A time series analysis of Gitcoin grants contributors to GR15

Context:

This research has been conducted as a contribution to the first Gitcoin Open Data Science Hackathon project: Sybil Slayer Bounties. Using data provided by Gitcoin the goal was to identify Sybil attackers.

Introduction

Gitcoin mission is to build a more open, collaborative and economically empowering world. By giving tools to communities, they help build and funds public goods and bring the future digital public infrastructure.

Gitcoin funds projects by using a quadratic funding mechanism. This mechanism helps to reduce plutocracy where the power of large donators because one dollar equals one vote and at the same time aims to balance the ideocracy principle of one person one vote.
This principle of quadratics funding still has a biased towards projects having more contributors. Thus, in the context of Gitcoin grants, Sybil attackers are people creating multiple addresses to fund a Gitcoin grant and then cheat the quadratic funding mechanism by using many votes by many contributors. Universally Sybil attacks are described as the abuse of a digital network by creating many illegitimate virtual personas to incentives an outcome.
The fraud report of grant 15 highlights that out of 40000 contributors almost 23% were Sybil donors. And the value of these donations is over $200k. Detecting and removing these Sybil attackers has a cost to Gitcoin in terms of human resources as some accounts must be reviewed manually by the team in order to determine if a contributor is Sybil or not.

To tackle that issue, Gitcoin already developed Fraud Detection & Defense (FDD) mechanism. It is a combination of two mechanisms to tackle Sybil's behavior. The first one is proactive: Passport. It is a decentralized identity verified on chain, the user can acquire stamps proving its uniqueness and humanness by registering accounts from web2 as Google, Twitter... and web3, Lens protocol to a decentralized identity. These credentials create a trust bonus for people collecting multiple stamps from various authenticators. The second method is retroactive and looks at the on-chain history of the contributor. It is called Sybil Account Detection (SAD). The combination of Passport and SAD allows to creation of a trust score that will help decide if the contribution is matched or not. The balance is critical as the on-chain history of a new user that wants to contribute to a friend's project may be low and thus could be squelched by the mechanism identifying him as a Sybil contributor.

The Gitcoin team identified 4 main types of contributors:

- The native, with a lot of on-chain history and that, want to contribute to public goods.
- The noob, with few on-chain histories, the noob could also be a secondary account to fund a project from

- The farmer, whose motive is to contribute by expecting to be rewarded later, should not count in the quadratic funding.
- The Sybil, his motive is to manipulate the QF mechanism, and he may create a transaction history to have a non-zero trust score as long as the cost of forgery is lower than the expected reward.

The goal of the analysis is to identify and find distinctive patterns to classify contributors to one of those categories.

Data provided by Gitcoin

The most useful data for the Sybil slayer challenge was the hackathon-contributions-dataset. The data gives a list of all the contributions made to the grants 15 collected. It gives relevant information about the address, transaction Id, user, grant Id , token used, amounts contributed, timestamp and the chain used by the contributor to participate in the grant. Each line of the data represents one contribution.

Data mining

The based idea of that research was to look at the on-chain history and maybe find a new method to identify Sybil's accounts. The contributor's wallet address was used, and the transaction history of those addresses was retrieved using EthersScan API and PolygonScan API. Note that only normal transactions were retrieved which is not the full picture of the activities of a wallet. The data set is publicly available at (https://huggingface.co/datasets/Poupou/Gitcoin-ODS-Hackhaton-GR15).

We restricted the study to the Ethereum and Polygon contributors, the same idea could be applied to the Zksync chain, and we suppose the results would be similar.

Feature extraction

The main idea was to see if the on-chain history of a wallet has a time series. All these time series are not 100% comparable as they do not have the same length (number of transactions). Yet, we considered that they were and extracted features from the transactions by analyzing each column of the data as a signal of the time series for an address. We extracted time series features from the timestamp, value, gas, and gas Price of each address. This has been done mostly using the python package tsfresh (https://tsfresh.readthedocs.io/en/latest/), some features were added manually for the analysis. The extracted features are also available at (https://huggingface.co/datasets/Poupou/Gitcoin-ODS-Hackhaton-GR15) or could be created by using scripts from the GitHub repo.
Each dataset contains 164 features. The Ethereum dataset contains 19027 addresses, and the Polygon dataset contains 8227 addresses. Because no transactions were found for some users, the total number of contributors to GR15 is higher than the number of addresses in the dataset.

