. Three kinds of centrality were calculated for each vertex in the core: betweenness centrality, closeness centrality and degree centrality. Also we add vertex degree as a characteristic. On the basis of the obtained data, we can form a vector that will describe the vertices of the graph (participants of the cryptocurrency community), split each characteristic into groups of segments and create binary vectors for deriving associative rules.

As a result, we obtained a 5-dimensional vector characterizing the vertices of the graph: 3 kinds of centralities; number of the k-core, which vertex belongs; degree of a vertex. Also, the dimensionality of these vectors could be increased by adding distance to the hubs for each of the characteristics, but this operation is planned for the next stages of the work. The resulting vectors can be clustered now.

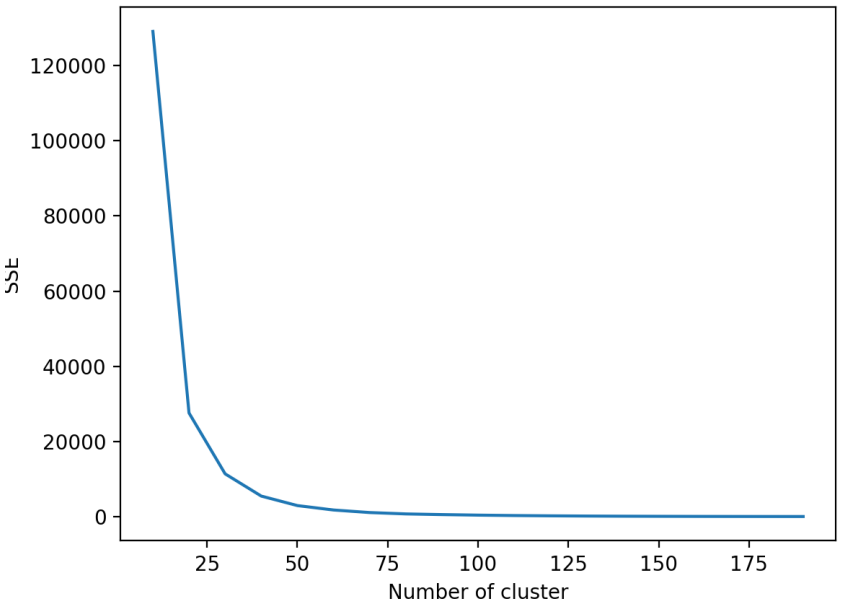
3.3 Clustering

Based on the vectors obtained in the previous section, clustering is performed using the standard k-means method. Since for this method it is necessary to initially know the number of clusters we must first determine this number. To determine the optimal number of clusters all numbers from 0 to 200 in steps of 10 were chosen, clustering was performed and the result were checked with Silhouette and the shoulder methods.

Table 1. The Silhouette Coefficients

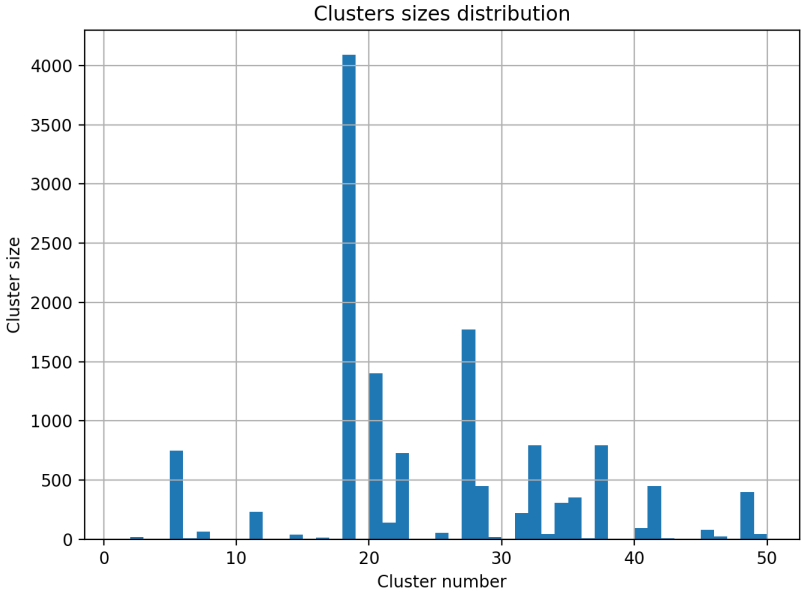
Number of clusters			
Coefficient value			
10	20	30	40
0.6768368853142156	0.7095846965311733	0.8111177029982155	0.8702494303581093
50	60	70	80
0.9146516160235698	0.9293358278581296	0.9387072507702181	0.9478746102234425
90	100	110	120
0.9505421030096185	0.950768946865395	0.7874773068611389	0.7884553347797727
130	140	150	160
0.7698049042610146	0.7467290976039868	0.7992449839069017	0.7467532658132512
170	180	190	
0.7044007926707637	0.6867901637311503	0.665831212711668	

