Applied Network Science

RESEARCH Open Access

Analyzing hack subnetworks in the bitcoin transaction graph

Daniel Goldsmith*, Kim Grauer and Yonah Shmalo

*Correspondence:
daniel.goldsmith@chainalysis.com
Chainalysis, New York, USA

Abstract
Hacks are one of the most damaging types of cryptocurrency related crime, accounting for billions of dollars in stolen funds since 2009. Professional investigators at Chainalysis have traced these stolen funds from the initial breach on an exchange to off-ramps, i.e. services where criminals are able to convert the stolen funds into fiat or other cryptocurrencies. We analyzed six hack subnetworks of bitcoin transactions known to belong to two prominent hacking groups. We analyze each hack according to eight network features, both static and temporal, and successfully classify each hack to its respective hacking group through our newly proposed method. We find that the static features, such as node balance, in degree, and out degree are not as useful in classifying the hacks into hacking groups as temporal features related to how quickly the criminals cash out. We validate our operating hypothesis that the key distinction between the two hacking groups is the acceleration with which the funds exit through terminal nodes in the subnetworks.

Keywords: Cybercrime, Network analysis, Complex networks, Hacks, Crytocurrency, Bitcoin, Cybersecurity, Temporal networks, Sociotechnical systems

Introduction

The Bitcoin network is a distributed, public ledger, secured through blockchain technology. All transactions occur between two distinct public addresses and are permanently recorded on the specific blockchain built for bitcoin. The process of securing these transactions is handled by bitcoin miners, who use their computing power to solve complex cryptographic problems and in the process verify blocks and transactions (Nakomoto 2009).

Anyone can create a bitcoin address to receive funds through a variety of software projects such as Blockchain.info (BLOCKCHAIN LUXEMBOURG S.A 2011) or Electrum wallets (Electrum 2011). Additionally, there is no limit to the number of bitcoin addresses that any individual or organization can make. There are also no requirements for verifying your identity in the process of address creation. It is completely free to make an address, however, it costs money to transfer money on the network by paying transaction fees.

Because of the ease of transactions between pseudonymous addresses, cryptocurrencies, and bitcoin in particular have been especially attractive to criminals who both exploit technological vulnerabilities and prefer to move funds through the pseudonymous bitcoin transaction network to avoid detection by law enforcement (Huang and et al. 2018). Indeed, the amount of cybercrime involving cryptocurrencies has grown via ransomware (Huang and et al. 2018), scamming activity, phishing scams, and hacking of exchanges or wallets (Chainalysis 2019). There have been several attempts to quantify the scale of criminal activity as well (Yin and et al. 2017).

Notably, exchange hacks are one of the most costly types of cryptocurrency related crime. Hackers have stolen $1.7 billion dollars worth of cryptocurrency from exchanges since 2011 (Chainalysis 2019). Tracing stolen funds in order to freeze the assets of the perpetrators is one of the most effective ways of safeguarding against future attacks, as this method removes bad actors from the ecosystem and disincentivizes similar activity from other actors. Typically, either government or private cyberinvestigators, take up the task of tracing stolen cryptocurrency funds. Their investigations begin with a known address that has been hacked. They then follow the funds through up to thousands of different addresses until the funds hit a service (an off-ramp), i.e. an alternative means of cashing out the stolen bitcoin. Ideally, an investigator will trace funds to a service so that a subpoena can be issued to the service to unmask the identity of the criminal. These investigations result in traced out subnetworks representing the flow of stolen bitcoin from the point of breech on an exchange through exit ramps.

We obtained six subnetworks from investigators at Chainalysis, a firm specializing in blockchain investigations. These were investigations carried out over several months, and which effectively trace all of the stolen funds through the entire bitcoin transaction graph. Each edge is a transfer of the stolen money to a node which is controlled by the hacker. The size and complexity of these graphs vary according to the amount of effort the hacker used to move funds and that hacker's level of technological sophistication.

Similar subnetworks can be collectively generated by the community of users that trace funds on the public Bitcoin ledger and often does occur after a criminal steals cryptocurrency on a public ledger (ErgoBTC 2019).

We present research to algorithmically visualize and analyze hack cash out subnetworks that capture the temporal behavior of hackers and locate the stolen funds. We then build similarity matrices based on eight graph features, run community detection over those matrices, and successfully classify certain hacks to the known hacking organization to have carried out the attack. We find that temporal features, such as the rate at which the hackers send funds to exit ramps, are the most effective features to use for grouping specific hacks together and classifying them to their hacking groups.

We find that this method might prove useful as a component of some automated classification system designed for anti-money laundering or anti-fraud detection of transaction ledgers, not only for the specific use case that we describe in the work below as specific to these investigations, Chainalysis, or even bitcoin as a whole.

Algorithmically traversing hack subnetworks and its limitations

We investigate bitcoin hacks by traversing subnetworks of nodes that have been built out by professional crime investigators. These hack subnetworks are comprised of nodes that have either directly or indirectly received hacked funds, see Fig. 1 for visualization.

We then create visualizations to identify trends in the hack and to better understand the time patterns specific to each hack as the stolen bitcoin flows to the

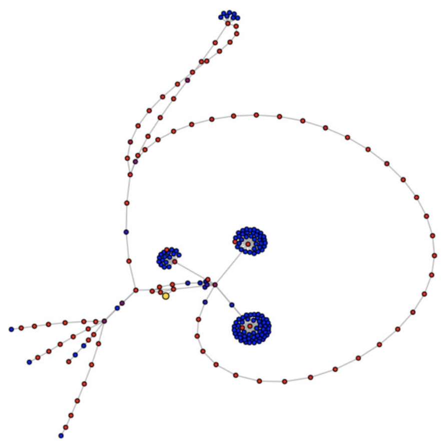
Fig. 1 Sample Hack Subnetwork, Hack A3

boundary of the networks generated, see Fig. 2. In some cases, when the level of obfuscation is minimal, investigations tracking stolen funds often terminate at services (see Methodology section on identifying services), simply because criminals want to change their stolen bitcoin for fiat currency, or at least convert it to a another cryptocurrency.