These features were cleaned by removing the duplicated columns and removing the columns with a standard deviation equal to 0. Then the data was standardized using a Standard scaler from sci-kit learn. Standardizing was mandatory as they were large order of magnitude differences between features.

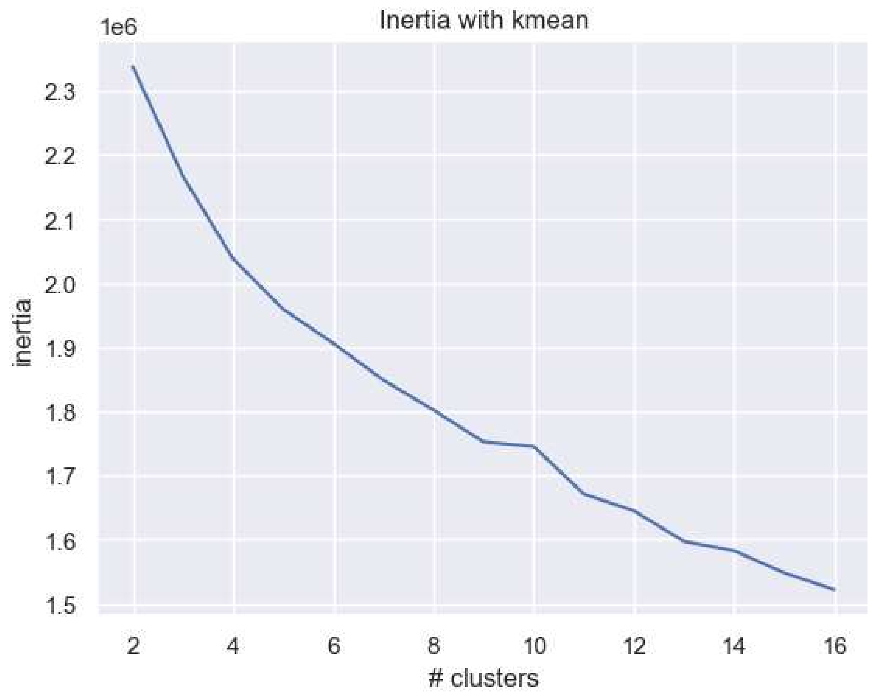
Unsupervised learning

No ground truth was provided by Gitcoin team; thus, it was an unsupervised learning task to detect Sybil patterns.

We tried a few unsupervised techniques including K-means, agglomerative clustering, DBSCAN and spectral clustering. The spectral clustering was not converging. DBSCAN was tested with various parameters configuration but was not leading to relevant clusters.
K-means clustering, and agglomerative clustering were leading to similar results. In the next section, we will dive into the method and results obtained with K-means clustering.

To represents the clusters obtained with K-means we used t-SNE. T-SNE convert affinities of data point to probabilities and allow to reveal of structures in a lower dimensional space. Thus, visualizing the cluster in a 2D space.

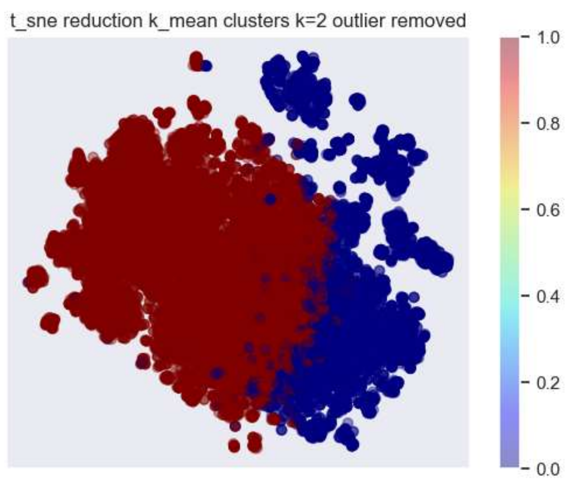
We started with Ethereum data.
We had an expected number of clusters to obtain that is 4 for the four categories: Sybil attacker, Noob, Native and farmer. Nonetheless, we ran K-mean on many clusters to try to find the ideal number of clusters. As we see in the following picture, the elbow method did not work well as they were no elbows in the chart, the inertia keeps decreasing like a logarithmic curve without showing any elbow as the number of clusters increased.

