Report Number:

2020-05-04

Centrality of Blockchain

Zixuan Li

Decentralization is widely recognized as the property and one of most important advantage of blockchain over legacy systems. However, decentralization is often discussed on the consensus layer and recent research shows the trend of centralization on several subsystem of blockchain. In this project, we measured centralization of Bitcoin and Ethereum on source code, development eco-system, and network node levels. We found that the programming language of project is highly centralized, code clone is very common inside Bitcoin and Ethereum community, and developer contribution distribution is highly centralized. We further discuss how could these centralizations lead to security issues in... **Read complete abstract on page 2.**

Centrality of Blockchain

Zixuan Li

Complete Abstract:

Decentralization is widely recognized as the property and one of most important advantage of blockchain over legacy systems. However, decentralization is often discussed on the consensus layer and recent research shows the trend of centralization on several subsystem of blockchain. In this project, we measured centralization of Bitcoin and Ethereum on source code, development eco-system, and network node levels. We found that the programming language of project is highly centralized, code clone is very common inside Bitcoin and Ethereum community, and developer contribution distribution is highly centralized. We further discuss how could these centralizations lead to security issues in blockchain. Our work can also provide some empirical background for future security analysis on blockchain systems.

Centrality of Blockchain

Zixuan Li
Computer Science & Engineering
McKelvey School of Engineering
Washington University in St. Louis
li.z@wustl.edu

Abstract—Decentralization is widely recognized as the property and one of most important advantage of blockchain over legacy systems. However, decentralization is often discussed on the consensus layer and recent research shows the trend of centralization on several subsystem of blockchain. In this project, we measured centralization of Bitcoin and Ethereum on source code, development eco-system, and network node levels. We found that the programming language of project is highly centralized, code clone is very common inside Bitcoin and Ethereum community, and developer contribution distribution is highly centralized. We further discuss how could these centralizations lead to security issues in blockchain. Our work can also provide some empirical background for future security analysis on blockchain systems.

Index Terms—Blockchain, Ethereum, Bitcoin, Security, Decentralization

I. Introduction

Recent years have witnessed the uprising of cryptocurrencies with a market capitalization of $220 B [1] against traditional currency. Comparing to the traditional currency system, one of the most significant advantage of cryptocurrencies like Bitcoin and Ethereum is that they are running on top of blockchain, which is widely considered as a decentralized system. Much of the current research focus on only the consensus layer to ensure the incentive structure is set up to enable decentralization with correctness guarantee. However, as with all other computing systems, decentralized system relies on several layers of computing to function, from hardware to software, from individual node to network, from incentization to applications based on consensus. In this project, we conducted a measure and analysis on some of the layer. To be more specific, we measure the (de)centralization of source code, (de)centralization of developer, (de)centralization of network node. We found the code contribution and language distribution is highly centralized behind the blockchain eco-system and large percentage of blockchain related project contains similar code snippets. These findings indicate that heavy centralization at various crucial layers of a distributed system which would significantly impact the security of system eventually.

II. Centralization on Source Code

A. Methodology

To understand how centralized the source code is within the whole blockchain eco-system, we first download 4606 repositories marked as in Bitcoin topic and 7362 repositories marked as in Ethereum topic from world biggest VCS GitHub. we filter out repositories which are least important to Bitcoin and Ethereum whose star number and fork number on GitHub is less than 5. After that, by manually go through all repositories' description and README file, we remove some repositories which are not relevant to Bitcoin and Ethereum and add some repositories which are actually important to bitcoin and Ethereum eco-system despite they have few stars and forks. Then we have our repositories dataset as 1539 Bitcoin related repositories and 2319 Ethereum related repositories. We measure centralization of source code from two important project properties: programming language and code similarity. To determine the programming language of project, we use Linguist developed by GitHub to identify the most used language within a single repository and mark it as the language of project. To quantify the similarity between two repositories marked as using same language, we first use our similarity comparison tool to find similar code block between two repositories. Then we use Jaccard Similarity Coefficient to calculate the similarity score.