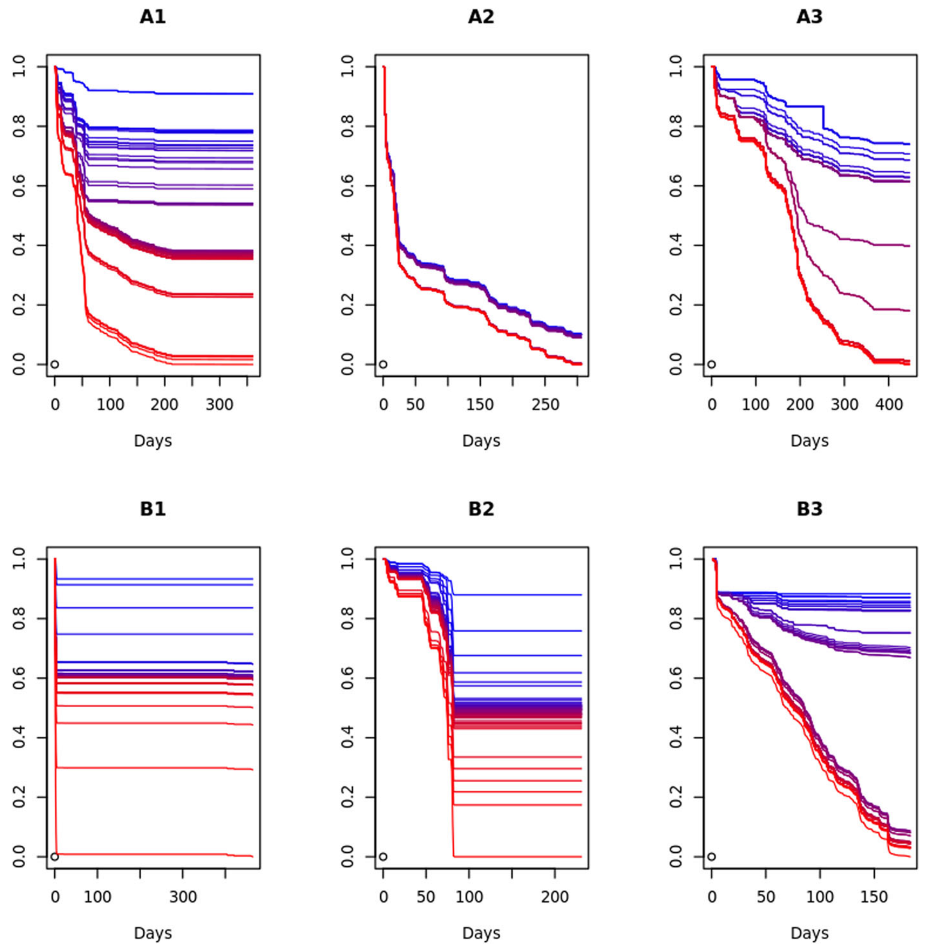

Fig. 2 Amount in Play over Time

Yet cryptocurrency investigations are usually much more complex then this (Nouh and et al. 2019). Often, the investigator may not know if a node belongs to a service, particularly in the case of a mixing service. Furthermore, stolen bitcoin from some of the largest hacks may utilize laundering mechanisms in which OTC brokers act as third party sellers allowing for a change of hands to an entity that is no longer behind the hack. This activity can not be detected through blockchain analytics unless their is a source of ground truth confirming the funds passing through on OTC broker. Without this confirmation, the funds would appear to move from one pseudonymous node to another.

Sometimes the investigations are so complex that the investigator simply cannot go through the process of tracing every single stolen bitcoin to an cash out point. In this case, the investigator may choose to chase particularly promising leads, rather then spend the time to analyze every single transaction that occurred. At any given time, stolen funds may be sitting idly in non-service clusters for extended periods of time. In practice, it is common for funds to slowly leak out of these "holding" clusters (Chainalysis 2019).

Generally, as networks are built out manually by subject matter experts, methods such as the one proposed below can help ensure that the proper classifications of these networks have been achieved.

Methodology

Pipeline

1. We first gather subnetworks of known hacks that have been built out by professional investigators.
    - Due to the sensitivity of this data and relative infrequency of hack events, the result of this process provided a small set of anonymized, curated subnetworks that trace stolen funds from the origin of the hacks to all end points of interest.
    - It is at this point that we introduce a new tool for analyzing these subnetworks for additional insights that we can eventually return to the investigators and compliance officers at exchanges.
2. We traverse these subnetworks from the starting clusters through the boundary of the subnetwork.
    - An element of complexity emerges in this analysis that requires additional attention, namely that the terminal nodes require a more rigorous definition than any cluster sitting on the outskirts of the subnetwork since many of these terminal nodes act as sinks but still slowly leak funds despite maintaining control over the majority of their hacked balance. This definition will be fleshed out in the subsection "Defining Terminal Nodes." Additionally, as seen in Table 1, the simple static network characteristics demonstrate that the data is tree like, with low average degrees (in- and out-degrees are equivalent on average) and low clustering coefficients. Yet the complexity due to the temporal nature of the subnetworks as well as the nature of these terminal nodes require additional features to be defined before information can be meaningfully extracted from the data, since it is not always the leaves of these tree-like subnetworks that play important roles, either from the temporally -

Table 1 Summary Statistics for Each Hack

Hack	Txs	Nodes	Avg In-Deg	Clust. Coeff
A1	1,981	1,257	1.02	0.001
A2	421	55	1.11	0.041
A3	607	218	1.05	0.008
B1	190	176	1.01	0.000
B2	374	335	1.06	0.002
B3	57,299	174	1.62	0.068

in that they arrive latest - or topologically - they sit farthest in the transaction graph.

3. Next, to better visualize the temporal activity in the hacks, we create two time series that display the activity of the hacked funds.
    - First, we measure how active the hackers are over time by computing the number of transfers the hackers make each day, as seen in Fig. 3.
    - Second, we measure the funds traced as they move to terminal nodes, as seen in Fig. 2. As the funds move through terminal nodes, the share of funds still

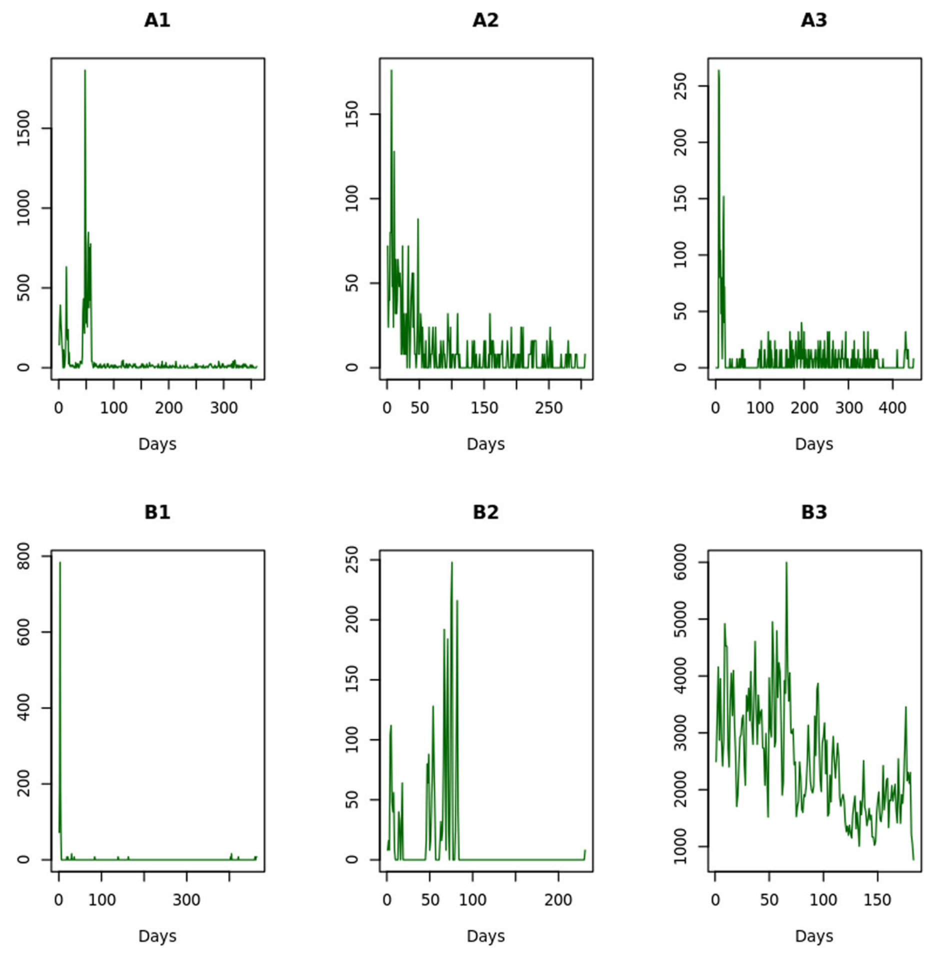

Fig. 3 Transactions over Time

held by hackers decreases. A fully tracked hack subnetwork would be visualized by the funds decreasing from 100% to 0% of funds still held by the hacker over the number of days that it takes to fully exit the funds through terminal nodes.

4. We then generate distributions for the following features for each hack subnetwork:

- Logarithm of Hack balance of all nodes, see Fig. 4.
- Weighted In-degree of all nodes, see Fig. 5.
- Weighted out-degree of all nodes, see Fig. 6.
- Average number of transactions to terminal nodes per day, across all ρ values, derived from data shown in Fig. 3.
- Terminal Nodes as a function of ρ, see Fig. 7.
- Logarithmic difference of the average percent of funds still in play, across all ρ values, derived from data shown in Fig. 2.
- Second difference of the average percent of funds still in play, across all ρ values, derived from data shown in Fig. 2.
- Logarithmic difference of the standard deviation of the percent of funds still in play, across all ρ values, derived from data shown in Fig. 8.