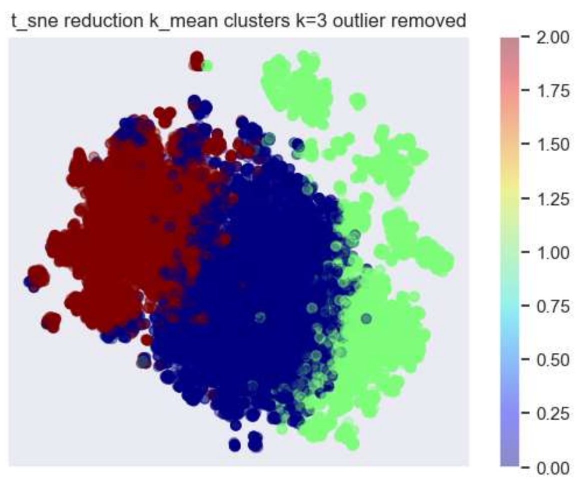


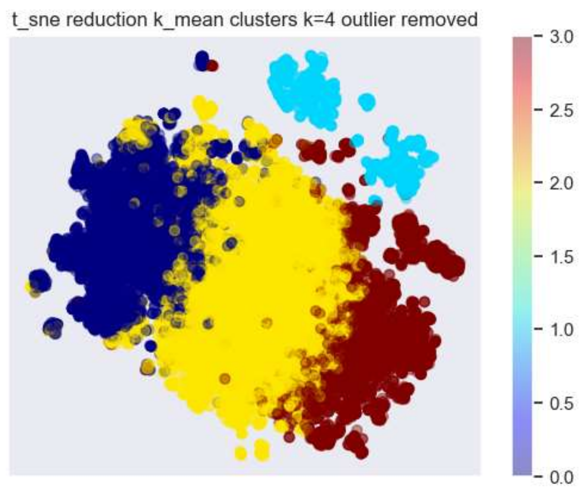
We remarked that by increasing the number of clusters some clusters were starting to be very small. As we see in the following silhouette score for k=14. We noted that these small clusters were outliers and decided to remove them and perform a new K-mean classification. Thus, we ran a first-time K-means with k=15 and took all clusters smaller than 1% of the total size of the data and removed these points from the data (occurrences highlighted in green in the table below).

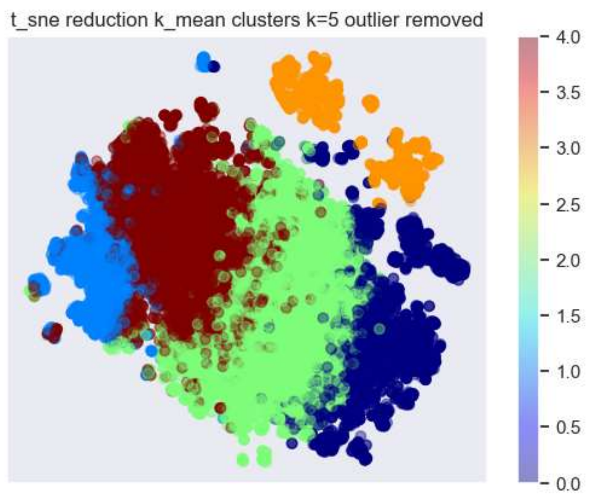
k	occurences	percentage
	3948	20.749461
14	3806	20.003153
3	2568	13.496610
1	2536	13.328428
2	1878	9.870184
13	1378	7.242340
11	803	4.220318
7	700	3.678982
5	607	3.190203
4	525	2.759237
9	172	0.903979
0	102	0.536080
8	2	0.010511
12	1	0.005256
6	1	0.005256

Next, we ran K-mean between 2 and 8 clusters to analyze the cluster obtained. Below are the results on a t-SNE representation of our feature space.



