Graph-Based Covert Transaction Detection and Protection in Blockchain

Zhenyu Guo, Xin Li, Jiamou Liu, Zijian Zhang, *Senior Member, IEEE*, Meng Li, *Senior Member, IEEE*, Jingjing Hu, and Liehuang Zhu, *Senior Member, IEEE*

***Abstract*— Covert communication is an method that plays an important role in secure data transmission. The technology embeds covert information into data and propagates it through covert channels. The communication quality depends on the choice of channel and data embedding techniques. Recently, blockchain has emerged to become the preferred channel to carry out covert communication for its decentralization and anonymity features. Existing covert transaction methods are constructed *transaction-by-transaction*, which makes them immune to text analysis-based detection methods. However, it is easy to expose their features on the transaction graph level. Unfortunately, there is yet no method to detect covert transactions by the features of transaction graph. In this paper, we propose a covert transaction detection method based on graph structure. By analyzing the statistical features of graph structure for addresses, we can infer whether they are the participants of covert transactions. Furthermore, we design a protection method of covert transactions based on graph generation networks. By adjusting the structural features between different addresses, our method enhances the security of multiple interrelated covert transactions. Experimental analysis on the Bitcoin Testnet verifies the security and the efficiency of the proposed methods.**

***Index Terms*— Covert communication, covert transaction protection, blockchain, graph generative networks.**

I. Introduction

HOW can secrete information be conveyed through a public communication channel and reach the intended receiver without being noticed? This question is explored by investigators of *covert transmission*. A solution to this problem would encode the secrete information and inject it within a digital carrier for transmission from which others cannot extract the covert message [1]. This problem is important in many applications. For instance, military intelligence can be encrypted and encoded before it is transmitted over covert channels to prevent information leakage to adversaries, and business intelligence may be injected into transaction records to bypass restrictions on anonymous transactions [2].

Traditional techniques for covert transmission are insufficient as they rely on network streaming medias as a carrier, which are prone to channel detection and exposure of participants. Network streaming media is easily affected by noise during transmission, where covert information can be corrupted or lost due to interference [2].

On the other hand, the recently emerged blockchain technologies have been recognized as a promising alternative information carrier for covert transmission [3], [4]. Blockchains have become prominent decentralized ledge technology and have widespread applications [5]. A blockchain is a P2P network that uses flooding to propagate transactions and users communicate with each other by the iteration forwarding of neighbor nodes. Therefore, messages in blockchain network have the features of non-directional sending and accidental receiving. This makes embedding and delivering messages in transactions possible [6]. As an information carrier, blockchain enjoys the following: (1) Due to the decentralized nature of public blockchains, transactions within are difficult to identify. (2) On a blockchain platform there is a large number of active users and transaction data packets, as well as numerous ways to encoder covert data [7]. An example of such covert channel applications is zombiecoin [8], [9], [10].

Studies on blockchain-based covert communication can be made from both a *defender* and an *attacker* perspective. A defender seeks to detect covert messages from communication data. The typical method is to train classification models that are able to distinguish covert messages from normal ones. Propelled by advancements in deep learning, neural-based models such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs) have been used for this task [11]. They can conveniently extract rich features from different fields in transaction data packages [12].

Conversely, an attacker aims to nullify the defender's methods and deliver messages without being detected. The attacking methods could be classified by their information carriers. In particular, the existing carries are either *timing* or *spatial* covert channels [13]. The former refers to the transmission time interval of blockchain transaction data packets, and a corresponding attacking method would simulate the time intervals of legitimate data streams while encoding information into data packets [14]. On the other hand, the latter refers

Manuscript received 26 April 2023; revised 18 August 2023; accepted 22 December 2023. Date of publication 28 December 2023; date of current version 8 January 2024. This work was supported in part by the National Key Research and Development Program of China under Grant 2021YFB2701202 and in part by the National Natural Science Foundation of China (NSFC) under Grant 62372149 and Grant U23A20303. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Shouling Ji. *(Zhenyu Guo and Xin Li contributed equally to this work.) (Corresponding authors: Zijian Zhang; Meng Li.)*

Zhenyu Guo, Xin Li, and Jingjing Hu are with the School of Computer Science and Technology, Beijing Institute of Technology, Beijing 100081, China (e-mail: zhenyuguo@bit.edu.cn; xinli@bit.edu.cn; hujingjing@bit.edu.cn).

Jiamou Liu is with the School of Computer Science, The University of Auckland, Auckland 92019, New Zealand (e-mail: jiamou.liu@auckland.ac.nz).

Zijian Zhang and Liehuang Zhu are with the School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing 100081, China (e-mail: zhangzijian@bit.edu.cn; liehuangz@bit.edu.cn).

Meng Li is with the Key Laboratory of Knowledge Engineering with Big Data, Ministry of Education, Hefei University of Technology, Hefei 230009, China, also with the School of Computer Science and Information Engineering, Hefei University of Technology, Hefei 230009, China, also with the Anhui Province Key Laboratory of Industry Safety and Emergency Technology, Hefei 230601, China, and also with the Intelligent Interconnected Systems Laboratory of Anhui Province (Hefei University of Technology), Hefei 230009, China (e-mail: mengli@hfut.edu.cn).

Digital Object Identifier 10.1109/TIFS.2023.3347895

to the fields of blockchain data [15]. These method try to embed secrete messages while trying to keep the statistical information of each field intact.

A. Motivations

Existing methods for blockchain-based covert transmissions, both for the defender and attacker, revolve around the features of single transactions. In particular, these methods all aim at extracting text features from a single transaction [16]: They regard the content of a field as a sentence, which is analyzed or modified to achieve the respective defence or attack objectives. However, when covert transmissions are applied to a blockchain platforms, it is going to be very different. The transactions used to transmit secret messages tend to exhibit notable *group-level* features that can only be identified when groups of transactions are considered together. These features which arise due to many factors such as objectives, frequency, and continuity of communications [17] This lead to our first motivation. **M1.** ***Groups of transactions should be considered together to exhibit group-level features***.

More precisely, if we construct a *transaction graph* (the specific graph definition and construction method is explained in detail in Section III.), where the nodes represent the addresses of the accounts involved in a transaction and the edges between the nodes represent the trading relationships that exist between the accounts, we can more intuitively identify *group-level* features and are best reflected from a *structural* perspective. For any transaction in the platform, by a *covert transaction* we mean one that involves convert transmission of information between the two parties in the transactions. A transaction that is not a covert transaction is called a *normal transaction*. We now separately analyse the nodes that are involved in normal as well as covert transactions:

- The aim of a normal transaction is to ensure the flow of funds between two addresses. In this sense, any address that is used by a normal transaction is likely to engage in long-term and frequent transactions. In the transaction graph, the nodes that correspond to these addresses are likely to form star-like sub-structures that consist of a central node who are connected with multiple other nodes via edges.
- On the other hand, a covert transaction aims to deliver information from one address to another; the flow of funds is a necessary but semantically inconsequential aspect of the transaction. In this sense, the funds could be used again for future covert transactions, while the addresses that are involved in covert transactions are likely to be one-off addresses that are discarded after being used. In this sense, the nodes that corresponds to these addresses in the transaction graph tend to form chain-like sub-structures that indicate the continuous flow of funds.

This lead to our second motivation. **M2.** ***Structural features can be used to detect or protect groups of covert transactions.***

B. Technical Challenges

We consider *structural properties* of the transaction graph as a basis for the detection and protection of covert transactions. and face with three technical challenges. **C1.** ***How to detect covert transactions with graph structural properties.*** he transaction records themselves are stored in text form and have no graph properties, while the structure of the transaction graph is so complex that we need reasonable measures to extract and select features for identifying covert transactions. **C2.** ***How to protect covert transactions with graph structural properties.*** The structural properties of the graph are jointly determined by all the nodes and edges in the graph, so protecting a single transaction record hardly changes the structural properties of the whole graph, and it is not possible for us to directly re-embed covert transactions into normal transactions that have already been executed. It is necessary to design a robust method for blockchain-based covert transmission that avoids being detected by the structural properties.