$$Jaccard(A, B) = \frac{\|A \cap B\|}{\|A \cup B\|} \tag{1}$$

$$Sim(A, B) = \frac{LOC(Sim(A)) + LOC(Sim(B))}{LOC(A) + LOC(B)} \tag{2}$$

The similarity score between two repositories which marked as different language is assigned to zero. Since similarity comparison through all files in all repositories would be very time-consuming, we only make the similarity comparison on the top 100 forked repositories for both Ethereum and Bitcoin repositories as they are good representative of highquality project.

B. Results

1) Centrality of Project Language: From dataset mentioned above, we plot the language distribution of Bitcoin and Ethereum repositories separately. The Figure 1 and 2 only shows the top 20 languages in the purpose of better visualization. As we can observe from them, language of Bitcoin related projects is centralized in JavaScript and Python, and language of Ethereum related projects is most centralized in JavaScript. Comparing to the language distribution of 2019 [2], we can clearly find out

that in the language distribution of the whole software development's curve is more flatten , in other words, more decentralized than our decentralized blockchain system. As one of the most clearly phenomenon shows in the figure 3, blockchain related project heavily relies on JavaScript than general software. This give attacker the possibility of taking control the majority of blockchain by only targeting JavaScript.

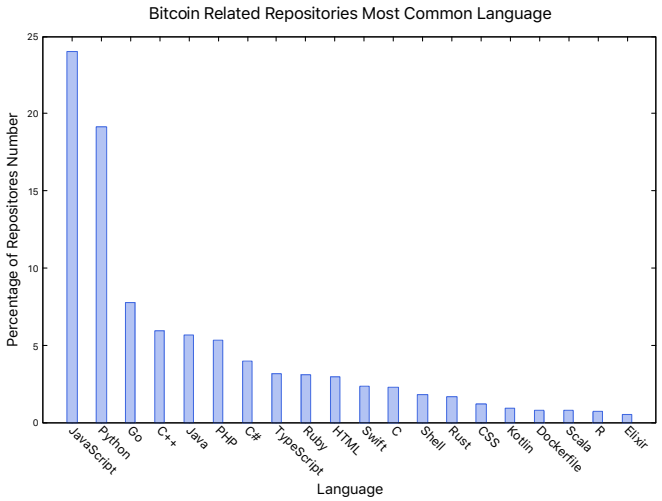

Fig. 1: Bitcoin Related Project Language Distribution

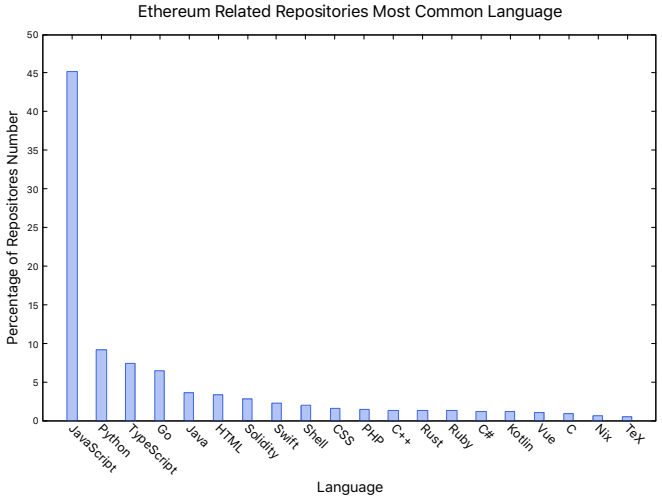

Fig. 2: Ethereum Related Project Language Distribution

2) Similarity of Project Source Code: In both Figure 4 and Figure 5, we graph the similarity of project source code. Each node represent a repository and edge between two nodes indicates two repositories contains similar code blocks. Thickness of edge is determined by the similarity score: if one edge is thicker than another means these two repositories are more similar than other. The node size is determined by the node degree. So the bigger node means this have more similarity connection than other nodes. All connected nodes are in the same color