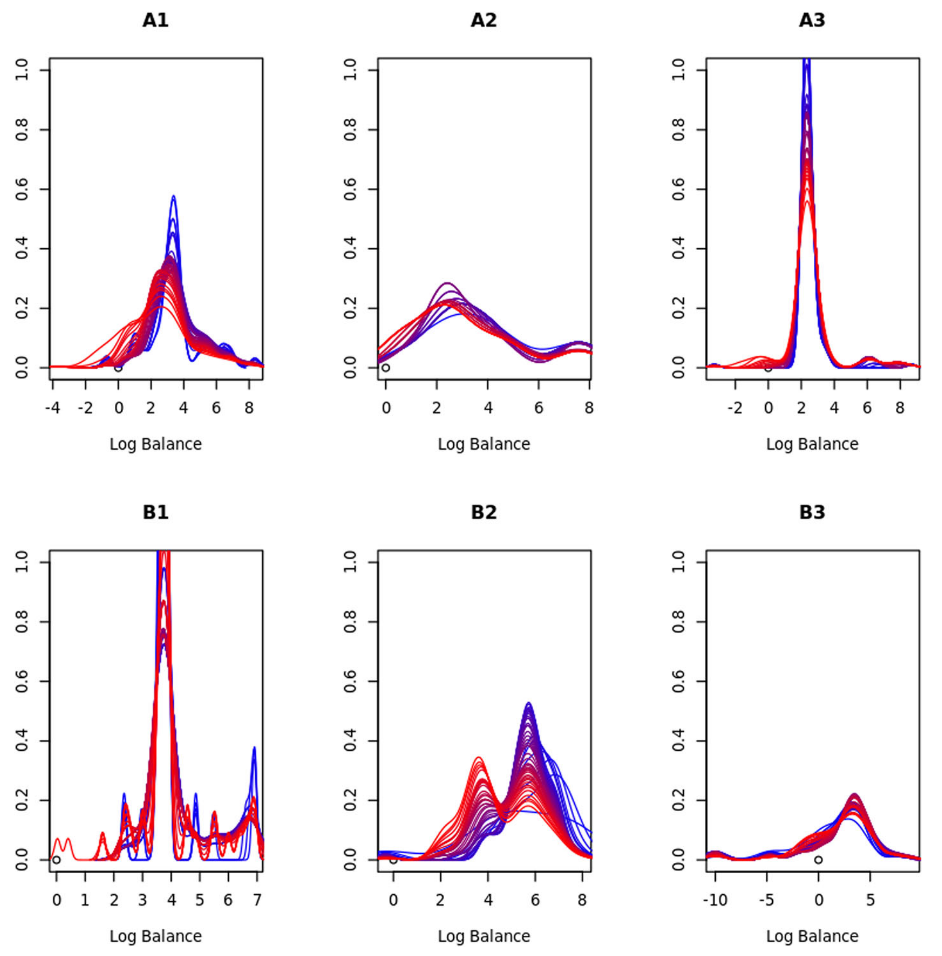

Fig. 4 Distribution of Log Balance

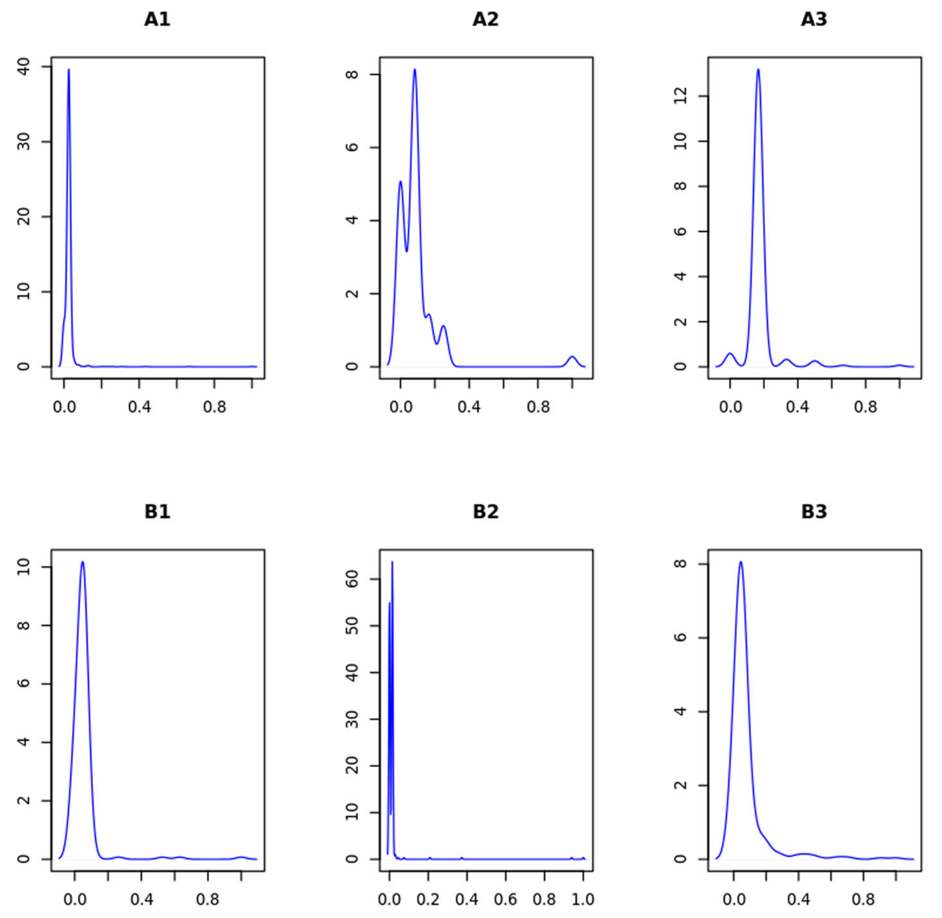

Fig. 5 Distribution of In Degree

5. Afterwards, we create similarity matrices corresponding to each distribution, whose elements are the pairwise similarities of the distributions corresponding to each of the hack subnetworks via the 1-Dimensional Wasserstein Distance, i.e. the Earthmover Distance (Villani 2003;).
6. We run two community detection algorithms, Modularity Optimization (Clauset and et al. 2004) and Walktrap (Pons and et al. 2013). We compare the output of the overall approach across the similarity matrices for all the distributions against our ground truth attribution of the two underlying hacking groups and demonstrate the potential for such a method by properly reattributing the hack networks to their respective groups. Both this step and the previous step are motivated by the idea that relational data is best analyzed using the tools of network science and the similarity of the distributions between the hacks in question fall into relational data. For a larger range of approaches utilizing complex networks for more general data clustering see (de Arruda and et al. 2012). The reason we employ purely topological distance here, rather than the exponent of a related distance as suggested in de Arruda and et al. (2012), is due to the inherent assumption that the behavior of the underlying hacking groups are similar to the point of minor