As solution for C1, we design an approach that distinguishes covert transactions from normal transactions. The idea of this approach amounts to some rather simple statistics of the graph structure. The fact that such a straightforward approach exists to accomplish the detection task means the inherent flaws when using blockchains as a venue for covert transmission.

As solution for C2, we leverage the recent advances in deep-learning based graph generation models. In particular, we propose an attack method of covert transactions based on graph autoencoders. By learning the clustering features of normal transactions, the graph autoencoder can direct how covert transactions could be conducted and reduce the exposure risk.

To validate our claims, we present experiments on the normal transactions and covert transactions carried out on a real blockchain platform. It is found that our detection method presented can achieve more than 90% classification accuracy. With the graph autoencoder-based model for attack, however, the accuracy of the detection method is reduced to less than 10%, which verifies the effectiveness of our proposed method for covert transactions protection.

C. Contributions

The contribution can be summarized as below:

- We are the first to consider the transaction graph and group-level properties in blockchain-based covert transmission.
- We design a simple analytics-driven method to extract group-level structural properties from the transaction graph.
- We design a blockchain-based covert transmission method using unsupervised graph generation models.

The rest of this paper is organized as follows. Firstly we recall the related works in Section II. Then, the detection and protection methods are proposed in Section III and Section IV, respectively. Following by that, Section V shows the experiments. Finally, the conclusion is drawn in Section VI.

II. Related Work

In this section, we review existing covert transmission methods that are based on blockchain, from both the attacker and defender perspectives.

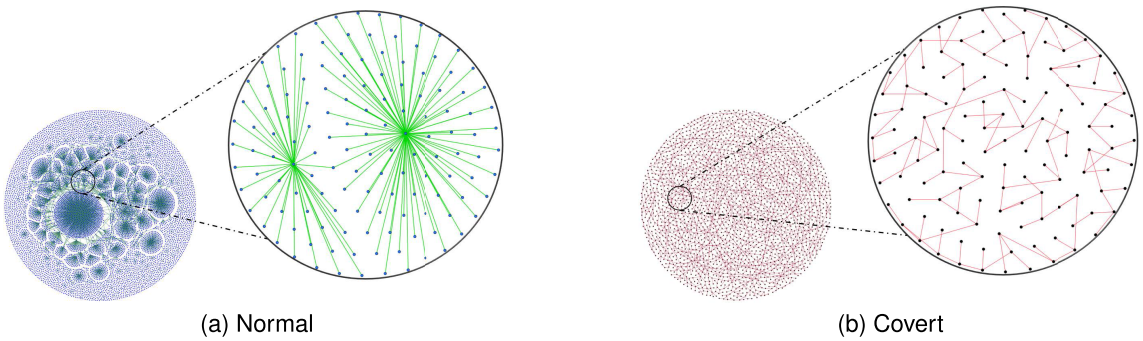


Fig. 1. Group-level features of transactions. (a) shows the clustering of normal transactions with star-like structures; (b) shows the clustering of covert transactions with chain-like structures.

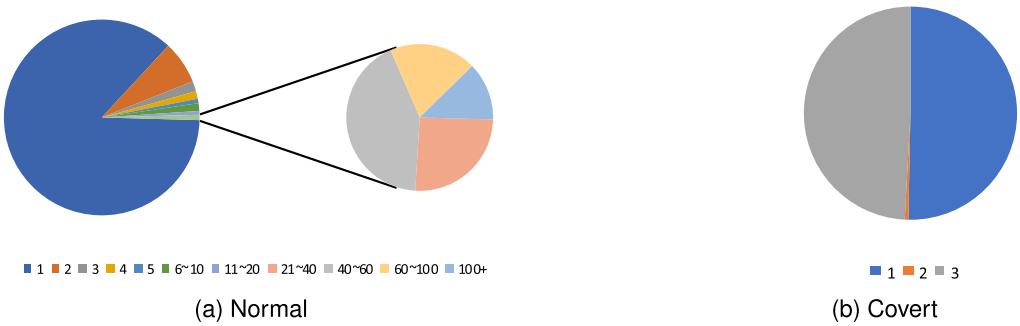


Fig. 2. Group-level features of transactions. (a) shows the clustering of normal transactions with star-like structures; (b) shows the clustering of covert transactions with chain-like structures.

A. Attack

Traditional covert transmission methods usually hide secret information by modifying data bits in multimedia files and re-encode them [18]. However, the immutability of blockchain makes it difficult to apply these methods directly [19]. One thus needs a different scheme to achieve covert transmission through a blockchain. So far only a handful of work have addressed this issue. These existing work mainly focus on two types of covert channels: *timing covert channels* and *spatial covert channels*.

1) Timing Covert Channels: Attacking methods that utilize timing covert channels achieve covert transmission through intervening the mode of the transaction. For an instance, the sender and the receiver can negotiate a parameter t, as a fixed time interval. For each time interval, if the sender needs to send hidden information '1', it sends a data packet within the time interval t. If it sends hidden information '0', it keeps silent during this time interval [20]. This is the most traditional timing covert channel. When people try to apply timing covert channels to transactions, they found transaction records that are generated by different protocols allow the users to fill in different fields. Therefore any attempt to design covert transmission methods should not only consider the information encoding mode, but also other methods such as channels and protocols. Xu et al. [14] proposed broadcasting steganography in the blockchain, they chose a part of transactions in a block according to a secret key, and embedded the secret data by repeatable-address arrangement to change the chronological distribution of addresses in the block, which inspired the research of time covert channels using rearrangement. Subsequently, Zhang et al. [15] proposed a covert communication method based on the whistler protocol to covertly transfer information in the Ethereum network. They compared the probability of data tampering in the blockchain and traditional channels through scenario simulation and parameter settings, and found that the amount of information in the proposed method is 4.7 times that of the traditional timing covert communication. For our paper, we assume that during the period of time where covert transmissions are supposed to take place, the communication protocol stays the same. Therefore, the temporal characteristics of all covert transactions are the same.

2) Spatial Covert Channels: Attacking methods that utilize spatial convert channels achieve covert transmission through modifying the transaction content. BLOCCE is the first attempt to establish a provably secure covert communication on the blockchain [21]. The method hides the communication message in the transaction address to establish the connection between the sender and the receiver. However, a single block can only encode 1 bit of data in this way, and it also requires the communication parties to consult about the marker with each others in advance, which introduces additional communication overhead. To solve these problems, Song and Peng [22] proposed BLOCCE+, they made use of the single transaction multi-address encoding and single block multi transaction submission and improve the communication efficiency of the system and the continuity of the communication process. Zhang et al. [15] used Vanitygen to generate addresses for encoding information, and proposed V-BLOCCE to improve the efficiency of information transmission. Liu et al. [23] proposed a multi-bit encoding method based on Hash to enhance concealment. Besides the encoding methods in different fields, there are also covert methods by the other sides. Tian et al. [24] proposed DLchain, which is

a dynamic label distribution generation algorithm based on the statistical distribution of real transaction data. It aimed at solving the problem of exposing covert transactions with fixed labels; Cao et al. [25] used hash chain and elliptic curve with Diffie Hellman chain to guide the derivation of the public key and outperformed than other covert methods.

B. Detection

The detection technology of covert transactions provides an important basis for the disclosure of covert information. But few studies focus on detecting covert transactions in blockchain so far as we know. Some work focused on analyzing the statistical features of the channels. For example, Giron et al. [26] carried out a series of experiments on this topic. They first selected the LSB field and nonce field of the address and used the idea published on Lerner [27] for statistical analysis, but this method failed to achieve good detection accuracy. Then, the authors put forward the idea of clustering analysis which depends on the specific clustering algorithm. On the other hand, Kappos et al. [28] chose Zcash coin as the research object. By simple heuristics they reduced the size of the overall anonymity of Zcash by 69.1%. Kharraz et al. [29] discussed the different features of blockchain for covert mining, and selected seven features to build a support vector machine (SVM) classifier.