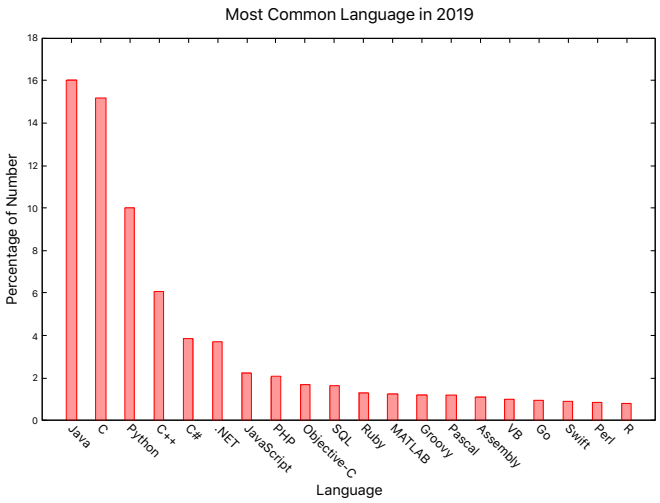

Fig. 3: Language Distribution of 2019

for better visualization. The uncolored nodes which are also discrete nodes means no code clone founded in this repositories. In Figure 4, we have 47 out of 100 repositories contain similar code blocks. In Figure 5, we have 56 out

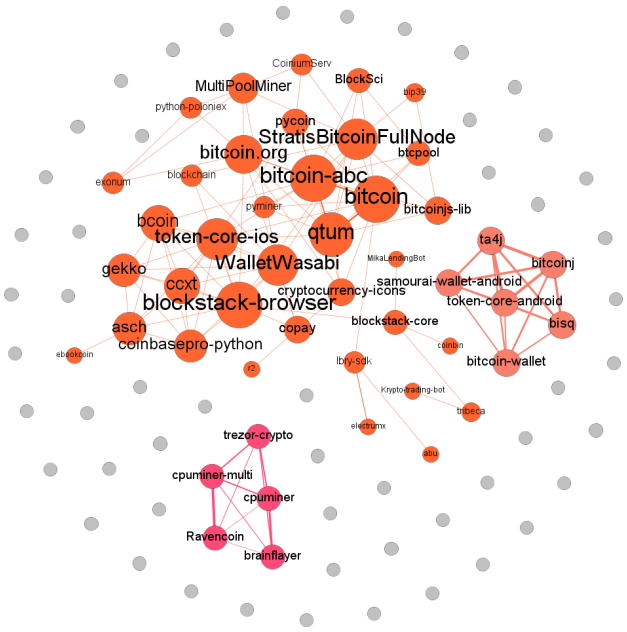

Fig. 4: Bitcoin Related Project Similarity Graph

of 100 repositories contain similar code blocks. Further comparing across Figure 4 and Figure 5, we can find overall similarity of Ethereum is higher not only because they have more colored node but also because they have more connected nodes. We can see a lot of dark blue nodes at the center of this picture. Although, we did not make similarity comparsion on all repositories due to the time restriction, the code clone situation of remaining

repositories are imaginable worse than these high-quality project. So the conclusion that blockchain related project faces problem of code clone is still solid.