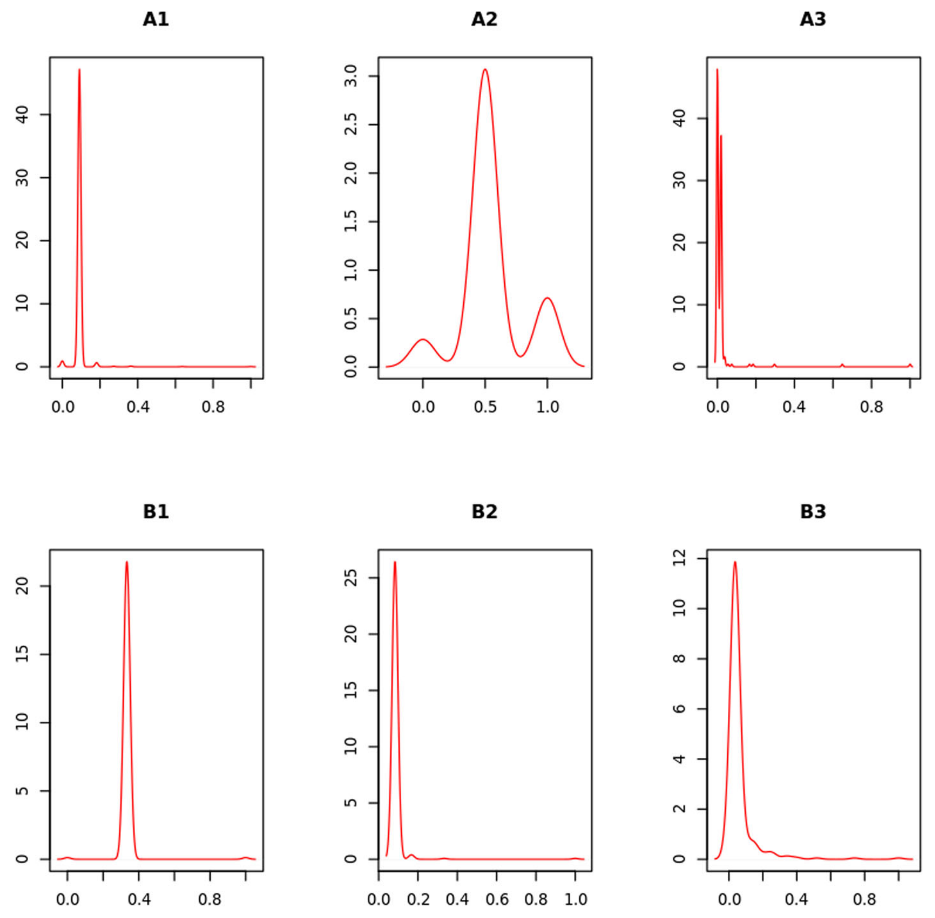

Fig. 6 Distribution of Out Degree

perturbations in the underlying distributions of activity, which we believe the Earthmover Distance is particularly well suited to detect.

7. Lastly, we review the output communities and test our hypothesis that the features relating to the hack dynamics are more informative in classifying the hacking groups than the static network features.

Identifying services

A typical service can control thousands of addresses, while larger services can even manage into the millions. We identify services by exploiting features unique to the Bitcoin blockchain. There are many different approaches that blockchains employ to cryptographically verify transactions, but the Bitcoin blockchain relies on Unspent Transaction Outputs (*UTXO's*) to record all transactions. A UTXO is the unspent output of a previous transaction that a user is entitled to transfer to another bitcoin address. Every wallet that holds a positive bitcoin balance is in possession of at least one UTXO. When multiple UTXO's are held by a single user and spent together in a transaction, it then becomes possible to definitively ascribe common ownership to all of the UTXO's that were spent together. This concept of a *cospend* is the basis of the clustering activity used

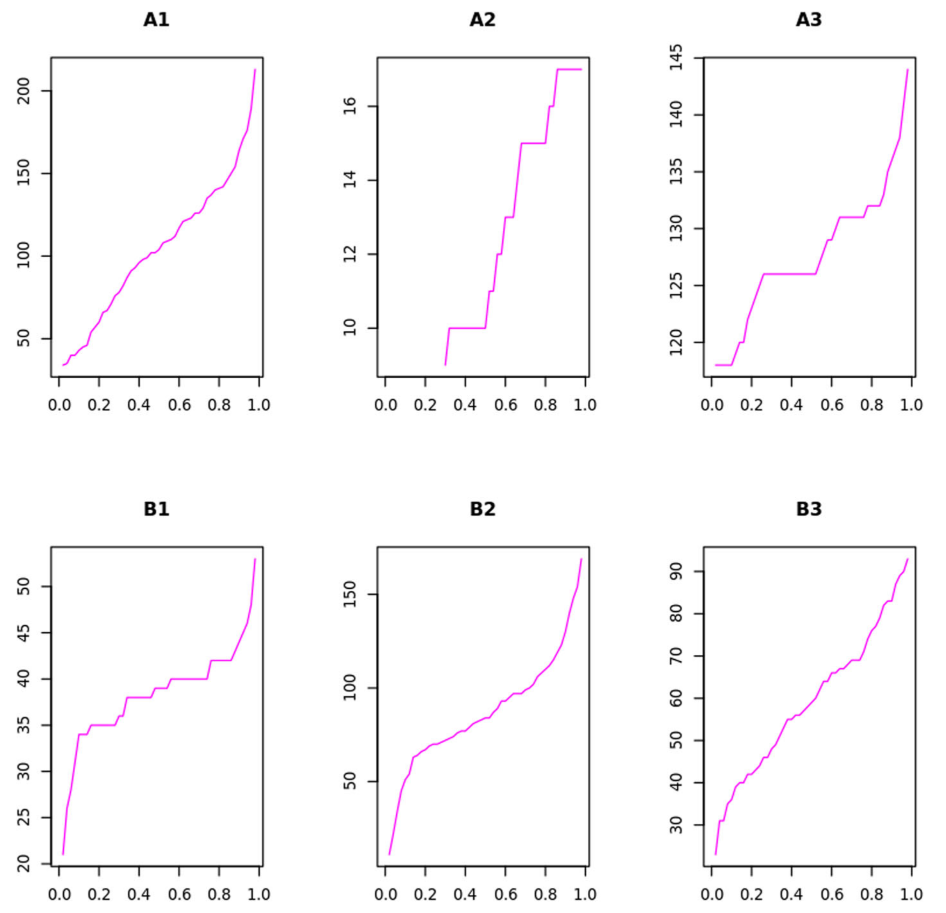

Fig. 7 Number of Terminal Nodes as function of ρ, TvR

by blockchain analysis firms such as Chainalysis to identify clusters of addresses controlled by a single entity. The network then becomes comprised of cospend clusters, i.e. nodes, composed of multiple addresses rather than long chains of single-use addresses (Meiklejohn and et al. 2013; Akcora and et al. 2019).

Once addresses have been mapped to a node through cospending activity, the node can be mapped to a named entity by interacting directly with it. For the example of an exchange, this process can occur by visiting an exchange's website, depositing funds on the exchange, and tracing that transaction via a block explorer (BLOCKCHAIN LUXEMBOURG S.A 2011). Only services with publicly available address information can be identified in this way.

When stolen funds arrive at a known service, such as a an exchange, we can assume that the hackers have attempted to cash out their funds. Professional investigators trace funds through these nodes to create hack subnetworks that capture as much of the meaningful movement of the stolen funds as possible.

Defining terminal nodes

There are two types of terminal nodes discussed in this paper. 1) A known service terminal node that is a confirmed service through the process mentioned above of pairing ground