As you can see, this method shows that the most optimal number of clusters is $n = 100$, but not too far from 50 to 100. The result of the shoulder method is shown below:



This chart shows that the optimal amount is $n = 30$. Since the optimal number of clusters depends very much on the type of data, both methods do not always correctly indicate the right result, so the combined solution of the two methods was chosen as the right answer. As a result, the optimal number of clusters was chosen to be 50 and clustering was performed for it using the k-

means method. Distribution of the cluster size can be seen in the figure number 3:

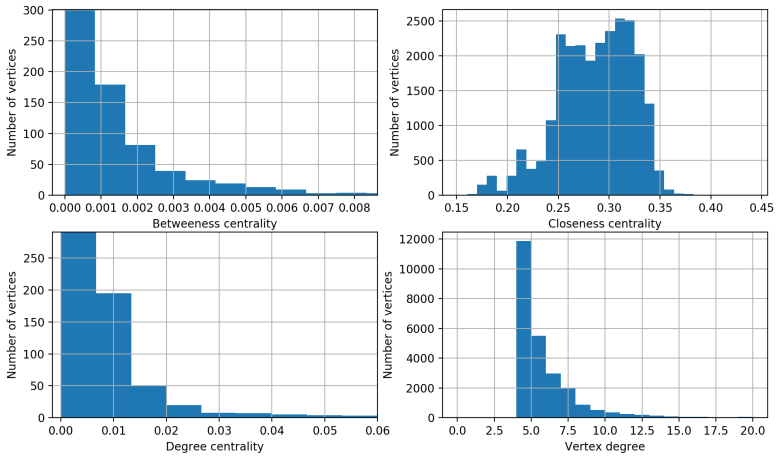


It can be seen from the distribution diagram that one strongly dominant in terms of the size of the elements cluster, two clusters slightly above the average level, about 15 clusters of medium size and the other clusters of a very small size were obtained. These results will be used in the next chapter to build associative rules.

3.4 Association Rules

Now based on all of the characteristics of the community we can try to find patterns that can be traced in this graph. For example, it may happen that if the vertex belongs to the 4-core, then its degree must necessarily be higher than 7. To do this, it is necessary to generate binary vectors in which the element at the i-th position will report whether this element belongs to a certain group or not. Having formed such vectors, we can find patterns, but on first step we need to form the groups themselves. For this, it is necessary to analyze the distribution of the characteristics of the vectors. To achieve this, we should construction distribution histograms for each of the characteristics. The first and third graphs of distributions are strongly shifted to the left, and the thresholds

between the groups are not visible, so we need to look at the left parts closer:



From the received observations it is possible to break each of the characteristics into ranges of values into which approximately equal number of elements will fall. These ranges will also form groups for finding associative rules. The result is the following groups:

Table 2. Groups

Degree centrality	Betweenness centrality	Closenness centrality
$0 < x \leq 0.005$	$0 < x \leq 0.0005$	$0 < x \leq 0.2$
$0.005 < x \leq 0.015$	$0.0005 < x \leq 0.001$	$0.2 < x \leq 0.3$
$0.015 < x$	$0.001 < x$	$0.3 < x \leq 0.4$
		$0.4 < x$

Generation of binary vectors is now carried out. To search for associative rules we used the mlxtend library, which is the apriori method to search for frequent sets of attributes, and then search for associative rules. In our script the rules with a minimum support = 0.8 are taken so that the existing patterns are highly probable. As a result of the launch, two associative rules were found:

Table 3. New rules table

Number	Antecedants	Consequents	Antecedent support	Consequent support
0	1 deg. cen. group	1 bet. cen. group	0.997946	0.994034
1	1 bet. cen. group	1 deg. cen. group	0.994034	0.997946
Support	Confidence	Lift	Leverage	Conviction
0.993876	0.995922	1.001899	0.001884	1.463008
0.993876	0.999841	1.001899	0.001884	12.922448

To get more associative rules, we need to decrease the support threshold, but then there will be less probabilistic associative rules, which will contradict the goal of finding typical signs of this community.