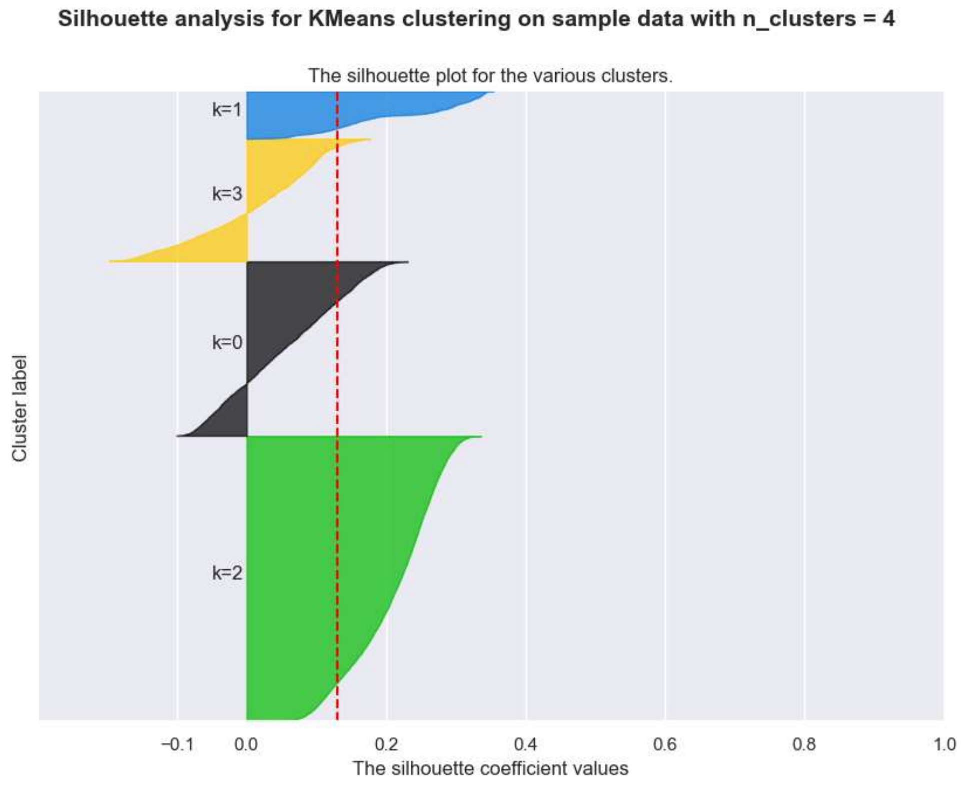






Some patterns are clearly remarquable.

Below is the silhouette score obtained for k=4



The inertia is not very high but still defines some boundaries between clusters.

Cluster occurrences is as follow for k=4:

k	Occurrences	Percentage
2	8500	45.335751
0	5193	27.697477
3	3644	19.435703
1	1412	7.531068

By looking at the on-chain history we can try to map a behavior see in the charts above with a type of user. Below is a sample of 4 random address of each cluster for each cluster.

Output exceeds the size limit. Open the full output data in a text editor

k=4, cluster = 2

```
['0x0d6a9963f7b362c3d50cfdfdc9207b4d8e766c13',
    '0x5f8b1d3143a042d6c2116cd0ea1db0503c176885',
    '0x0f3f7c8d5c7f93b223ba2cb0cfd9514fba15ed96',
    '0x0cc0fdfa2a9edfc89087e405a4d4b6120ea686fb'],
```

Mostly ENS registered and with on-chain history but low balance, could be a sophisticated Sybil attacker.

k=4, cluster = 0

```
['0xad2b2f8044986cb8f0148a03fe14c5551272386f',
    '0xb9c29b3ef0cf74b6591fce755e79616c3e77f024',
    '0xbdbc9751d73f29cfddb285eba08ffdb7b3395dfe',
    '0x8afeb17641ce3bfe786a6c4e1bf000cd8a268c96'],
```