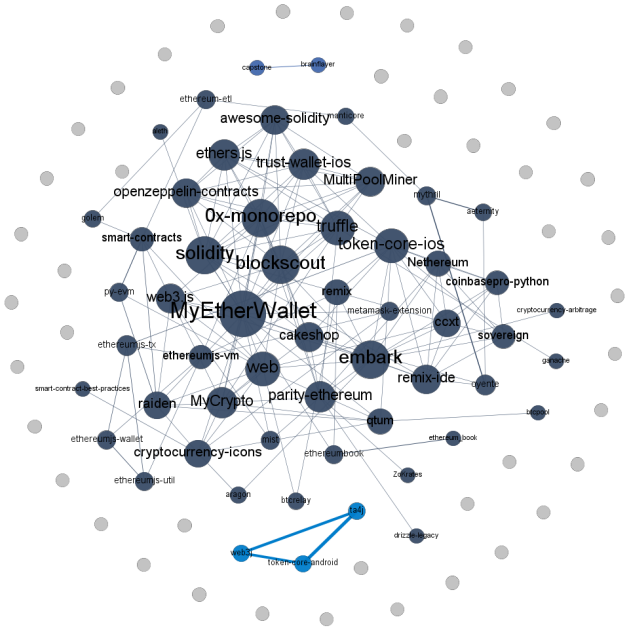

Fig. 5: Ethereum Related Project Similarity Graph

III. Centralization on Development Eco-System

A. Methodology

We use the repositories dataset above and build a contribution dataset by mapping each commit to a specific contributor. The detail of mapping is discussed in the Difficulties part. To calculate the contribution made by developer within single repositories we use Equation 3 proposed by Xu [3].

$$Contri(i) = \frac{LOC(i)}{\sum_{k=1}^{n} LOC(k)} + \frac{Commit(i)}{\sum_{k=1}^{n} Commit(k)} \quad (3)$$

$$Contri(i,j) = \frac{LOC(i,j)}{\sum_{k=1}^{n} LOC(k)} + \frac{Commit(i,j)}{\sum_{k=1}^{n} Commit(k,j)} \quad (4)$$

$$W\_Contri(i) = \sum_{j=1}^{p} \frac{Contri(i,j) \times Forks(j)}{\sum_{0<j\leq m, 0<j\leq n} Contri(j,k) \times Forks(k)} \quad (5)$$

This equation takes both number of lines of code have been modified and numbers of commits into account is considered to be more precise compared to choosing only one of them.

Since more fork number of a repository means more project developed on top of it, the fork number can be a good indicator of how much contribution a repository made to whole development eco-system. Therefore, we use another fork number weighted version of equation 1 as Equation 4 and 5 to calculate contribution to the whole development eco-system made by developer. Contri(i, j) means developer i's contribution in j repository. p is the total number of developer i have contributed.

Then we analyze the centralization on development Eco-System from two perspective: centrality of developer contribution in single project and centrality of developer contribution across projects. For centrality in single project we choose the most the popular repository "bitcoin/bitcoin" and "ethereum/go-ethereum" from each Bitcoin community and Ethereum community. "bitcoin/bitcoin" is not only the most popular repository which have 43.1k stars and 25.6k forks on GitHub, but also one of most important repositories among Bitcoin development eco-system, it develops the official code of bitcoin core component. It is the source of bitcoin system and source of decentralized blockchain system.

As for "ethereum/go-ethereum" which have 25.8k stars and 9.4k forks on GitHub is the official implementation of the Ethereum protocol, it contains multiple small project including official Ethereum client geth, core Ethereum protocol and EVM. Those projects together built the Ethereum and made Ethereum as so called blockchain 2.0. The result of these two projects is a good representation of other repositories.

For centrality across projects, we analyze two datasets. One of them is the original contribution dataset made by all repositories inside the repositories dataset for both Bitcoin and Ethereum, another is a contribution dataset made by top 100 forked repositories inside the repositories dataset for both bitcoin and Ethereum. The necessity of latter one is that there may be a lot of developer who only have slightly interest into blockchain. They only commit one time and never come back again. Those developer could give us false impression of centrality by increasing large number of contributors with small amount of contributions. To mitigate this issue, we need the latter dataset as the contributor of the top 100 forked repositories can be referred as developers who have continuous interest in blockchain.