4 Top strangeness rating

Now, when there is a whole set of characteristics of the vertices (clustering results, clustered vectors, associative rules), we can start isolating their typical values and finding all the values that will go beyond this framework. For each overshoot of these boundaries points are entered, which will award the vertices with these atypical characteristics. For each "suspicious" characterization points from 1 to 100 will be given depending on the criticality. At the end, the promised rating of the suspicious vertices of the community graph will be obtained.

Let's start with the simplest - associative rules. Here everything is simple, because if a vector does not obey this rule, then it is atypical for this community. Here, the scores in the scores will be determined by the value of the support of the associative rule multiplied by one hundred.

The next will be the scoring of vertices whose vectors are too far from the center of their clusters. We can calculate the average distance to the center of each cluster and the variance. Accordingly, if the distance of the vector does not fall within the range of the mean ± variance, then this vector is atypical. Points will be calculated according to the following formula:

$$\frac{|mean - distance|}{difference_{max}},$$

$mean$ is the average distance to the center in the cluster
$distance$ - the point to the center of a particular windbreaker
$difference_{max}$ is the maximum difference between the average and the distance from the vector to the center inside the cluster

Next will be scoring the vertices, whose characteristics of the vectors also go beyond the limits of the mean ± variance. Here the same normalized formula will be used, as well as with distances.

The last will be awarded for belonging to atypical small clusters. Since the sizes of some clusters reach even one, this clearly indicates the atypicality of

the vertices in such clusters. Here we can not use the metric, as in the previous two cases, because the dimensions are too scattered, the average is quite heavily shifted to the left and the variance is several times larger than the average, which moves the left border to the minus. With a normal distribution, the segment of the mean ± variance covers approximately 68 percent of the values and for a given number of clusters equal to 50 these 68 percent are 34. Therefore, it was decided to score points to the vertices contained in the 16 smallest clusters. Since the number is 16 the vertice that hit the latest one will get 6.25 points; which falls in the second from the end - 12.5 and so on up to 100. As a result of summation of the values on the vertices we get the rating of the suspicious vertices of the community graph for the exchange of ether.

5 Conclusion

5.1 Result

As a result of this work the open API Ethereum platform was used to download data on all transactions for the week. A community graph on ether exchange was built for the week. Further, the analysis of the obtained graph, the separation of the characteristics of vertices, clustering of vertices according to the selected characteristics and the derivation of associative rules were carried out. The result of the work is the algorithm for constructing the rating of the suspicious vertices of the community graph. Since the algorithm generates a rating based on deviations from the standard values inside the similar wallets groups, the algorithm finally reveals the most suspicious wallets that differ from the general background. These wallets most likely have large transaction volumes on the balances or passing through them.

After summing up all the points by the nodes the rating of the suspicious vertices for the exchange of the Ether was obtained. Below are the top of 5 values of this rating:

Table 4. Top 5 suspicious vertices

Value	Node number
499.66345040230755	311861
488.3862740302283	633953
481.4078191332377	16010
457.7035133901014	314045
437.19194102533953	1627

As you can see, there are vertices with a fairly large number of points at the top of the rating, given that the maximum is 700 points. This indicates that the same vertices received points of 4 or more rules. Hence, it is stands out of

the general background immediately by several characteristics, which proves its strangeness.

References

1. Christian Cachin, AD Caro, Pedro Moreno-Sanchez, Björn Tackmann, and Marko Vukolic. The transaction graph for modeling blockchain semantics. Technical report, Cryptology ePrint Archive, Report 2017/1070, 2017.
2. Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):15, 2009.
3. Aric Hagberg, Pieter Swart, and Daniel S Chult. Exploring network structure, dynamics, and function using networkx. Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2008.
4. Jan Lansky. Possible state approaches to cryptocurrencies. *Journal of Systems Integration*, 9(1):19–31, 2018.
5. Wenke Lee and Dong Xiang. Information-theoretic measures for anomaly detection. In *Security and Privacy, 2001. S&P 2001. Proceedings. 2001 IEEE Symposium on*, pages 130–143. IEEE, 2001.
6. Mary McGlohon, Leman Akoglu, and Christos Faloutsos. Statistical properties of social networks. In *Social network data analytics*, pages 17–42. Springer, 2011.
7. Caleb C Noble and Diane J Cook. Graph-based anomaly detection. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 631–636. ACM, 2003.
8. Animesh Patcha and Jung-Min Park. An overview of anomaly detection techniques: Existing solutions and latest technological trends. *Computer networks*, 51(12):3448–3470, 2007.