ENS registered large on-chain history and higher balance amount; probably a native

k=4, cluster = 3

```
['0xad6fd97ab003c66a1121c6a8801ff917cd799f61',
    '0xcdfeb3b0a870764d0ef4e882038fc8367380ccc6',
    '0xc8edec5d005ebb18f7ba3f816fd1d1c606c10dd6',
    '0x20e3f2155fe9bfb22413882b52783c289be4937e'],
```

Low balance and few transactions history

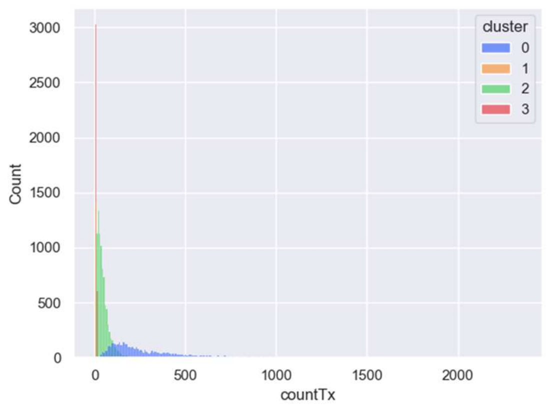
k=4, cluster = 1

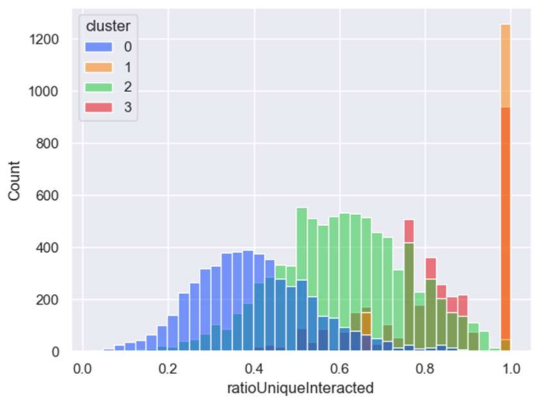
```
(['0x90773f946de726a7be61d00f6c5542241a9441cb',
    '0xa70d36cd8ab6454ff736bd66b3089b3a693ea045',
    '0x67ba333d253d12b2cda615382d4e09a4f79f932b',
    '0xd01db03984661e8a01122235a55b31a577325aa0']
```

Very few transactions, the only activity was the donation, probably a farmer may be a noob or not sophisticated Sybil attacker.

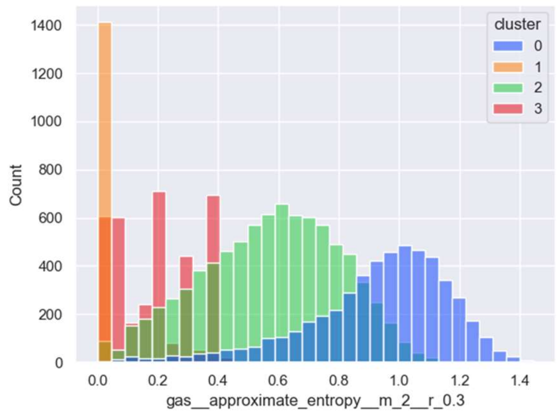
The most interesting part was seeing these clusters on a histogram of the features. Here we show some of the most interesting patterns:

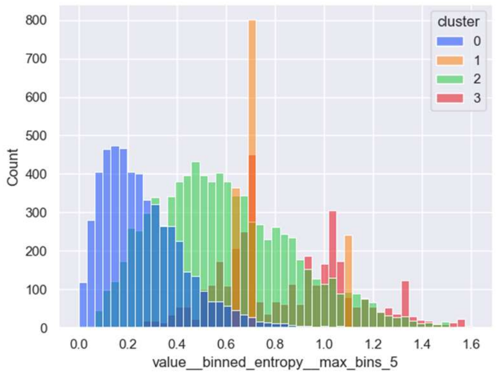
Please mind that the colors are not the same in the scatterplot and the histogram. Also, the table above is ordered by the number of occurrences and not by the name of the cluster.