B. Results

1) Centrality of Developer Contribution in Single Project: The development contribution of these two projects is not decentralized as the blockchain system or at least it claims to be as we can see from Figure 6 and Figure 7[1]. To be more precisely, for "bitcoin/bitcoin", GitHub user "lannwj" alone made 34.25% of total contribution. And top 50 contributors together made 83.37% of total contribution while remaining 761

[1]The contributor's ID in figure is the first 5 characters of GitHub Handle, if contributor can not associated to a GitHub account, it's ID is the the first 5 characters of Email address

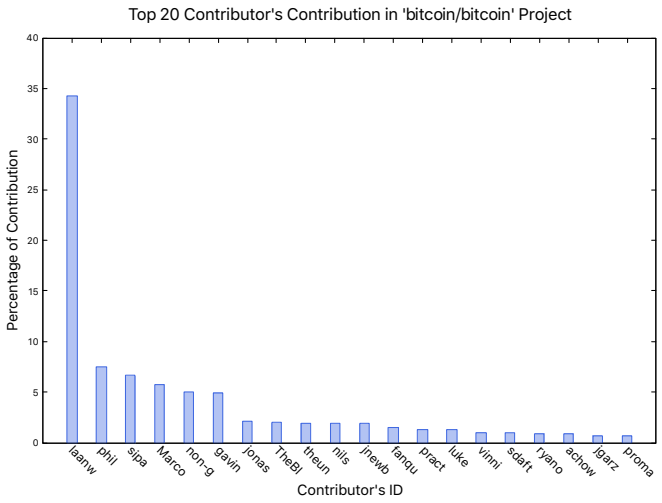

Fig. 6: "bitcoin/bitcoin" contribution distribution

developer together merely contributed 16.63%. Analogously, for "ethereum/go-ethereum", GitHub user "obscure","karalabe","fj" and "CJentzsc" each made 35.375% , 14.49%, 12.395%, 11.005% of total contribution. And top 50 contributors together made 96.09% of total contribution while remaining 428 developer together merely contributed 3.91%.

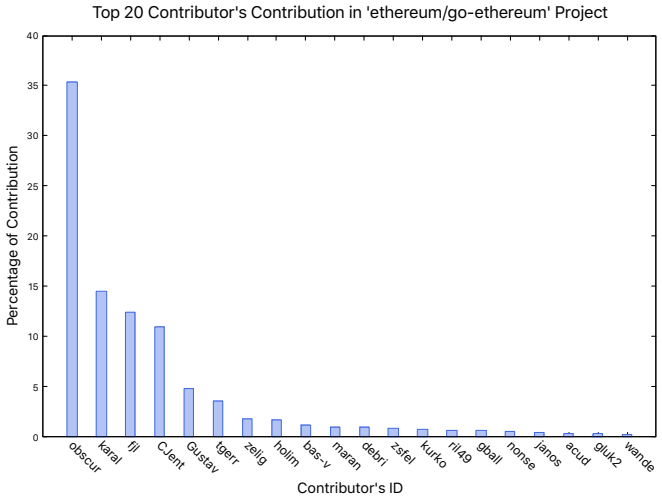

Fig. 7: "ethereum/go-ethereum" contribution distribution

2) Centrality of Developer Contribution Across Project: To better visualize how centralized those distribution is, we use cumulative percentage of contributors as x axis and cumulative percentage of contribution as y axis. We also add an idealistic decentralized distribution curve as a reference. We can clearly observe the inequality of contribution distribution from both Figure 8 and Figure 9. It is obvious that the most contribution are made by only small number of developers in both development ecosystem. Since the more one curve fit to the idealistic decentralized distribution curve, the more decentralized this curve is. We can use the area between real world curve and decentralized distribution curve as a indicator of level of centrality. Based on that we can observe from Figure 8 and 9, the contribution distribution is more centralized if it is weighted and even it is not weighted it is still highly centralized. Weighted top 100 result is slightly less centralized than the across all result as we predict in methodology part, but their all still highly centralized comparing to the idealistic decentralized distribution curve. One interesting finding is that the Bitcoin development eco-system is more centralized than Ethereum's, if we compare across these two figures.