Other covert detection methods aim at detecting illegal transactions by analyzing the changeable patterns of channel features. Neto et al. [30] proposed a dynamic online mechanism to detect and block covert cryptocurrency mining flows by machine learning on software-defined networking, and regarded the classification probability as the features of incremental learning classifier. In this way they achieved high classification accuracy. Gangwal et al. [31] utilized the hardware performance counters (HPC) to create signatures representing positive examples and used deep learning technology to build classifiers. They emphasized that the proposed method was not limited to cryptocurrency mining, and can achieve reasonable classification results for each single transaction recor.

In general, existing studies on blockchain-based covert channels have been mainly devoted to spatial covert channels, in which the information encoding of transaction content is the main research topic, but they ignore the correlation and concealment between multiple transactions. Besides, the detection of covert transactions has been put into practice but there is a lack of specific covert transaction analysis methods for blockchain in the existing work, which means that there is no special auxiliary detection mechanism in the current blockchain network. Considering these two effects, we firstly designed a graph-based detection method by analyzing the correlative features between multiple transactions. Then we proposed a protection method by graph generative networks to protect the concealment between multiple transactions. As far as we know, the problem of detection and protection for covert transactions with graph has not been studied.

III. How the Graph Reveal Covert Transactions

As mentioned before, when multiple covert transactions appear continuously, they often expose the correlation features different from normal transactions. Therefore, we propose the graph-based covert transaction detection with structure measurements. Specifically, we firstly build a transaction graph according to all the transaction information monitored in a period of time, then try to recognize these features from the graph structure and detect the related covert transactions.

A. Threat Model

We assume a threat model similar to existing work for covert transaction represented by BLOCCE+ [22], [32] [33], [34], whereby participants would use multiple methods to embed covert information in different fields of the transaction record. Most participants in covert transactions are honest and will only occasionally use covert transaction to pass on information truthfully. A small part of them is malicious. They may use covert transactions to launch attacks on users or platforms [23]. Attacks against users mainly refer to tracking them. By embedding covert information, the sender of a covert transaction can snoop on the unknowing recipients of the transaction. Once an unsuspecting user is involved in this covert transaction, they receive covert information associated with the sender. It allows the sender to track the address and its user, further probing their information and undermining the anonymity of the blockchain. Attacks against the platform mainly refer to disrupting the platform network. When transmitting information, participants in a covert transaction may initiate a large number of transactions in a short period of time. These large-scale transactions are not inherently economical activities, but they still need to be dealt with by the platforms. These transactions would push the normal transactions asides and occupy the network resources, placing an unexpected burden on the platform.

B. The Transaction Graph

We consider the following graph defined on transaction records on a blockchain platform, which we call the *transaction graph*:

Definition 1: Definition[Transaction graph] A *transaction graph* is a acyclic graph with at least one connected component (subgraph), where each *node* denotes an address that participates in a transaction, and connect an *edge* between any two nodes if they have flows of funds between them.

According to the rule of blockchain transactions [35], each record represents flows of funds from input addresses to output addresses. The transaction platform records the total amount involved in the transaction but cannot analyze the account distribution between individual addresses. In fact, almost all input addresses and output addresses will inevitably flows of funds due to "Unspent Transaction Output (UTXO)" [36]. Therefore, we regard a transaction record as a *transaction bipartite graph*:

Definition 2: Definition[Transaction bipartite graph] A transaction graph is a complete bipartite graph, where input addresses and output addresses become the two point sets and any input address is connected to each output address by a unique edge.

A transaction bipartite graph indicates flows of funds between input addresses and output addresses in a single

record. And we can combine transaction bipartite graphs with the same address node to draw a transaction graph for all the transaction records.

According to Definition 2, we construct two versions of a directed graph and an undirected graph for transactions in the same period of time, and the direction in the directed graph is from input address to output address as shown in Figure 3.

The purpose of constructing a transaction graph is to investigate the feature relationship between multiple connected transactions and detect covert transactions. Therefore, we hope to find more common addresses between different transactions and associate them. The ideal result is that the transaction records per a period of time can be drawn in the same connected graph. However, the actual transaction cannot guarantee that all addresses in a period of time can be connected with each other, and there are often some outlier nodes and transactions. In this case, there will be multiple small-scale subgraphs in the transaction graph, such as multiple connected components with only two associated nodes. Although such a structure can also be analyzed by graph theory, there are few structural features worthy of learning due to the scarcity of the number of nodes and edges. In fact, if the number of edges of a subgraph is less than 3, we believe that the address corresponding to this subgraph has carried out at most two transactions and using the text-based analysis method to directly analyze those transactions will be faster than using the graph-based method. That is to say, it is not necessary to treat it as a transaction subgraph. Therefore, we can analyze the transaction subgraphs with the number of edges less than 3 by some text-based method and deal with remaining data as the *effective data* we can model by graph theory.

At the same time, the measurement method of graph structure is often aimed at a connected graph. Therefore, we divide the transaction graph per a period of time into multiple transaction subgraphs according to the degree of connectivity and measure the relevant indicators of each subgraph respectively.

C. Method

There are many methods to identify the graph structure, among which the methods based on structure measurement is the most basic and commonly used. It has the advantages of low time complexity, high interpretability, and convenient implementation. It can summarize the structural features of the graph from the aspects of aggregation degree and node distance, especially when the number of transactions per a period of time is large and the transaction graph presents complex network features, structural measurements can still better express its covert information with lower time cost. This advantage is particularly significant When the model is deployed to the real world and used to analyze the high volume of transaction records in a short period of time.

The most prominent feature of covert transactions on the graph is the direction and distance of capital flow. Therefore, we investigate the typical structural measures in the directed graph: the in-degree of nodes, the out-degree of nodes, and the longest path length. In fact, there are many graph structure metrics, besides the three we have used, there are also graph density, graph modularity, graph pagerank, etc., but they are more complex than those we have chosen, e.g. pagerank requires a lot matrix multiplications, whereas the degree-based methods only need to count the edges of the graph. We have preferred simple graph structure metrics where they can deal with the detection task.

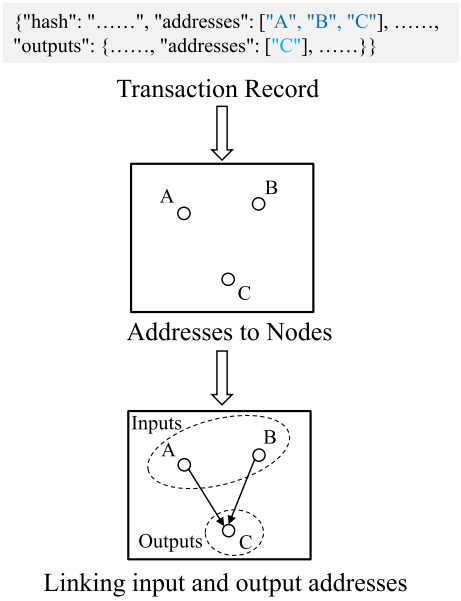


Fig. 3. Construct a transaction graph from transaction records.

- In-degree of nodes refers to the number of edges pointing to other nodes starting from the current node. In a transaction graph, it shows the ability of a node to create transactions. The more the in-degree is, the more the amount of funds paid by the address in the current node;
- Out-degree of nodes refers to the number of edges pointing to the current node starting from other nodes. In a transaction graph, it shows the ability of a node to receive transactions. The more the out-degree is, the more times the address in the current node receives funds;
- Longest path length is also known as network diameter, it refers to the maximum value of the shortest distance between any two reachable nodes in the graph. In a transaction graph, it shows the maximum length of the flows of funds in a series of connected transactions bipartite graph. The longer the longest path is, the more complex the current transaction process is.

However, for the transaction records in a period of time, the structural measures calculated in subgraphs of different sizes cannot be compared in the same horizontal line. In other words, the measurement results need to be standardized to objectively reflect the structural differences of different subgraphs.