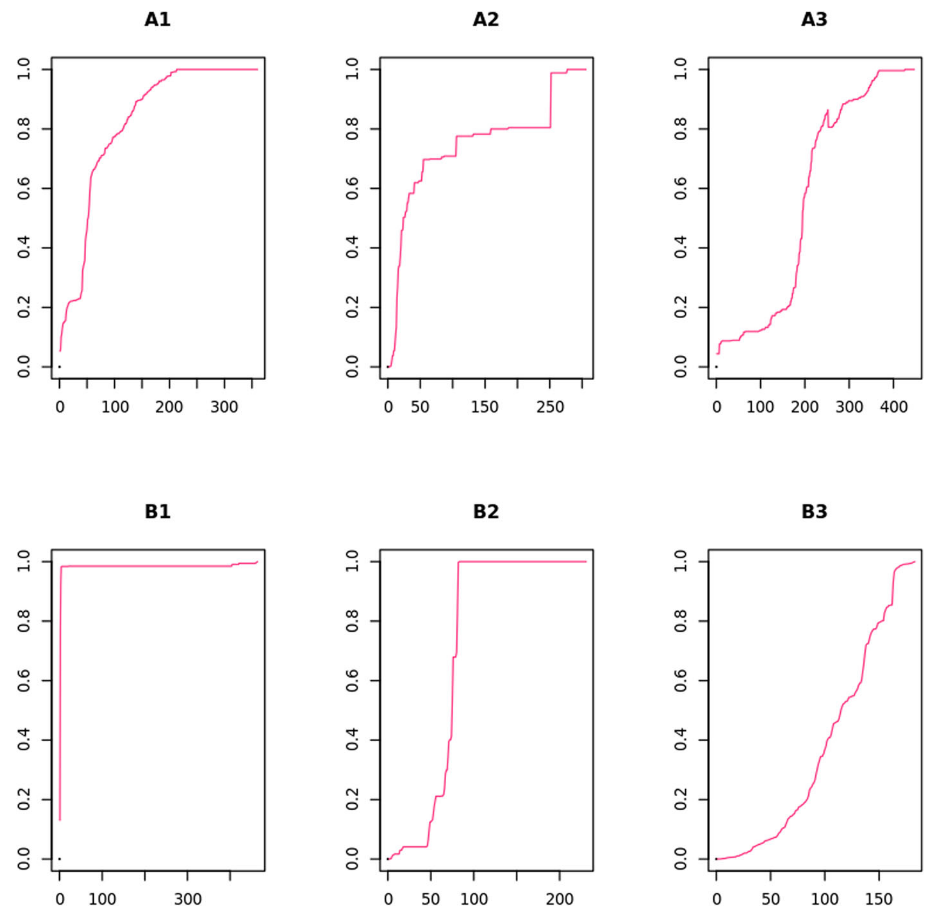

Fig. 8 Standard Deviation of Amount in Play over Time

truth knowledge with cospending activity. These services can be exchanges, mixers, gambling sites, merchant service platforms, or any exit ramp through which a criminal can off-load stolen bitcoin to an institutional cryptocurrency player. 2) An unknown service node, where the investigator has reason to believe a node is behaving like a service and will therefore terminate the investigation at that point.

One problem may arise when the investigator simply chooses to stop pursuing a lead. At this point, the boundary of their investigated subnetwork might resemble a terminal node. This limitation should be further investigated in future work. In the cases of the subnetworks chosen for this research, the investigators followed all leads, which limited the terminal nodes to those described above.

By default, terminal nodes are the edges of the graph subnetwork. Ideally, a subnetwork of a hack would track 100% of the funds from the point of a hack through all exit ramps. This would allow us to set $\rho = 0.00$, as the terminal nodes would simply be all the natural edges of the graph. In this case, the investigator would trace funds to a service, whether it be an exchange, mixing site, gambling site, etc. $\rho = 0.00$ indicates that a node has only ever received funds within the subnetwork.

We focus on the ratio rather than the difference of funds sent to received because we want to maximize the number of meaningful leads for investigators rather than raw amount due to hacked funds. By returning this normalized list of terminal nodes and resulting charts, we find all partial sinks "of interest" in the subnetwork that may facilitate the issuance of subpoenas or other leads, as well as wallets to watch because they still contain funds, large or small. As a secondary filter, we can sort by balance due to the hack, but this feature is only relevant in the operational stage for investigators, not when conducting our analysis.

We define ρ as:

$$\rho = \frac{\textit{weighted in-degree}}{\textit{weighted out-degree}},$$

i.e. the ratio for a given node of the total amount of funds it sent to the total amount of funds it received.

Others have proposed using ratios of the in/out degrees when studying the Bitcoin Transaction Graph, but in different contexts and not as a node-level feature (Bovet and et al. 2018). We introduce this ratio as a means of classifying individual nodes based on features specific to networks of financial transactions. This is particularly important when trying to capture the underlying behavior of the nodes over time, as value flows in the temporal network that they collectively compose.

Subnetworks that vary over time, such as hack investigations, generate terminal nodes throughout the duration of the network's activity. Terminal nodes with high ρ values should represent an optimal list of possible leads for an investigation, since they represent sinks of value in the transaction graph and are therefore plausibly operated by the true perpetrator of the hack or another entity of interest.

Figure 9 shows the spectrum of ρ values and their subsequent interpretation.

Visualizing temporal behavior in the hack subnetworks

The temporal visualizations are shown in Figs. 3 and 2. Figure 3 shows the number of transfers over time within the hack subnetwork so that the investigator can get a sense of how active the hackers are over time. They can answer questions such as: does the hacking group consistently make transactions over time, or do they tend to move funds according to a temporal pattern. A pattern may be indicative of an algorithm moving the funds, as opposed to actual individuals approving the transactions.

Figure 2 shows how the funds exit over time through terminal nodes. It allows an investigator to see the exiting strategy of the hacking group in time. For example, do the hackers exit the funds in one period of time, or consistently over a longer duration of time? Each of these strategies has implications for how the investigator profiles the hacking group overall. For example, a hacking group that exits all the funds through one exchange in one day

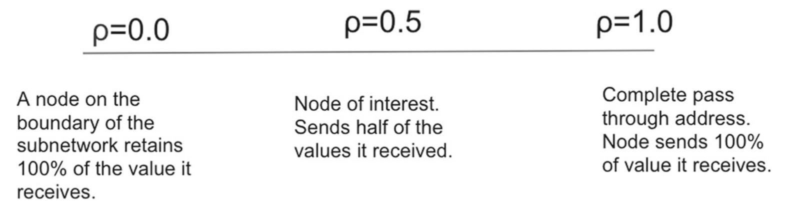
ρ=0.0
ρ=0.5
ρ=1.0
A node on the boundary of the subnetwork retains 100% of the value it receives.
Node of interest. Sends half of the values it received.
Complete pass through address. Node sends 100% of value it receives.

Fig. 9 Spectrum of rho values and their significance

may be less organized and less well-funded than a hacking group that gradually, through thousands of strategic transactions, exits the funds over a long period of time.

The trends are made visible by restructuring the hack subnetworks into time series. Figure 3 demonstrates how active the hackers are by using the number of transactions they carry out as proxies.

Figure 3 allows us to see the way the hackers utilize terminal nodes. Hacking group alpha (A1) is much more active, slowly moving funds through terminal nodes over a shorter period. Hacking group beta (B1) utilizes fewer transactions in general, but tends to send all of their transfers to terminal nodes in a short period of time. In the case of chart B1 in Fig. 3, the hackers sat on their funds for a long period of time before abruptly exiting over 70% of the funds through a few exit ramps within a one week period.

To test the hypothesis that the hackers are best classified using temporal features such as the rate at which funds cash out at terminal nodes, we vary ρ in the following sensitivity analysis section to observe stolen bitcoin exiting through terminal nodes under a range of conditions.

Sensitivity analysis of ρ

We allowed ρ to range from 0.02 to 0.98 to test the implications of gradually change the ρ parameter. A cluster with a very low ρ value, e.g. $\rho = 0.1$, would have to hold on to more 90% of the funds it received to be considered a terminal node. On the other hand, a very high ρ value, e.g. $\rho = 0.9$, allows a cluster to retain only 10% of the funds it received from the hack in order for it to be considered a terminal node. A higher ρ will capture many more terminal nodes, as it is an easier condition for nodes to meet.

A lower ρ value means that the there are fewer terminal nodes picked up in the graph, and the criteria for being "of interest" to an investigator is extremely high. A very low ρ specifies that wallets of interested are those which may only hold small amounts of the total funds that it received. A node holding over 90% of the funds might be a holding wallet gradually leaking out funds, it might be a consolidation wallet for a criminal ring, a wallet associated with other types of criminal activity, or even a point of conversion to another cryptocurrency if, for example, the wallet is an Exodus wallet, which allows for wallet level cryptocurrency conversions.