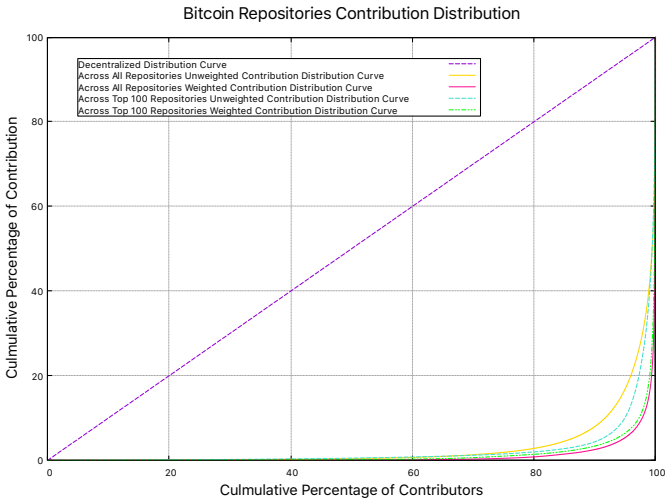

Fig. 8: Bitcoin Related Project Contribution Distribution

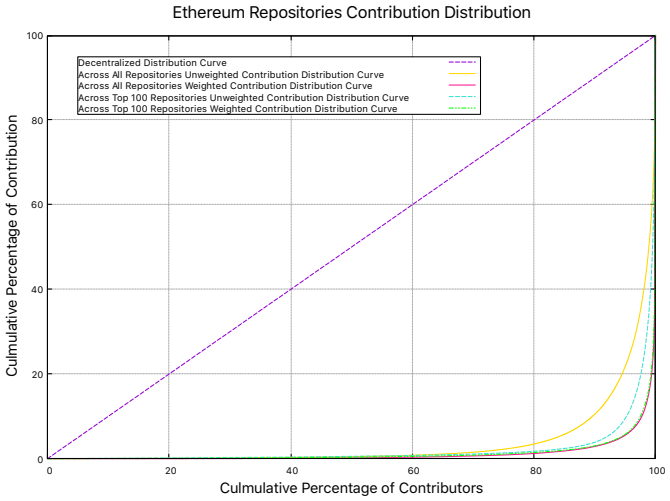

Fig. 9: Ethereum Related Project Contribution Distribution

IV. Centrality on Network Node

In the aspect of network node, previous work have successfully measure bandwidth latency, peer-to-peer latency [4], node geographic location etc. In this project

we measure its centrality from node security perspective. We want to find out whether there exist common security vulnerabilities across Ethereum nodes's host machine. We collected 11927 nodes with 15156 IPs from ethnode.org website. We tried use shodan port scan data to find common vulnerability but shodan only contain information about 2898 nodes with 2657 IPs. And the data from shodan shows 627 vulnerable IP in total and 400 of them are ssh username enumeration, 167 of them are remote code execution, 36 of them are denial of service and 13 of them are directory traversal. We believe ssh username enumeration is not a severe security vulnerability and the shodan dataset are too small compared to the number of node we collected. The result cannot provide useful insight.

V. Difficulties

A. Collecting repositories from GitHub

To collect repositories dataset mentioned above, we need to download all projects whose topic marked as Bitcoin or Ethereum. GitHub API [5] provides simple method to request the search by constructing a query with topic qualifier, however, the returned result is always not completed. After carefully examining GitHub API documentation, we found GitHub API will only return the first 1000 matched results per query. To resolve this issue, we construct our query carefully and limit the number of matched results. There are few qualifiers we can used in the query such as repository size, number of forks, number of stars, when a repository was created or last updated. However, considering we are interacting with an active system, the repository size, number of forks, number of stars or last updated date can change dynamically, the only static qualifier we can use is the date of creation. In order to make sure each the number of repositories within a specific date interval does exceed 1000, we use binary search to build date interval correctly and quickly. By using this approach, we successfully collected 4606 repositories with topic marked as Bitcoin and 7362 repositories [2] with topic marked as Ethereum.