- The *out-degree* and *in-degree* of nodes in the graph essentially measure the distribution of transaction frequency at different addresses. Therefore, after measuring the out- (in-)degree of all nodes in a transaction subgraph, we calculate its variance, and take the variance value as an index to measure the deviation of transaction volume distribution in the subgraph;
- The *longest path length* of the graph essentially measures the flow direction and reuse of funds. Therefore, after

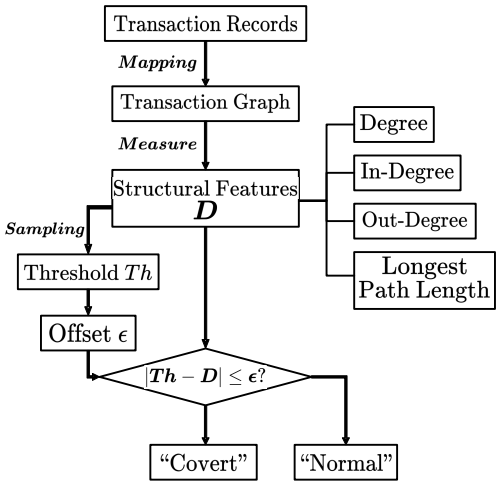


Fig. 4. Steps for detecting by graph features.

measuring the maximum path length in a transaction subgraph, we divide it by the total number of nodes in the current subgraph and take the result as an indicator to measure the length distribution of the capital chain in the subgraph.

We have experimented with these three measurement methods on the observation data set and found that using the ratio of the longest path length to the number of nodes can effectively distinguish the normal transaction subgraph from the covert transaction subgraph. Specifically, we follow the principle of the Occam razor and use a relatively simple judgment process, as shown in Fig. 4.

Firstly, for the collected block transaction records, we got the *effective data* after filtering and construct the transaction graph according to the rules we described earlier. Then, we calculate the relevant graph structure measures and their statistical features (D), including the variance of node degree and the $maximum\ path\ length/number\ of\ nodes$. From the calculated statistical feature values, we randomly sample a value as the comparison scale (threshold, Th), and select the minimum value of the difference between any two measure numbers as the base of the offset, then use its integer multiples as multiple offsets.

If a feature of a subgraph is within the floating range, the subgraph is determined as a covert transaction subgraph, in which the designed transactions are determined as covert transactions. Otherwise, it is judged as a normal transaction subgraph.

IV. Covert Transactions With Graph Generative Networks

To protect group covert transactions in terms of graph structure features more comprehensive, it is necessary to make covert transactions have the same or similar graph distribution features as normal transactions, which requires covert transaction participants to agree on relevant strategies before creating transactions and avoid structural exposure risks through agreed transaction dynamics. Theoretically, we can analyze the specific distribution features of a single covert method and compare it with normal transactions, to design how to avoid risks. However, in practice, the covert methods we need to use are different according to the specific scenarios, and even multiple covert methods will be applied alternately, Therefore, we need to adopt a graph structure transformation method that does not depend on the specific types of covert transactions.

On the other hand, to analyze the structural features of transactions better, it is necessary to obtain a large number of transaction data. However, most of the transaction flow data that can be obtained and used for analysis are normal transactions, and few covert transaction records can be obtained for the outside world, so it is impossible to extract effective group hidden features. In fact, as covert transaction participants, they will neither agree to provide historical transaction data nor the planned covert transaction data to the machine learning model for analysis, because it will undoubtedly let their private information out.

To sum up, the supervised learning method in the general sense can neither meet the needs of the currently covert transactions scene nor be recognized by the transaction participants. Therefore, we propose a method that learns the feature of normal transactions and imitates its structural distribution to provide a reference method for the planned covert transactions rather than analyzing the feature differences between normal transactions and covert transactions.

A. Graph Autoencoder

Graph auto-encoders (GAE) [37] is an unsupervised learning model widely used to learn graph representations. By learning the embedding representation of nodes in the graph through the Encoder-Decoder structure and applies to downstream tasks, it can map each node to a vector where the distance characteristics among nodes is preserved and makes the relational knowledge of interacting nodes to be stored and accessed efficiently, and this is what graph representation learning (or embedding) does [38].

Given an unweighted graph $G = (V, E)$ with $N = |V|$ nodes, we introduce an adjacency matrix A of G and the degree matrix D. According to III-B, each node represents an address in transaction records and edges indicate they have flows of funds between them. We further introduce the node features matrix X. However, the addresses do not keep any feature information in blockchain transactions, so we randomize the feature matrix in the transaction graph so that each node follows the same feature distribution. And we send A and X as the input to the GAE. The encoder is used to convert the input graph into latent variables (embedding) of nodes Z with the feature matrix X and the adjacency matrix A. The decoder is used to reconstruct the adjacency matrix graph A^* according to the latent variables Z by inner product:

$$A^* = \sigma(ZZ^T)$$

Therefore, the reconstruction error is taken as the loss function and calculated by cross-entropy:

$$L = \mathbb{E}_{q(Z|X,A)}[logp(A|Z)] \tag{1}$$

Based on the unsupervised deep learning model represented by the GAE, we use the normal transaction data as the training

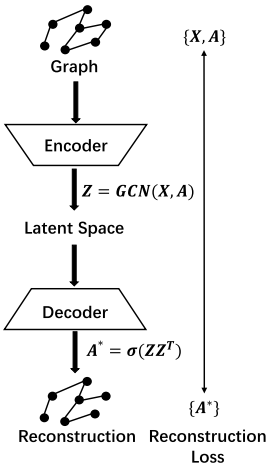


Fig. 5. The Architecture of GAE.

set training model to make it learn the association relationship and distribution between different addresses and transactions in the normal transaction. After that, we only need to know the number of addresses planned for covert transactions as N nodes, then a $N \times N$ matrix can be constructed as input, e.g. a random matrix whose elements follow a uniform distribution over [0, 1]. And the trained model can supplement the transaction relationship between these addresses, which can lead this batch for covert transactions.

The GAE structure we use can be shown as Fig. 5. The structure of the encoder can be selected according to specific data or tasks. Here, we choose GCN to build the encoder, and the latent variables Z is calculated by:

$$Z = GCN(X, A) = \hat{A}ReLU(\hat{A}XW_0)W_1 \tag{2}$$

where $\hat{A} = D^{-\frac{1}{2}}AD^{-\frac{1}{2}}$, W_0 and W_1 are parameters that need to be learned.

The encoder part is composed of two layers of graph convolution, which is used to extract the association relationship from the adjacency matrix and feature matrix of the transaction graph. Then the encoder can convert them into the feature vector in the latent space; The decoder part is in the form of matrix inner product, which is used to reconstruct the input adjacency matrix. We use the random matrix instead of the feature matrix for the input of the encoder, which can not only avoid the learning impact of various fields of specific transactions on the structural features in the training and improve the detection ability of structural features, but also reduce the dependence on covert transaction information in the application stage. In other words, the covert transaction participants do not need to input the specific message to be delivered into our protection model, so they don't have to worry about the model giving away the content of the message.

B. Variable Graph Autoencoder

In the experiments, we found that the transaction subgraphs generated by GAE model have almost the same features, which will make the covert transactions guided by GAE highly similar. That is because the generation ability of GAE depends on the structure distribution in the training dataset, the model can only generate the structure that is shown in the training process, which is not conducive to the development of covert transactions [39]. Therefore, we introduce VGAE.

