EXTENDED ABSTRACT: REVERSE ENGINEERING THE BLOCKCHAIN AS ILLUSTRATED USING EIGEN DECOMPOSITION

This work has been presented at Workshop on Blockchain Horizons, ECIS, Portsmouth, UK June 25th 2018

Ghassemi Toosi, Farshad, Lero, University of Limerick, Limerick, Ireland, farshad.toosi@lero.ie

Buckley, Jim, University of Limerick, Limerick, Ireland, jim.buckley@ul.ie

Sai, Ashish Rajendra, Lero, University of Limerick, Limerick, Ireland, ashish.sai@lero.ie

Le Gear, Andrew, Horizon Globex Ltd., Limerick, Ireland, andrew.legear@horizon-globex.ie

Context

Software reverse engineering is the generation of abstracted views of large software systems from detailed implementation artifacts. As such, reverse engineers work towards views obfuscated by the scale, and complexity of those artifacts. In a similar vein, software reverse engineering, provides a wealth of techniques that are of potential use if applied to the Blockchain. Some work has already been undertaken (Badev and Chen, 2015), (Kondor et al., 2014), (Dorit and Adi, 2013), and the work of our group builds on this existing research by investigating and leveraging approaches from the field of software reverse engineering, towards aggregated, insightful views of Blockchain transactions.

Objective

To demonstrate the potential, we chose one such technique - Eigen Decomposition (Arenas, Diaz-Guilera, and Pérez-Vicente, 2006) - and apply it to an extract of the transaction records of the Ethereum Blockchain. The raw Blockchain only shows relationships between wallets and individual contracts and gives no abstract view aggregating wallets - by organisation, for example. We show how sensible groupings of wallets and contracts, otherwise opaque to a viewer of the Ethereum Blockchain, can be derived. We see future, potential applications in policing tax, fraud detection or, more generally, revealing intra and inter company views of Blockchain transactions.

Approach

Our implementation of Eigen Decomposition, is a binary matrix (P), $N \times M$ where N is the total number of items (in this case wallets) and M is the total number of attributes (in this case contracts). The entries of such a matrix represent the association of each item to each attribute. For example, an entry 1 in the matrix indicates the use of a specific contract, indicated by the column index, by a specific wallet, as indicated by the row index.
(Arenas, Diaz-Guilera, and Pérez-Vicente, 2006) shows the use of Eigen Decomposition of a square matrix in order to find the number of disconnected groups of nodes in a network. They have shown that the number of zero Eigen values of a similarity matrix of N items is equal to the number of clusters of those N items. We make use of the same technique and convert the aforementioned matrix P into a similarity matrix in order to find a number of groups of wallets according to their use of similar contracts. Since P is not a square matrix and also it is not a similarity matrix, the transpose of the matrix P has to be calculated - we name the transposed matrix P^T. The multiplication of $P \times P^T$ results in a square matrix that indicates the similarity between each pair of wallets based on the number of contracts used by them. The list of Eigen values of such a matrix is calculated and the number of independent clusters of wallets is obtained. In this manner the groupings of wallets can be changed based on a given threshold. For example, any entries less than 3 could be converted to 0, meaning that these pairs of wallets (that have less than 3 contracts in common) should be considered as not-similar wallets.

Illustration

To illustrate this approach, a 300 block snapshot (approximately 1 hour given average Ethereum confirmation times) from block 5159642 to 5159942 (Feb-26-2018 12:48:13 PM +UTC until 02:06:23 PM +UTC) was isolated and

analysed[1]. Applying an Eigen Decomposition analysis with a threshold of 4 common contracts produces 2 distinct multi-wallet groups of size 35 and 3 respectively. For simplicity of illustration, we examine the smaller, second group here. The wallets `0xbbcb6e7102561c351658564ae0cfff34464b0796`, `0x21c83b0245a2418de1eb6e1f40e1b042bfc66562`, and `0x1ee32250e83515642aa10726df00a9b365e1b379`, all used the same four contracts within that time slot:

1. `0x041fe8df8b4aaa868941eb877952f17babe57da5`: "NYB Coin" (NYBC); an ERC20 Contract[2].
2. `0x1530df3e1c69501d4ecb7e58eb045b90de158873`: "Bitcoin EOS" (BITE); an ERC20 Contract with a proof-of-burn[3] implementation to convert to an external currency.
3. `0x01995786f1435743c42b7f2276c496a610b58612`: "San Dian Zhong" (SDZ); an ERC20 Contract.
4. `0x45555629aabfea138ead1c1e5f2ac3cce2add830`: "CandyHCoin" (CANDY); an ERC20 Contract.

At first glance the correlation between wallet and contract seems likely: It could be that these are very popular tokens and that there are therefore many unrelated trading accounts chasing that equity. But this only holds to scrutiny if this was an intersection of sets and not an exact correlation. However, further manual investigation into the transaction history of these accounts reveals that they have the exact same transaction histories, interacting with the same contracts for the same amount. This cannot be the result of coincidence and we must conclude that these accounts are tightly coupled, probably owned by the same individual or company. While this subsequent analysis was manually performed in this instance, there is no reason while the analysis could not be automated going forward.

Conclusion

Our initial example shows how Eigen Decomposition analysis can reveal a number of tightly coupled clusters of wallets otherwise obfuscated from the basic blockchain graph. We would like to expand our analysis to a larger data set and define clear metrics when determining that clusters of wallets are indeed interrelated in order to test our hypothesis. Some of these metrics are hinted at in the previous section - e.g. correlations of transaction numbers and target contract signatures.

Another inherent property of note is that Eigen Decomposition analysis reveals disconnected sub-graphs of the blockchain. We would like to investigate how this property of the analysis could be used to potentially implement advanced sharding strategies (Luu et al., 2016).

Finally, we have demonstrated the potential value of software reverse engineering techniques when applied to blockchain data mining and analysis. Given the wealth of existent reverse engineering and analysis techniques available in software engineering (Eisenbarth, Koschke, and Simon, 2003), (Kitchenham, 2010) we plan to do a broad trawl of these techniques and the data contained in the blockchain, to identify potential analysis techniques and analysis targets. We then plan to match these targets to the appropriate techniques and evaluate the efficacy of those techniques in this novel context.

References

Arenas, A., A. Diaz-Guilera, and C. J. Pérez-Vicente (2006). "Synchronization reveals topological scales in complex networks." *Physical review letters* 96 (11), 114102.

Badev, A. and M. Chen (2015). *Bitcoin: Technical background and data analysis*. Technical Report 2014-104. Federal Reserve System.

Dorit, R. and S. Adi (2013). "Quantitative Analysis of the Full Bitcoin Transaction Graph." In: *Financial Cryptography and Data Security: FC 2013. Lecture Notes in Computer Science, vol 7859. Springer, Berlin, Heidelberg*. Lecture Notes in Computer Science 7859. Springer.

Eisenbarth, T., R. Koschke, and D. Simon (2003). "Locating Features in Source Code." *IEEE Transactions of Software Engineering* 29 (3), 210–224.

Kitchenham, B. (2010). "Whats up with software metrics? - A preliminary mapping study." *Journal of Systems and Software* 83 (1), 37–51.

Kondor, D., M. Posfai, I. Csabai, and G. Vattay (2014). "Do the Rich Get Richer? An Empirical Analysis of the Bitcoin Transaction Network." *PLoS One* 9 (2).

Luu, L., V. Narayanan, C. Zheng, K. Baweja, S. Gilbert, and P. Saxena (2016). "A secure sharding protocol for open blockchains." In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, pp. 17–30.

[1] The date holds no special significance - It is arbitrary and simply the moment we decided to perform the analysis.

[2] A contract implementing a standard interface for the issuance and exchange of custom tokens, representing shares in a company

[3] Currency transferred to this contract is irretrievable and is used by other platforms as proof of purchase for their crypto-currencies.