B. Mapping commits to contributor

After we clone all repositories into local hard drive using method described above. We are trying to map the commits to each unique contributor. According to the Git specification [6], Git allow user to change their name and email at will which means, it is highly possible a single person using multiple alias and how to merging those alias is a big problem. Fortunately, GitHub assigns a unique ID for each user associated with an GitHub account. To retrieve author's ID of one commit, we only need to request the detail of commit based on commit SHA through GitHub API. However, if we request for all commits from GitHub API, it will be very time consuming since GitHub have restriction on request times over a period of time we will lose our time on waiting. To reduce the total request time, we build an email-to-id lookup table. We first query GitHub API to get contributors list and contributor's public email for each repository. Based on GitHub API documentation, if user choose to keep their address private, the commit email will show as $'id+username@users.no-reply.github.com'$ or $'username@users.no-reply.github.com'$. So, we also construct those two email address and mapped them to GitHub ID. Then we use Git log command to get the commit details and matching its email against the email-to-id lookup table. If author's email cannot be found in the lookup table, we then query the API and get the information for this specific commit. Through this approach, we successfully match 26336 email address from total 30223 email address to 21085 GitHub account in 10 hours. The unmatched 4950 address partially cause by GitHub user changing username with using GitHub provided no-reply email address at same time [7]. After we done the picking mentioned in Centralization on Source Code's methodology, we have 3858 repositories in total and have 21898 email address associated with 18204 GitHub account.

[2]Repositories data is collected in Dec 2019

VI. Security Impact

Why is centralization bad? The intention of building a decentralized system is to make sure the fairness of each node. All node should be treated exactly as the consensus claims and not a single node can impact the system through a method that is not mentioned or acknowledged in the consensus. However, centralization gives some nodes or entities the possibility to influence other nodes or system against the consensus's will. Therefore, the consensus is broken and system is in jeopardy.

A. Impact of centralization in source code

If attack want to attack a decentralized system programmed in all kinds of language, they will need the power to find different vulnerabilities so they can attack 51% of the system to make it work. In contrast, the centrality of programming language could give attacker the advantage to impact large scale of blockchain network only exploits the vulnerability inside the JavaScript or Python. The level of difficulties of finding vulnerabilities inside one or two language is way lower than finding vulnerabilities for multiple languages. The code clone situation in the similarity of project source code make it even worse as attacker may only need to target some widely used libraries. The Parity Wallet Hack incident in 2017 is a great real-life example. It is cause by a simple bug written inside a library that Parity multi-sig contract called. Anyone used the Parity wallet or wallet which also used this library was affected. Eventually this hack allowed an attacker to steal over 150,000 ETH [8].

B. Impact of centralization in developer contribution

Another centralization we found is the centrality in development team. Speaking of security, human is always a key factor since they always make mistake deliberately or accidently. This is exactly why the DAO(Decentralized Autonomous Organization) merges. Its goal is to codify the rules and decision making process of an organization, eliminating the need for documents and people in governing, creating a structure with decentralized control. The DAO once raised over $150m from more than 11,000 enthusiastic members [9]. However they put too much attention on the desiging a system which can not be interfered by human factor but they neglect the developer who actually implemented system. A simple reentrancy bug [10] written by the development team allowed attacker to stole 3.6 million ether. What make things more interesting is the development who wrote this bug not only known this bug but also made a patch few days before the DAO attack. They even wrote a blog about how they patch this bug and claimed, "The important takeaway from this is: as there is no ether whatsoever in the DAO's rewards account — this is NOT an issue that is putting any DAO funds at risk today." [11] As the developer's incapability of writing correct and secure code is the direct reason of this attack, the negligence of importance of developers is the ultimate reason. Code written by a small group of developer increase the risk in security since it will magnifier both developer's stupidity and cleverness. As they may be too stupid to write secure code or clever enough for colluding on stealing other's assest. Furthermore, developer tends to reuse code or coding logic from one project their wrote to another project they are working on. This could result in more severe code clone situation and endanger the system in a directly way.