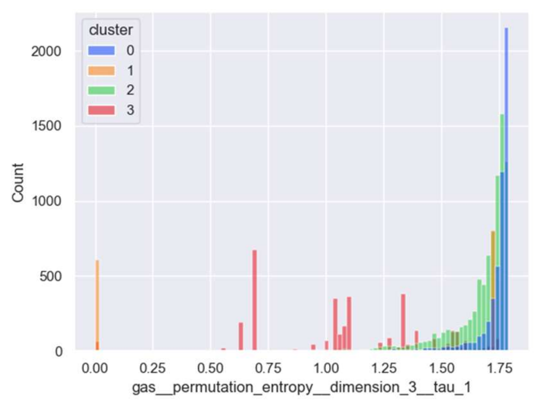


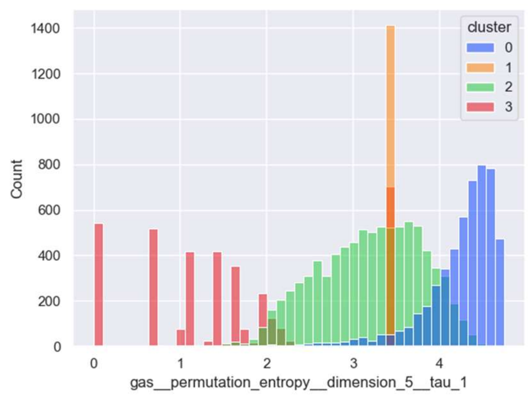


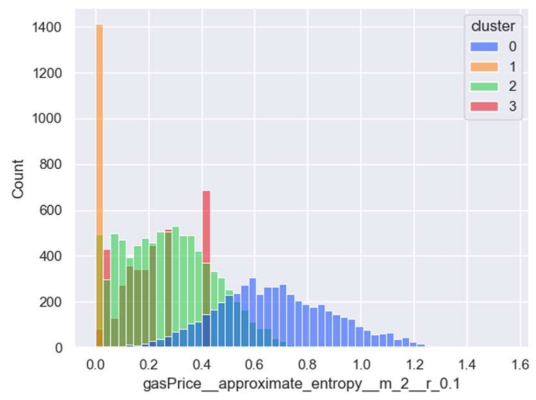
We can find the patterns we discovered manually above, cluster 1 has almost 0 transactions, and cluster 0 is probably the native as the ratio of unique addresses interacted with is lower.