Variable graph auto-encoders (VGAE) is developed from GAE. However, Z in VGAE is not directly obtained from the encoder, but rather is attached to a sampling process from Gaussian distribution to improve the generation ability of the model. The encoder in VGAE is used for computing the mean μ and variance σ of a Gaussian distribution. Then the latent variables Z is sampled from the Gaussian distribution. They can be shown as follows.

$$q(Z|X, A) = \prod_{i=1}^{N}(q(z_i|X, A)) \tag{3}$$

$$q(z_i|X, A) = N(z_i|\mu_i, diag(\sigma_i^2)) \tag{4}$$

By transforming the latent space described numerically in GAE into the latent space described by a probability distribution, VGAE model can sample the graph structures from a Gaussian distribution, it means we add some noise into the model, which helps to deal with the invariable deadlock of GAE. Also we utilize a reparameterization trick to calculate Z: we firstly sample a ϵ from the standard normal distribution and then get Z by $z = \sigma\epsilon + \mu$. After that, we can complete the sampling process and ensure that the gradient can propagate backward correctly. Finally the generative model is given by:

$$p(A|Z) = \prod_{i=1}^{N}\prod_{j=1}^{N} p(A_{ij}|z_i, z_j) \tag{5}$$

$$p(A_{ij} = 1|z_i, z_j) = \sigma(z_i^T z_j) \tag{6}$$

The loss function of VGAE is determined by the reconstruction error and the Kullback-Leibler(KL) divergence. Aiming to add standard Gaussian noise to Z, the distribution of Z should be as close to the standard normal distribution as possible, so the loss function can be:

$$L = \mathbb{E}_{q(Z|X,A)}[logp(A|Z)] - KL[q(Z|X, A)||p(Z)] \tag{7}$$

$$KL(q||p) = \sum_{x\in X} q(x)log\frac{q(x)}{p(x)} \tag{8}$$

$$p(Z) = \prod_i p(z_i) = \prod_i N(z_i|0, I) \tag{9}$$

The process of "sampling" from Gaussian distribution by VGAE can be regarded as introducing Gaussian noise into Z. In other words, the uncertainty created by this generation process makes the model predict some edges that have never appeared in the training set, which aims to improve the generation ability of the model and adapt to downstream tasks better. The VGAE structure is shown as Fig. 6.

According to the reparameterization trick, the model can learn two parameters σ and μ of Gaussian distribution from two different GCN layers respectively, and then generate the reconstructed graph A^*; As a generative model, we strengthen the ability of downstream tasks.

Based on the existing protection loss, we add the link prediction task to make the GAE obtain a better graph structure

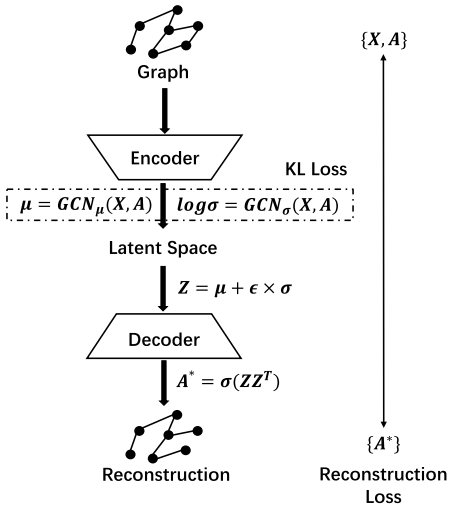


Fig. 6. The Architecture of VGAE.

generation ability. We hope that the model can not only reconstruct the input graph but also supplement the existing transaction edges. Specifically, for a transaction graph with adjacency matrix A, we partially mask the edges to obtain its subgraph with adjacency matrix A', then regard A' as the input of the GAE to obtain the reconstructed graph A^*, and reckon the reconstruction loss between A^* and A rather than A', to improve the reasoning and generation ability of the model for the vacant edges.

C. Transaction Records Reconstruction

With the help of graph generative models, we can obtain a transaction graph that is reconstructed based on the existing address nodes. In this section, we will explore how to reconstruct the transaction records based on the transaction graph.

According to the transaction graph construction rules we mentioned before, the edges in the graph represent a fund flow between two associated nodes, and the exact amount of fees can be determined by the users, so we propose a reconstruct method based on address merging. We first define a transaction record unit as a transaction record consisting of only an input address and an output address, then the reconstructed transaction graph can be considered as a set S of many transaction record units whose number is equal to the number of edges in the graph. Each of these units corresponds to an edge, and the two nodes associated with this edge are the input address and output address of this transaction record respectively. Since there are almost no records consisting of only two addresses in normal transaction records, we need to merge some of these units to form new records.

For any record unit, we randomly select another unit and merge their input address field and output address field, respectively. Note that when selecting, we need to guarantee that the input address of one unit does not appear in the output address of the other unit, because addresses are not allowed to transfer funds back to themselves in a blockchain transaction records. Until the number of remaining transaction records $|S|$ meets the user's expectations R. Algorithm 1 describes this process formally.

Algorithm 1 Reconstruct Records From a Graph

Require: S: The set of transaction record units from the graph; R: The number of transaction records we need; F: The number of failures allowed in a selection;
Ensure: S: The set of transaction records reconstructed from the graph;
while $|S| > R$ **do**
  Select two records A and B from S randomly;
  if $A[Input][Addresses] \cap B[Output][Addresses] == \emptyset$ And $B[Input][Addresses] \cap A[Output][Addresses] == \emptyset$ **then**
    Create an empty transaction record C;
    $C[Input][Addresses] = A[Input][Addresses] \cup B[Input][Addresses]$
    $C[Output][Addresses] = A[Output][Addresses] \cup B[Output][Addresses]$
    $S = (S - A - B) \cup C$
  else
    if Failed to select with more than R times **then**
      Break;
    end if
  end if
end while

V. Experiments

Based on the two methods, we firstly carried out the graph-based covert transaction detection with structure measurements in the data collected on the real blockchain trading platform to show the effectiveness of the method.[1]

For these detected covert transaction clusters, we use the graph generative networks to reconstruct their relative structure and use the detection method to detect again to prove the protection for covert transactions.

A. Detection

At first, we carry out a covert transaction detection experiment to prove the effectiveness of the proposed method and explain that the current commonly used covert data embedding methods do not protect the clustering features of covert transactions.

1) Datasets: The transaction records we used for the experiment were all derived from real blockchain trading platforms, totaling about 3,000 records, involving more than 10,000 addresses.[2] Based on the rules we described earlier, we organized them as a transaction graph with a total of 9000 edges. After preprocessing, these transactions are arranged into 167 subgraphs involving 7,410 nodes.

In the data we collected, covert transactions can be divided into three types according to their embedding methods. Covert information is embedded into addresses fields [22], signature fields [32] and the OP_RETURN in the return fields [33], [34] respectively. These methods are commonly used in the current blockchain covert transactions and are representative.

2) Evaluation: For the covert transaction detection, we regard it as a binary classification task. Therefore, we hope that the detecting method we used can find out as many covert transactions as possible, achieving a high $recall$.

[1]The code underlying this article are available at https://github.com/Zarachodar/GAEforBTC.

[2]The data underlying this article are available at https://github.com/1997mint/covert-transaction-model.