VII. Limitation

Although we found some centralization on some level of the decentralized blockchain system, there are still some limitation on our work. In our work of measuring the centrality of project language, we use the primary language as the language of project and neglect other language occurs in the project. And such negligence could increase the centrality of language distribution. Except that, in the measurement of centrality of code contribution, we use lines of code in the equation of calculating contribution. However, each programming has its own characteristic and the importance of 1000 lines of C++ code is not the same as 1000 lines of Python code. This could increase or decrease the centrality of code contribution.

VIII. Conclusions

In this project, we measure the centrality of blockchain from centralization on source code, centralization on development eco-system and centralization on network node. We found that the programming language of Bitcoin is centralized in JavaScript and Python while the programming language of Ethereum is centralized in JavaScript. We measured the code similarity across top 100 forked repositories in both Bitcoin and Ethereum and found 47% of Bitcoin and 56% Ethereum project have code clone problem. For the Bitcoin and Ethereum development team, we found most contribution are made by the only a small group of developer. We tried to find common security vulnerability on the Ethereum network node, but data sample is too small to make insightful conclusion. We successfully found centralization on some layer of the blockchain system. Furthermore, our work can provide some empirical background for future security analysis on blockchain systems.

Acknowledgment

This is a sub-project of a joint research conducted with Griffn Shaw from Washington University in St. Louis and Yang Xiao from Virginia Polytechnic Institute and State University. Thanks to the guidance and advice from Professor Ning Zhang and Yang Xiao. This paper is written during the COVID-19 pandemic. Special thanks to all those who keep us safe and healthy.

References

[1] Cryptocurrency market. [Online]. Available: https://www.tradingview.com/markets/cryptocurrencies/global-charts/

[2] B. Putano. (2019) A look at 5 of the most popular programming languages of 2019. [Online]. Available: https://stackify.com/popular-programming-languages-2018/

[3] X. Ben, S. Beijun, and Y. Weicheng, "Mining developer contribution in open source software using visualization techniques," in 2013 Third International Conference on Intelligent System Design and Engineering Applications, 2013, pp. 934–937.

[4] A. E. Gencer, S. Basu, I. Eyal, R. van Renesse, and E. G. Sirer, "Decentralization in bitcoin and ethereum networks," 2018.

[5] Github api v3 | github developer guide. [Online]. Available: https://developer.github.com/v3/

[6] Git - documentation. [Online]. Available: https://git-scm.com/doc

[7] Setting your commit email address - github help. [Online]. Available: https://help.github.com/en/github/setting-up-and-managing-your-github-user-account/setting-your-commit-email-address

[8] S. Palladino. (2017) The parity wallet hack explained. [Online]. Available: https://blog.openzeppelin.com/on-the-parity-wallet-multisig-hack-405a8c12e8f7/

[9] D. Siegel. (2016) Understanding the dao attack. [Online]. Available: https://www.coindesk.com/understanding-dao-hack-journalists

[10] P. Daian. (2016) Analysis of the dao exploit. [Online]. Available: https://hackingdistributed.com/2016/06/18/analysis-of-the-dao-exploit/

[11] S. Tual. (2016) No dao funds at risk following the ethereum smart contract 'recursive call' bug discovery. [Online]. Available: https://blog.slock.it/no-dao-funds-at-risk-following-the-ethereum-smart-contract-recursive-call-bug-discovery-29f482d348b