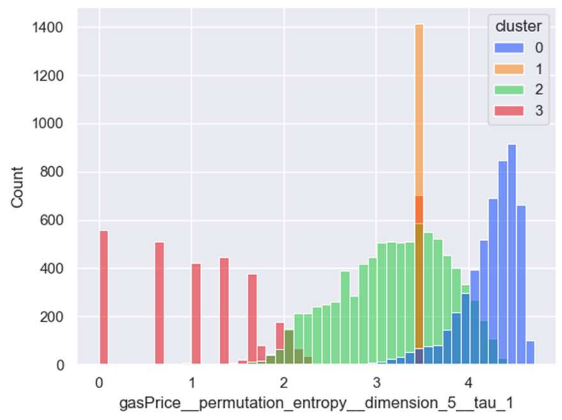




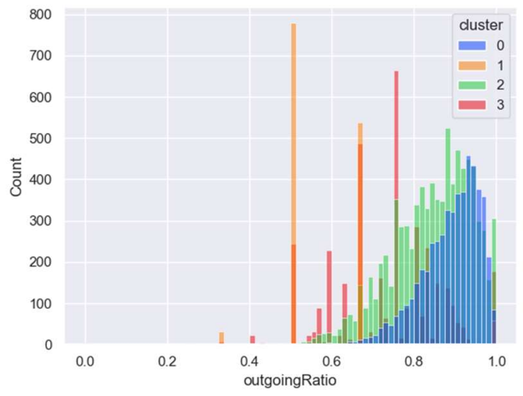




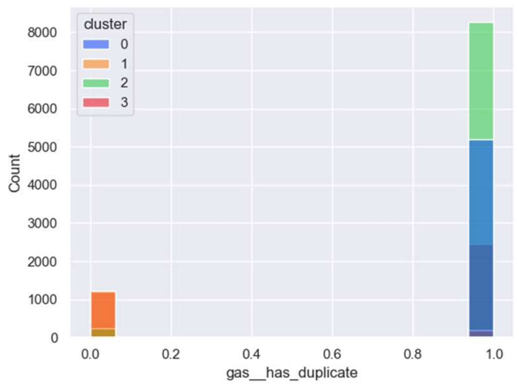


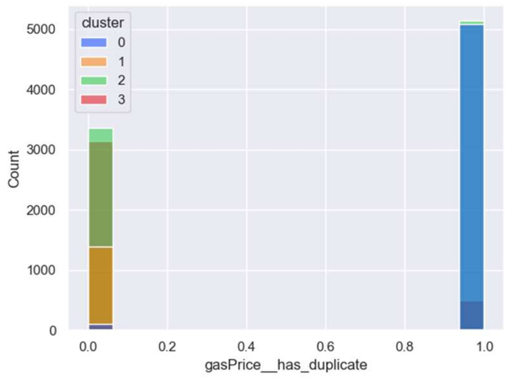


From the previous figures, we see that entropy brings a beautiful discriminant to identify groups of users. We find the same patterns with all the metrics used to compute the entropy. The parameters of the entropy computation are important as they allow to discriminate of the clusters.

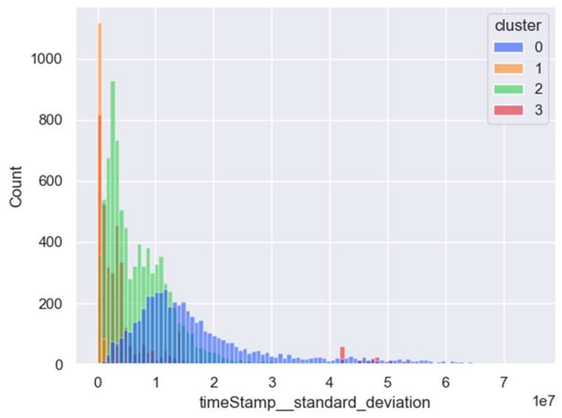


The outgoing ratio shows that heavy user has higher values of the ratio.





The has_duplicate lead to surprising results, we would have expected that attackers are using bots thus the gas or gas price may have duplicates, but we are observing the contrary on both gas and gas prices. Though the gas price is more balanced. And most native users have duplicates. It could be explained by exchanges using fixed gas prices, this would mean that if an address has no duplicates, it has less than 2 incoming transactions from an exchange or it is an attacker using another address to fund that account. Nonetheless, the attacker will probably use the same amount of gas/gas price for sending all its transactions. Here incoming and outgoing transactions are mixed, in future work, we should look at duplicates both in incoming and outgoing transactions.



The time stamp standard deviation provides insights, attackers may have a high standard deviation (in red) because they are reusing the same addresses. Or they are using new ones in orange or green.

With the analysis performed we could identify the groups as follow:

k	Occurrences	Percentage	User
2	8500	45.335751	Native or Sophisticated Sybil
0	5193	27.697477	Native
3	3644	19.435703	Sybil / farmer
1	1412	7.531068	Sybil / noob

It is a difficult task to label with certitude as there is no ground truth.

The results are not exposed here for the Polygon chain, but they were very similar to the ones with the Ethereum chain. They can be reproduced by running the jupyter notebooks available in the repo.

Conclusion

The unsupervised task of identifying Sybil's attacker and farmers is hard. Most of the behaviour may overlap between users. Only the native group was identified with “accuracy”. The user with only 2 or 3 transactions is also very easily identifiable, but we may never know if it is a new user or an attacker. Thus, the simpler way may be to prevent him from getting his funding match if he is not verified by the passport stamps with a high trust score.

The labelling done by the Gitcoin team for GR15 may bring more insights into explaining and verifying the clustering found by studying the normal transactions of the contributor.

Future works may include the addition of ERC transactions into the data set as well as the number and amount of previous contributions.