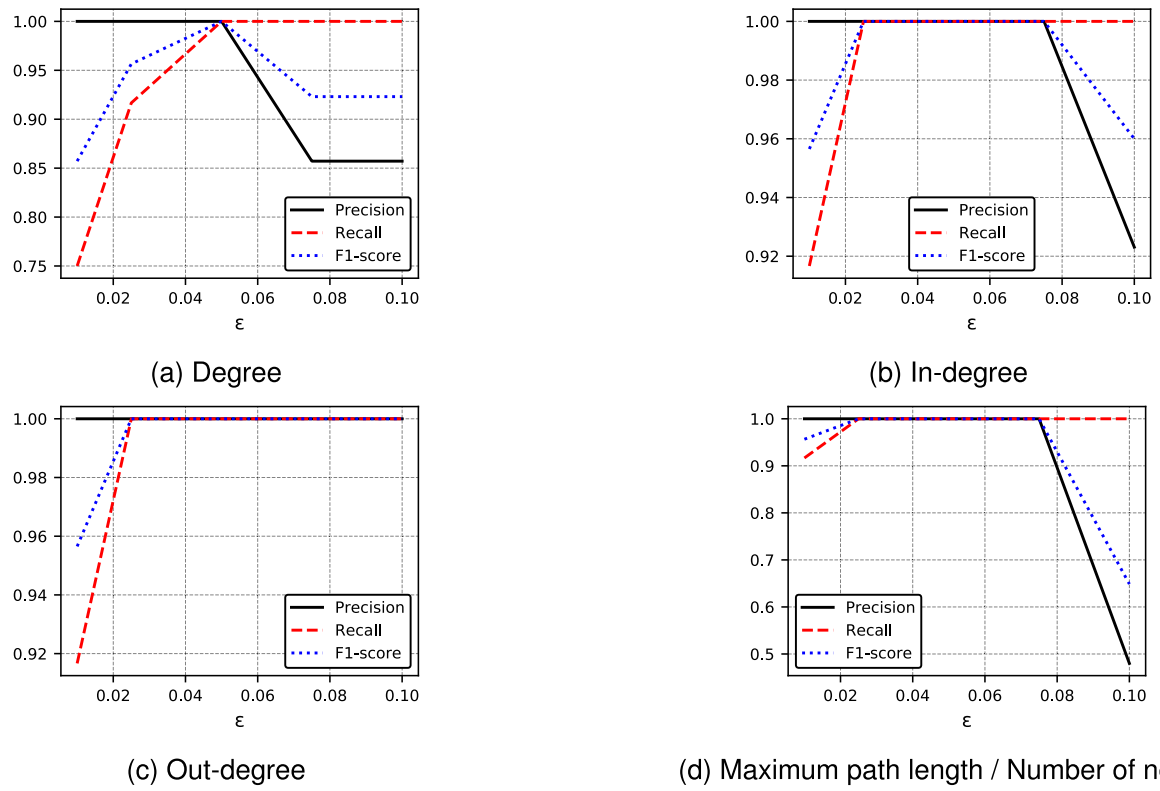


Fig. 7. Results for different metrics in different ϵ. For each graph structure metric, we can find a range of thresholds that makes the detection of covert transactions very accurate.

Recall is the ratio of the TPs to the sum of the TPs and the false negatives (FNs), which represents the completeness of the classification:

$$Recall = \frac{TP}{TP + FN} \tag{10}$$

At the same time, we do not hope that the method easily identifies a normal transaction as a covert transaction, which means the model has great $precision$.

$Precision$ means the ratio of the true positives (TPs) to the sum of the TPs and the false positives (FPs), which represents the confidence of the classification:

$$Precision = \frac{TP}{TP + FP} \tag{11}$$

Therefore, we choose the $F1 - score$ to evaluate the results, and it can be shown as:

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{12}$$

$F1$ is the harmonic mean of the $precision$ and $recall$. It can comprehensively consider them to better measure the detection capability of the method than a single one. The higher the F1 score, the stronger the detection ability of our method for covert transactions.

3) Settings: Different measurement metrics show different sensitivities to covert transactions, so we set the thresholds(Th) respectively, as shown in Table. I. We choose {0.01, 0.025, 0.05, 0.075, 0.1} as three offsets for detecting.

4) Results: Fig. 7 shows the of applying four graph structure metrics to detect covert transactions on the real transaction scenario data set we collected. It can be seen that in the process of judging 167 transaction subgraphs involving 7410 nodes, **the detection method based on graph structure shows a pretty high** $recall$ **(almost 100%) for the covert transaction subgraphs,** which proves that almost all the existing covert transaction development methods have not considered the protection of group features when carrying out covert transactions, and also shows the feasibility of the method based on graph measurement in the process of covert transaction detection.

TABLE I
THRESHOLDS FOR METRICS

Metric	Th
Variance of node degree	1
Variance of in-degree	0.01
Variance of out-degree	1
$Maximum\ path\ length/number\ of\ nodes$	0.5

Differences in results between metrics:

Analyzing the degree of nodes performs better than $maximum\ path\ length/number\ of\ nodes$ in terms of average $precision$ and average $recall$. Specifically, The $maximum\ path\ length/number\ of\ nodes$ feature has always maintained good results in terms of $recall$, However, when we set ϵ to 0.1, it will mistakenly identify many normal transactions as covert transactions, reducing $precision$. Although the node degree also has such a shortcoming, the number of samples misclassified is less. In fact, from our previous statistics on the distribution of node degrees in the transaction graph, we can see that there are great differences between normal transactions and covert transactions, and the molecular graph shows this distribution more clearly, to obtain better detection results.

TABLE II
LINKING PREDEICTION RESULTS ON BTC DATASETS BY GAE

Dataset	ValAUC		ValAP		TestAUC		TestAP	
	Records	Avg	Records	Avg	Records	Avg	Records	Avg
BTC	0.62329	0.58404	0.68963	0.67337	0.60211	0.59786	0.68387	0.68527
	0.52473		0.62187		0.54987		0.65720	
	0.60447		0.68685		0.58606		0.67121	
	0.59315		0.69495		0.65146		0.73176	
	0.57455		0.67356		0.59980		0.68233	
BTCpart	0.75938	0.75679	0.82804	0.82566	0.77388	0.74423	0.83098	0.81703
	0.70120		0.77442		0.74738		0.83080	
	0.75671		0.83152		0.73744		0.80619	
	0.77578		0.84711		0.69251		0.78477	
	0.79086		0.84722		0.76995		0.83241	

Effects of parameters:

For different kinds of degrees of nodes, the detection effect of out-degree and in-degree after subdivision is significantly better than a single degree, showing that the chain structure and star structure mentioned earlier are indeed the main structural features used to distinguish between normal transactions and covert transactions. Combined with the sampling scale, when taking 1, only one covert transaction subgraph of the out-degree of nodes falls outside the floating range of 0.01, The performance is better than taking the in-degree of 0.01, so we prefer to take the out-degree of nodes as the detection measurement feature; As for the $maximum\ path\ length/number\ of\ nodes$, it is greatly affected by the offset. Therefore, special attention should be paid to the selection of the offset value when using it. When we set the sampling scale of node degree and out-degree as 1, or set the scale of the node in degree as 0.01, or set the scale of $maximum\ path\ length/number\ of\ nodes$ as 0.5, our method achieves the best results. As long as the corresponding scale and offset range can be selected, these graph structure measurement methods can achieve a better distinction between clustering normal transactions and covert transactions, which proves the effectiveness of the graph-based method.

B. Protection

On the basis of the detection experiments, we implement the protection method based on the graph generation model for the existing covert transactions and use the proposed method for secondary detection. Then we compare the results before and after to illustrate the effectiveness of the proposed protection method.

1) Datasets: In order to comprehensively estimate the feature learning ability of GAE and VGAE on different transaction graphs, we divide the obtained data into two sets: BTC and BTC_{part}. BTC is the transaction graph formed by all normal transactions in a period of time, and in this set we can find multiple connected components of different sizes; BTC_{part} is the transaction graph formed by the normal transactions related to each other in a period of time, and we have only one connected component in this set. Obviously, BTC_{part} is included as a proper subset of BTC.

2) Settings: For fair comparisons, we set the same GCN layers for all baselines: each encoder consists of two GCN layers, and the first hidden layer has 32 units while the second one has 16. We set the initial learning rate as 0.01 and the dropout rate as 0. The specific structure and parameters of the network are listed in Table. IV.

For link prediction, we randomly sample 90% neighbors of each node for training and the rest for testing. We train the model for 100 epochs, repeat the prediction procedure 5 times and evaluate the performance of all the methods in terms of AUC and AP, which respectively represent the probability that a randomly selected unobserved link is more similar than a randomly selected non-existent one and the average precision.

3) Evaluation: In addition to the detection task, we also introduced the link prediction task in this experiment to measure the generation ability of the model. Therefore, we need to use AUC and AP to evaluate the results of the link prediction task.

AUC means the $Area\ Under\ the\ Curve$. It is a single scalar value representing the ROC performance. ROC is a receiver operating characteristics graph which visualize models based on their performance [40]. It is a two-dimensional graph in which the $True\ Positive\ Rate(Recall)$ is plotted on the Y axis and the $False\ Positive\ Rate$(the percentage of negatives incorrectly classified in total negatives) is plotted on the X axis. After drawing this curve, we can calculate the area of the area enclosed by the X axis. Since the AUC is the portion of the area under the ROC curve, it is always between 0 and 1.0, and it shows the relative tradeoff between true positives and false positives of our model.

AP is the $mean\ Average\ Precision$. It is the weighted average of the average accuracy (AP) of all classes from the classifier, which demonstrates the accuracy of the classifier in distinguishing all categories. In our experiments, it means the average accuracy of the transactions that should be reconstructed and those that should not appear.

In our experiments, we adapt the same classifier for both GAE and VGAE, so the AUC and AP reflect the ability of the model to learn the graph features.