Choosing the right value for ρ allows us to optimally grow the hack subnetwork such that it would include the paths of interest without becoming too large to meaningfully analyze. We found that setting the ratio too high resulted in a less meaningful yet larger hack subnetwork, where the terminal nodes did not adequately capture dynamics of interest, and setting the ratio to be too low did not include clusters that likely should have been included.

Applying a range of ρ from $\rho = 0.02$ through $\rho = 0.98$, in increments of 0.02, had very large implications for the amount of funds considered to be tracked. While changing ρ typically revealed how much of the funds the investigator tracked, at the same time, changing the ρ value does not impact the overall cash out trend witnessed by the investigator.

These results indicate that varying ρ may not be useful for understanding the behaviors of the hacker, but is a useful tool for identifying nodes of interest that could be possible leads to the investigator. Indeed the variance in the ρ parameter proved one of the most useful tools for running community detection.

We finally then needed to handle the introduction of funds at a time later than the hack by either the same or different user. To account for this, we either add these new flows to the funds at the start and work with the new total as our amount of hacked funds, or we incorporate these flows into our ρ definition, by stating a further constraint that if $\rho > 1$, then it is a terminal node and we do not follow its flows forward in time. In the case of the former, we can track all funds engaged in clearly illicit activity, regardless of source, while in the case of the latter, we are actively restricting the subnetwork to funds that explicitly originated from the source of the hack.

Feature definitions

The goal when selecting which distributions to analyze was to capture the behavior of movement of the hacked funds in a precise way. To confirm the hypothesis that the two hacking groups exhibit different cashout strategies, we decided to consider the empirical distributions of 8 different features, as mentioned in Step 4 of the Pipeline.

In the following definitions, the expectations are defined over the nodes of the subnetworks (and terminal nodes in the case of *Transactions*). Additionally, the time units are discretized at the daily level. Lastly, the Initial Hack Amount is the value stolen from the exchange by the hacking group which was the source of the investigated subnetworks.

We define several of the features in our analysis as follows:

1. Amount in Play.
   $AIP = Initial\ Hack\ Amount - \sum_{terminalnodes} weighted\ in - degree$
2. Hack balance of all nodes.
   $Bal = \log(weighted\ in - degree\ -\ weighted\ out - degree)$
3. Logarithmic first difference of the average, LDA, percent of amounts still in play, AIP, across all ρ values.
   $LDA = \log\left(\frac{\mathbf{E}[AIP(t+1)]}{\mathbf{E}[AIP(t)]}\right)$
4. Second difference of AIP, across all ρ values.
   $Second\ Diff\,(AIP) = \frac{LDA(t+1)\ -\ LDA(t)}{LDA(t)}$
5. Logarithmic difference of the standard deviation, $LDST$, of the AIP, across all ρ values.
   $LDST = \log\left(\frac{\mathbf{E}[(AIP(t+1)-\mathbf{E}[AIP(t+1)])^2]}{\mathbf{E}[(AIP(t)-\mathbf{E}[AIP(t)])^2]}\right)$
6. Average number of transactions to terminal nodes, TTN per day, across all ρ values.
   $Transactions = \mathbf{E}[\ TTN]$

Similarity matrices

Once all of the normalized histograms were generated, we measure the pair-wise similarity between them, per variable, via the 1-Dimensional Wasserstein Distance, a.k.a. the Earthmover Distance or L^1 Norm. Generally, the L^p Norm is defined as:

$$W_p(F,G) = \left(\int_0^1 |F^{-1}(u) - G^{-1}(u)|^p\, du\right)^{1/p},$$

where F and G are empirical distribution functions with generalized inverses, F^{-1} and G^{-1} (Villani 2003;).

Community detection

After the similarity matrices are computed for the distributions of interest, the goal becomes differentiating between the two hacking groups. We propose a method of representing the similarity matrices as networks and searching for two distinct communities via both Modularity Optimization and Walktrap and comparing the results.

Modularity Optimization (Clauset and et al. 2004) consists of finding a near maximal value for Modularity, Q, returned from the communities applied to some null model of network formation, typically a Random Network.

$$Q = \frac{1}{2m}\sum_{vw}\left[\left[A_{vw} - \frac{k_v k_w}{2m}\right]\delta(c_v, c_w)\right],$$

where m is the number of edges in the network, A_{vw} is 1 when nodes v and w are connected and 0 otherwise, k_v is the sum of A_{vw} over w, and $\delta(i,j)$ is 1 when i and j are equal and 0 otherwise.

Walktrap (Pons and et al. 2013) operates similarly, also attempting to optimize the same modularity, but with a focus on short random walks exiting communities as the explicit motivation and approach.

Both algorithms are built for analyzing large networks, and their true modularity optimization functions are not explicitly the Q written above, but a derived form.

We utilized both methods as independent confirmation rather than any benefits from their relative optimizations. As the resulting networks are small, with one node corresponding to each hack, are eight distributions analyzed, and two applications of community detection, any conclusions drawn from our method are only tentative since no conclusive results can be drawn from such small amounts of data. Nevertheless, we propose the full method as technically sound and a novel tool in the analysis of hack subnetworks in the bitcoin blockchain.

Results

As discussed in the Methodology, the communities shown in Fig. 10 correspond to those identified by two clustering algorithms with the first two rows being Walktrap's output communities on each distribution's similarity network as seen in Fig. 11, and the second two rows being the results obtained via Modality Optimization. As can be seen, similarity matrices derived from different distribution comparisons, whether analyzed by the same or different algorithm lead to different observed communities. Though they are often different, the communities do share some common characteristics with each other. For example, for all but the clustering of Balance similarity and TvR, nodes $\{B1, B2, B3\}$ are always clustered together. Furthermore, 9 out of the 16 clusters have at least two members of group A together.

To better quantify consensus among the results in Fig. 10, we first find one node N which remains in the same group through all of the methods (we chose node B6) so as to establish a common group naming (in other words, it is no longer the case that a node is either in the blue or the red group seen in Fig. 10, rather that each node is either in the same group as our fixed node or in the opposite group), and then we generate a number $n_{i,j}$ associated to each node i and community j, with $j \in \{1, 2, 3, \ldots, 16\}$, setting $n_{i,j} = 1$ if i is in the same group as N and $n_{i,j} = 0$ otherwise. We then compute the probability of node i being in the same group as N with $p = \frac{\sum_{j=1}^{16} n_{i,j}}{16}$. Finally we bisect the vector of values to along its median and obtain the grouping $\{A1, A2, A3\}$, $\{B1, B2, B3\}$.

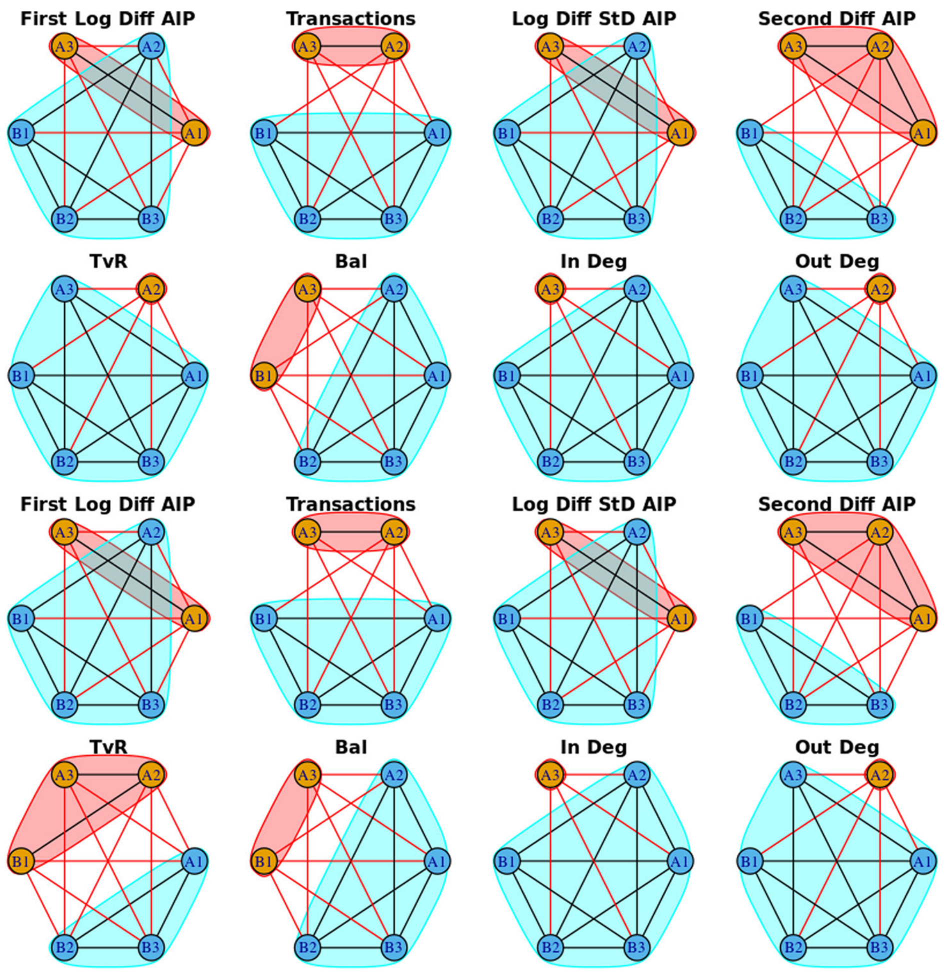

Fig. 10 Communities for all features' similarity matrices - First by Walktrap then Modularity Optimization

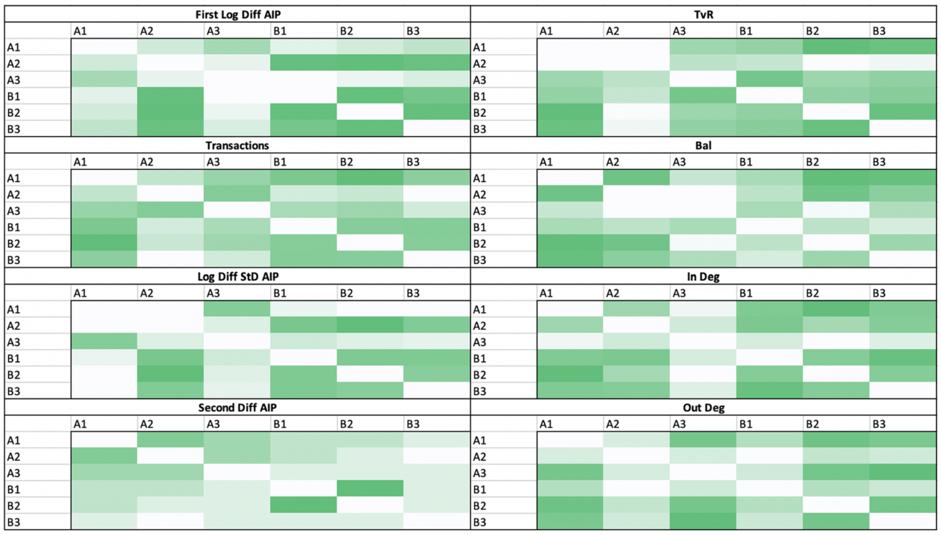

Fig. 11 Similarity Matrices of Feature Distributions for Hacking Groups A and B

This process was repeated using two feature set combinations. The first set contained all 8 features, and its resulting vector was (0.625, 0.5, 0.1875, 0.8125, 1, 1). The second set included only temporal features, namely: LDA, Second Diff(AIP), LDST, and ATVR and had a resulting vector of (0.25, 0.5, 0, 1, 1, 1). Note, that the ground-truth vector is simply (0, 0, 0, 1, 1, 1). In both cases, the bisection works to successfully find the two communities. In the case of only temporal features, the results are even more compelling where 0.5 can be used to bisect the set of hacks into their respective communities.

Discussion

We ran this analysis on historical hacks curated by Chainalysis investigators. The 6 hacks analyzed were carried out by 2 distinct and well-known hacking groups that have been active for the past several years. Each hack was manually classified by the investigators into one of the two groups, which we take as ground truth. We did not include images of these investigations because they visually did not contribute towards understanding the hacking methods.

Analyzing the subnetworks using our proposed methodology allowed investigators to observe the cash out methods for the different hacking groups. Furthermore, the analysis of each subnetwork based on the features above facilitated greater understanding of each specific hack and hacking group, as well as the ability to successfully classify the subnetworks into their respective hacking groups via our pipeline.

Hacking group alpha

We analyzed three distinct hacks carried out by hacking group alpha. Hacking group alpha is a large, well-funded organization. The hacks analyzed in this paper reveal that the subnetworks tracing funds stolen by hacking group alpha are highly complex, with the stolen funds moving through many nodes. The stolen bitcoins are slowly cashed out through terminal nodes overtime. Investigators confirmed this trend.

Funds flowing to terminal nodes from the three hacks visualized in Fig. 2 further confirm this trend. Stolen bitcoin being moved by hacking group alpha appear to slowly leak out of possession of the hackers through terminal nodes. Taking both the first and second differences for the amount in play visualized in Fig. 2 demonstrates that the acceleration at which stolen funds exit through terminal nodes is a significant means of clustering the graphs. Just taking first differences successfully clusters hack A3 and hack A1 together. Visually, A1 and A3 are more similar. Looking at the second differences, i.e. the acceleration, for the amount in play visualized in Fig. 2 is most successful at finding communities of hacks. Running community detection on the similarity matrices for the second differences of the amount in play successfully identifies that A1, A2, and A3 belong in the same community.

The number of transfers that the hackers use to move the funds has also proven significant for helping to effectively classify the hacks according to their hacking groups. As shown in Fig. 3, hack A2 and A3 appear to have similar trends in terms of the number of transfers made each day following the hack. The community detection that we ran on the hacks classified these two hacks together when looking only at trends in the frequency of transactions sent to terminal nodes.

Analysing the variance in the ρ parameter, as visualized by Fig. 8 captures how the share of funds exiting through terminal nodes changes as ρ approaches 1. The standard deviation for the ρ parameter as ρ approaches 1 approximates the variety in behavior for terminal nodes. Using the log difference in standard deviation across the amount in play by varying ρ allows us to classify hacks A3 and A1 together. Both these hacks had similar changes in the amount in play for each ρ over time, whereas A2 had some uncharacteristic behavior for hacking group alpha around day 250. A2 was a much smaller sized subnetwork, with only 55 nodes, than A1 and A3, with 1257 and 218 respectively. This made the standard deviation of the amount exiting through terminal node more sensitive as ρ increased.

We investigated whether the distribution of balances across all the nodes in the hack would be a useful indicator to help classify hacks. This was one of the weakest features used to classify the hacks into hacking groups. As shown in Fig. 4 there is a wide variety in the distribution across all the nodes in the graph based on their hack balances. Hack A3's distribution, for example, had a higher peak, meaning many of the hacks in A3 held a similar balance. Yet A2 had much more variety across the nodes within the graph in terms of how much stolen bitcoins each node ended up holding. Using the distribution of the log balance by nodes was not useful on its own to help classify hacks, and caused one of the few instances of mistakenly grouping hacks A3 and B4 together as seen in Fig. 10.

Hacking group beta

We then analyzed three hacks carried out by the second hacking organization referred to here as hacking group beta. When visualizing the hack subnetworks for hacking group beta, there are striking differences in the cash out mechanisms. Hacking group beta tends to send a majority of its funds through terminal nodes over a short period of time. They tend to sit on their funds quietly, sometimes moving some funds through wallets of interest, but have a characteristically abrupt cash out pattern.