4) Results: Firstly, we analyze the learning ability of the model for different graph structure. Based on the link prediction task, we conducted five independent repeated pieces of training and testing on GAE and VGAE on each dataset respectively to evaluate the generation ability of the model. The results are shown in Table. II and III.

The generation ability of GAE on two datasets:

The results of GAE shows a considerable difference between the two BTC datasets. On the BTC_{part} dataset, the $TestAUC$ can reach more than 70%, and $TestAP$ can reach

TABLE III
LINK PREDICTION RESULTS ON BTC DATASETS BY VGAE

Dataset	ValAUC		ValAP		TestAUC		TestAP	
	Records	Avg	Records	Avg	Records	Avg	Records	Avg
BTC	0.68094 0.69123 0.76783 0.71090 0.74800	0.71978	0.76166 0.77637 0.82798 0.78616 0.80037	0.79050	0.69366 0.67022 0.79411 0.69349 0.75869	0.72203	0.77548 0.74058 0.84731 0.76360 0.81986	0.78937
BTCpart	0.98108 0.97692 0.98848 0.96911 0.98092	0.97930	0.98336 0.98098 0.98950 0.97247 0.98254	0.98177	0.98631 0.98332 0.97438 0.97489 0.98290	0.98036	0.98563 0.98593 0.97587 0.97791 0.98501	0.98207

83% at most, which shows that GAE has learned the structural features of BTC transaction network and can reconstruct most of the records. However, the $TestAUC$ is less than 60%, and $TestAP$ is less than 70% on the BTC dataset. We believe that it is because the proportion of nodes and edges in the two datasets are different, which makes the transaction graph of BTC relatively sparse. More specifically, although the transaction graph of BTC can show the complete transaction information in a period of time, there are multiple connected components, and the connectivity between nodes is greatly reduced, so GAE can hardly learn the structural features of some edge nodes in the connected component well, then ignore these nodes and increase the reconstruction error; On the other hand, because the connection prediction task needs to drop out some edges firstly, GAE mistakenly assumes that there are dropout edges between independent connected components during reconstructing, and makes them up, which further reduces the prediction accuracy.

The generation ability of VGAE on two datasets:

For the results of VGAE, the generation ability of the model on each dataset has been significantly improved after adding variation. It is surprising that the $TestAP$ has increased by 24% on $BTCPart$ compared with GAE, while others have increased by about 12%. This fully shows that VGAE is much suitable than GAE for the task of transaction graph generation. We believe the latent space described by probability distribution takes into account the possibility of edge nodes and can be selected when reconstructing. At the same time, both the $TestAUC$ and $TestAP$ on BTC_{part} dataset can reach 98% strikingly, while the results on the BTC dataset are stable at 70%. It means that VGAE has almost memorized every node and the transaction it is associated with, then it has nearly reconstructed it perfectly on the transaction graph. We believe the noise caused by Gaussian distribution in the model makes it possible to reckon the distribution form of edges more completely when reconstructing the single connected component graph, which expands the possible range of transaction generation, so it can get better prediction accuracy.

Results of protecting with trained models:

After training the models, we can use them for covert transaction protection. Using two different data sets BTC and BTC_{part} and two different networks, we have learned a total of four different transaction graph generation models. We deploy these four networks to the existing 12 covert transaction subgraphs, reconstruct their transaction forms, and use the previously proposed covert transaction detection method based on graph measurement to detect these subgraphs again.

TABLE IV
THE UNITS AND HYPERPARAMETERS OF GAE & VGAE

Layers	Encoder	Decoder
1	GraphConvolution(6, 32, 0), ReLU	InnerProduct, Sigmoid
2	GraphConvolution(32, 16, 0), ReLU	InnerProduct, Sigmoid

Obviously, according to the protection result, the covert transaction graph almost no longer has the previous significant node distribution features. Under the previously determined scale, all covert transaction graphs can not be detected, so we need to reset these values. Taking the node degree with good performance as an example, we resample the variance values of normal transaction node degree and covert transaction node degree to determine the discrimination scale; Similarly, the previous offset is too strict for the reconstructed covert transaction graph, so we set all the offset to 1, which means that we increase the detection range. The test results are shown in the Fig. 8.

Compared with the previous unprotected results, the F1 score of the detection results under all scales decreased significantly, and the best one was only 0.33, which was in stark contrast to the previous results that could detect almost all covert transaction graphs. In the worst case, the detection F1 score is reduced to about 0.10, and the misjudged normal trading volume is greatly increased. This proves the effectiveness of our proposed method for covert transaction protection at the graph structure level.

From the perspective of the model, **the combination of the VGAE model and $BTCpart$ dataset shows the best protection effect in the link prediction experiment before combining different networks and data sets.** The F1 score under the three scales is reduced to less than 0.2, while the combination of the GAE model and BTC dataset reduces the F1 score to less than 0.4, which once again proves the powerful generation ability of the VGAE model, And the positive impact of connectivity transaction graph as a data set on the performance of the model in our graph-based method.

C. Case Study

To further illustrate the protection mechanism of the proposed graph generation model for covert transactions,

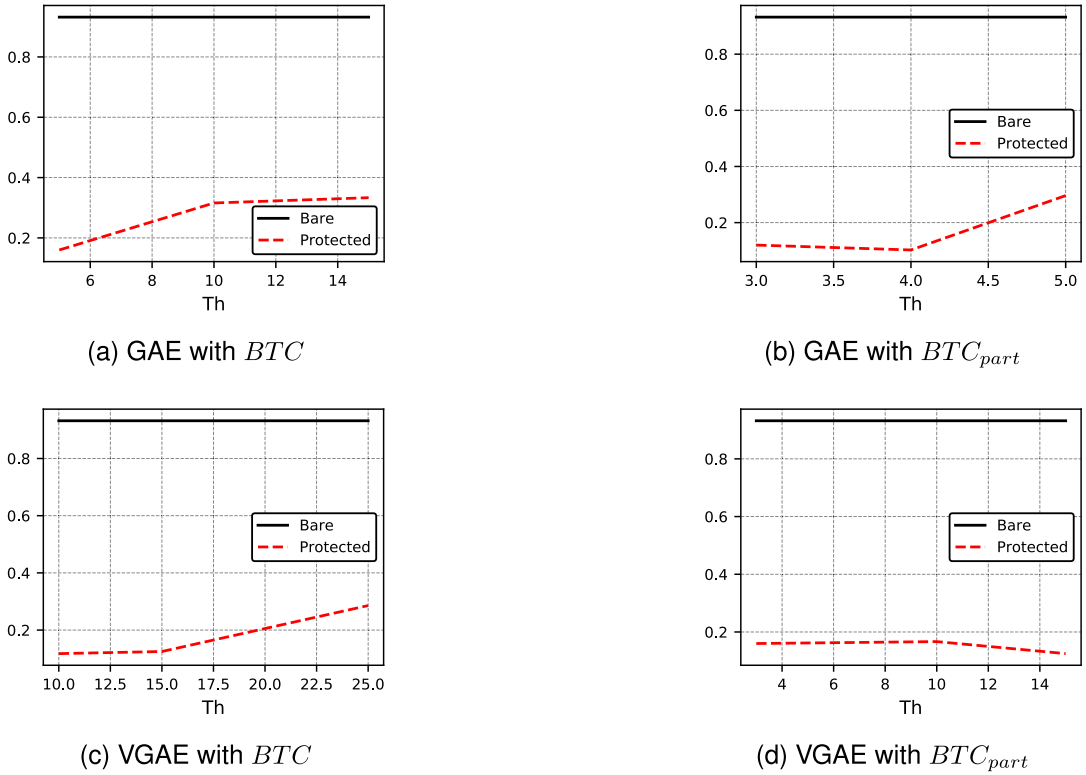


Fig. 8. Protection Results for different models with $\epsilon = 1$. The accuracy of using graph structure metrics to detect covert transactions after protection becomes very low, which is in contrast to the results when they are bare.