This pattern is visualized in Fig. 2, where hacks B1, B2 and B3 all have notable vertical drops, representing abrupt moments of cashing out through terminal nodes. Running our community detection algorithms on the first differences of this activity successfully classified all B hacks as belonging together, see Fig. 10, yet also identified hack A2 as fitting a similar pattern. The second differences for the amount in play chart is the best at predicting the proper community assignment. Its top performance can be attributed to its correctly capturing the acceleration of the funds exiting through terminal nodes, which confirms the hypothesis put forward by investigators about temporal trends in exiting funds.

All of the hacks from hacking group beta have a large variance for ρ as ρ approaches one, which can also be visualized in 2. This signifies a large range in sending versus receiving behavior for the nodes within the hacking group beta hacks. Funds are exiting through a wide variety of nodes, and not simply hitting one exit point which only ever received funds.

Looking at the distribution of balances held by the nodes within the subnetwork demonstrates the variety of node behaviors present. However, this was again a weak feature when it came to classifying the hacks through community detection. Hack B1 had many nodes that passed through mixing services which were unclustered in the subnetwork. The mixers would siphon off parts of the stolen funds into consolidator wallets in similar patterns. The investigator only tracked the fattest paths, leaving many of the known nodes passing through mixers with a similar balance. Using balance distribution when the graph is not fully built out was shown not to be useful for community detection.

We next looked at the variation in the AIP over all ρ as visualized in Fig. 8. The shape of this graph visualizes how ρ affects the share of funds exiting through terminal nodes. Almost all of hack B1's funds exit through a wide variety of terminal nodes on the first day. The standard deviation peaks at this point, followed by a long period of no fund movements. We successfully classified hacks B1, B2, and B3 together using our community detection algorithms, but hack A2 was mistakenly grouped in when using this feature, as shown in Fig. 10.

We then analyzed the number of transactions going to terminal nodes in Fig. 3. The number of transactions showed no clear visible pattern to help classify the hacks into hacking groups. While the community detection algorithms successfully classified all three hacks from hacking group beta together, it also picked up hack A1.

Key takeaways

We began this analysis by talking with Chainalysis investigators about what they knew about the hacking groups. They indicated that the key differentiation between the two groups, is the pattern by which they hold funds and the subsequent rate at which they cash them out. Our analysis confirms this hypothesis.

We conclude that static features of the charts, such as balance distributions, in degrees, and out degrees are not useful features for classifying the hacks into hacking groups. There are many limitations to these static features. To start, they likely require a fully built out, comprehensive graph. Many of the graphs we chose to analyze were incomplete from the start. This means the takeaways from the static features of the charts were also fundamentally incomplete. Table 1 contains general summary statistics that further reinforce the relative scarcity of meaningful information from the static features for the hacks.

More importantly, our hypothesis of focusing on the temporal features of the subnetworks, rather than the static features was validated. The results indicate that the patterns by which the subnetworks evolve over time serve as useful features for optimal classification based on the method described in this paper. The optimal classifications in Fig. 10, specifically the second difference - or acceleration - of AIP, are most characteristic of the subnetworks temporal nature. Varying ρ to alter our level of resolution into terminal nodes also plays a role in the usefulness of our temporal features and the resulting classifications. The correct classifications were obtained when similarity matrices were built from these temporal features and the community detection algorithms was subsequently run to differentiate the hacking groups based on these features exclusively.

Conclusion

Hacks represent an important challenge for law enforcement, the Bitcoin community, and financial institutions. There is opportunity for an algorithmically informed approach to analysis of existing hacks as well as real time monitoring of hacks. This research represents an attempt at building a more rigorous framework for such an approach via an analysis of both the static and temporal features of hack subnetworks and suggests that the temporal features represent an important avenue of exploration for a deeper understanding of the hack subnetworks.

Future work

In this paper, we have described our proposed approach for analyzing characteristics of the hack subnetworks within the broader bitcoin transaction graph as a means of classifying specific hacks to their respective perpetrating hacking groups. We find that specifically, the temporal characteristics are the most effective for allowing this categorization to occur. Our methods, however, can also be used in other contexts. Open source investigations, for example, can exploit these methods to more effectively track stolen funds from the breach point on the exchange that has been hacked.

This technique also, for example, could be used even in fiat systems such as the swift network. For example, once a potential fraud flag is raised on an account, this method could be used to learn from the behavior of the fraudulent actors. There are, however, limitations to extending this method to the fiat system. One key distinction between our use case and the fiat example stems from how we knew the hack was initiated by either of only two actors. If however one considers a much larger system with many more potential criminal actors, it might take many ground truth examples and a more robust learning algorithm to distinguish between the broader scope of potential illicit actors.

Abbreviations
AIP: Amount in Play; Bal: Balance; LDA: Logarithmic first difference of the average; TvR: Number of Terminal Nodes vs ρ; UTXO: Unspent Transaction Output

Acknowledgements
We thank the Chainalysis investigators for their collaboration.

Authors' contributions
DG, KG, and YS designed research, performed research, and wrote the paper. All authors read and approved the final manuscript.

Funding
This research was funded by Chainalysis.

Availability of data and materials
Due to the sensitivity of the underlying data, we cannot currently release our dataset.

Competing interests
The authors declare that they have no competing interests.

Received: 18 October 2019 Accepted: 4 March 2020
Published online: 16 April 2020

References

Akcora CG, et al. (2019) BitcoinHeist: Topological Data Analysis for Ransomware Detection on the Bitcoin Blockchain
BLOCKCHAIN LUXEMBOURG S.A (2011) Block Explorer. https://www.blockchain.com/explorer. Accessed 12 Dec 2019
Bovet A, et al. (2018) Network-based indicators of Bitcoin bubbles
Chainalysis (2019) Chainalysis Cryptocrime Report 2019. https://blog.chainalysis.com/2019-cryptocrime-review. Accessed 12 Dec 2019
Clauset A, et al. (2004) Finding community structure in very large networks. Phys Rev E 70:66–111
de Arruda GF, et al. (2012) A complex networks approach for data clustering. Phys A 391:6174–6183
Electrum (2011) Electrum Wallet. https://electrum.org. Accessed 12 Dec 2019
ErgoBTC (2019) Tracking Plustoken Funds. https://medium.com/@ErgoBTC/tracking-the-plustoken-whale-attempted-bitcoin-laundering-and-its-impact-on-wasabi-wallet-787c0d240192. Accessed 12 Dec 2019
Huang DY, et al. (2018) Tracking Ransomware End-to-end. In: 2018 IEEE Symposium on Security and Privacy (SP): 20-24 May 2018. IEEE, San Francisco. pp 618–631
Meiklejohn S, et al. (2013) A fistful of bitcoins: characterizing payments among men with no names. In: IMC '13 Proceedings of the 2013 conference on Internet measurement conference: 23 - 25 October 2013. ACM, Barcelona. pp 127–140
Nakomoto S (2009) Bitcoin: A Peer-to-Peer Electronic Cash System. https://bitcoin.org/en/bitcoin-paper. Accessed 12 Dec 2019

Nouh M, et al. (2019) Cybercrime Investigators are Users Too! Understanding the Socio-Tehnical Challenges Faced by Law Enforcement. In: Proceedings of the 2019 Workshop on Usable Security (USEC) at the Network and Distributed System Security Symposium (NDSS), 24-27 February 2019. ACM, San Diego

Pons P, et al. (2013) Computing communities in large networks using random walks. In: IMC '13 Proceedings of the 2013 conference on Internet measurement conference: 23 - 25 October 2013. ACM, Barcelona. pp 127–140

Villani C (2003) Topics in Optimal Transportation, Graduate Studies in Mathematics. Am Math Soc. https://doi.org/10.1090/gsm/058

Villani C Optimal Transport: Old and New. Springer

Yin S, et al. (2017) A first estimation of the proportion of cybercrminal entities in the bitcoin ecosystem using supervised machine learning. In: 2017 IEEE International Conference on Big Data (Big Data). p 17504747. https://doi.org/10.1109/bigdata.2017.8258365

Publisher's Note