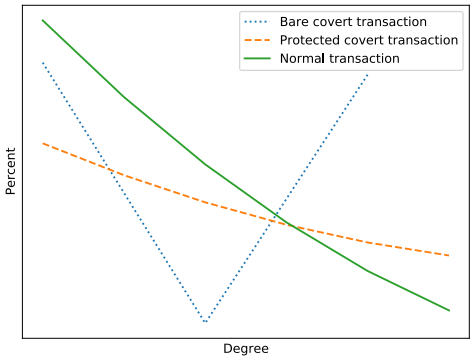


Fig. 9. Degree distribution in different graph. The node degree distribution of the protected covert transaction is significantly different from that of the bare one, and becomes closer to the distribution of normal transactions.

we choose a covert transaction subgraph as a case to analyze the changes before and after protecting.

For a planned covert transaction implementation, we extract the wallet addresses list of participants and the proposed transaction process to construct a covert transaction subgraph, and get the adjacency matrix of this graph. We then use the adjacency matrix as input to our protection scheme and reconstruct this graph using the trained VGAE model to obtain the reconstructed graph adjacency matrix. Next, the covert transaction graph is reduced based on the reconstructed adjacency matrix and the protected covert transaction records are obtained using Algorithm. 1, where the wallet addresses in the transaction records have been updated. Finally, it is sufficient for the participants to perform covert transactions based on the wallet addresses in the protected transaction records.

Fig. 9 shows the distribution of node degrees. We can see that the node distribution of the bare covert transaction graph is highly similar to the distribution we mentioned in Fig. 1, which can be divided into two opposing extremes. And the node degree distribution of normal transactions follows a significant downward curve. Therefore, our analytics-driven method can easily distinguish these two graphs. However, with the help of graph generation model to reconstruct the covert transaction graph, its node distribution is quite different from that before, as shown in the figure. The shape of this curve is closer to the distribution of normal transactions, and it is more difficult for our detection method to identify it.

More specifically, we also calculated the Jensen-Shannon (JS) divergence between node degree distributions under these three conditions, and quantitatively described the differences between them. JS divergence is a variation of KL divergence and usually used to describe the similarity between two distributions, and it can be obtained based on the KL divergence, where KL divergence we can calculate as shown in Eq.(8).

$$JS(P||Q) = \frac{1}{2}KL(P||\frac{P+Q}{2}) + \frac{1}{2}KL(Q||\frac{P+Q}{2}) \quad (13)$$

The value varies from 0 to 1 and the closer the JS divergence between two distributions is to 0, the more similar the two distributions are. As it shown in the Table V, we first randomly select two normal transaction graphs and measure the JS divergence between them as a baseline, and the JS divergence between the bare covert transaction and the normal transaction differs sharply from it. With the help of our protection method, the JS divergence between the covert transaction and the normal transaction reduces to one-tenth of the original

TABLE V
JENSEN-SHANNON DIVERGENCE OF DEGREE DISTRIBUTIONS

P	Q	$JS(P\|\|Q)$
Normal	Another Normal	0.0117
Bare covert	Normal	0.2193
Bare covert	Protected covert	0.1670
Protected covert	Normal	**0.0181**

value, and becomes very close to the result between two normal transaction graphs, which prove that our method can effectively protect covert transactions from being detected.

VI. CONCLUSION

The development of covert transactions depends on the transactions between different addresses. Firstly, we illustrate the exposure risk of multiple covert transactions in the cluster structure from the perspective of graph structure, and propose a detection method based on graph structure measurement, which proves that the current covert transaction methods lacks the protection of the association relationship between multiple covert transactions. Then we propose a covert transaction protection mechanism based on the graph generation model, which reconstructs the covert transaction graph by learning the features of normal transactions, to improve the similarity between its clustering structure and normal transactions. The protection experiments and secondary detection on real transaction data sets prove the effectiveness of the proposed methods. In the future, we will consider using other graph embedding methods to search for the best architecture for the proposed model. We expect that the generative model can precisely capture data dependence between transaction attributes and address links for any covert method with minimal human efforts.

REFERENCES

[1] Z. Zhang et al., "The research on covert communication model based on blockchain: A case study of Ethereum's whisper protocol," in *Proc. Int. Conf. Frontiers Cyber Secur.* Singapore: Springer, 2020, pp. 215–230.

[2] S. Yan, X. Zhou, J. Hu, and S. V. Hanly, "Low probability of detection communication: Opportunities and challenges," *IEEE Wireless Commun.*, vol. 26, no. 5, pp. 19–25, Oct. 2019.

[3] J. Bonneau, A. Miller, J. Clark, A. Narayanan, J. A. Kroll, and E. W. Felten, "SoK: Research perspectives and challenges for Bitcoin and cryptocurrencies," in *Proc. IEEE Symp. Secur. Privacy*, May 2015, pp. 104–121.

[4] M. Li et al., "Astraea: Anonymous and secure auditing based on private smart contracts for donation systems," *IEEE Trans. Dependable Secure Comput.*, vol. 20, no. 4, pp. 3002–3018, Jul./Aug. 2023.

[5] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," bitcoin.org, Tech. Rep. 1, 2008.

[6] M. Li, Y. Chen, C. Lal, M. Conti, M. Alazab, and D. Hu, "Eunomia: Anonymous and secure vehicular digital forensics based on blockchain," *IEEE Trans. Dependable Secure Comput.*, vol. 20, no. 1, pp. 225–241, Jan. 2023.

[7] M. Shen, X. Tang, L. Zhu, X. Du, and M. Guizani, "Privacy-preserving support vector machine training over blockchain-based encrypted IoT data in smart cities," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 7702–7712, Oct. 2019.

[8] S. T. Ali, P. McCorry, P. H.-J. Lee, and F. Hao, "ZombieCoin 2.0: Managing next-generation botnets using Bitcoin," *Int. J. Inf. Secur.*, vol. 17, no. 4, pp. 411–422, Aug. 2018.

[9] A. Kurt, E. Erdin, M. Cebe, K. Akkaya, and A. S. Uluagac, "LNBot: A covert hybrid botnet on Bitcoin lightning network for fun and profit," in *Proc. Eur. Symp. Res. Comput. Secur.* Cham, Switzerland: Springer, 2020, pp. 734–755.

[10] J. Yin, X. Cui, C. Liu, Q. Liu, T. Cui, and Z. Wang, "CoinBot: A covert botnet in the cryptocurrency network," in *Proc. Int. Conf. Inf. Commun. Secur.* Cham, Switzerland: Springer, 2020, pp. 107–125.

[11] M. Weber et al., "Anti-money laundering in Bitcoin: Experimenting with graph convolutional networks for financial forensics," 2019, *arXiv:1908.02591*.

[12] S. Sayadi, S. B. Rejeb, and Z. Choukair, "Anomaly detection model over blockchain electronic transactions," in *Proc. 15th Int. Wireless Commun. Mobile Comput. Conf. (IWCMC)*, Jun. 2019, pp. 895–900.

[13] A. Fionov, "Exploring covert channels in Bitcoin transactions," in *Proc. Int. Multi-Conf. Eng., Comput. Inf. Sci. (SIBIRCON)*, Oct. 2019, pp. 59–64.

[14] M. Xu, H. Wu, G. Feng, X. Zhang, and F. Ding, "Broadcasting steganography in the blockchain," in *Proc. Int. Workshop Digit. Watermarking*. Cham, Switzerland: Springer, 2019, pp. 256–267.

[15] L. Zhang, Z. Zhang, Z. Jin, Y. Su, and Z. Wang, "An approach of covert communication based on the Ethereum whisper protocol in blockchain," *Int. J. Intell. Syst.*, vol. 36, no. 2, pp. 962–996, Feb. 2021.

[16] M. Bartoletti, B. Pes, and S. Serusi, "Data mining for detecting Bitcoin Ponzi schemes," in *Proc. Crypto Valley Conf. Blockchain Technol. (CVCBT)*, Jun. 2018, pp. 75–84.

[17] A. Sharma and A. Bhatia, "Bitcoin's blockchain data analytics: A graph theoretic perspective," 2020, *arXiv:2002.06403*.

[18] J. C. Wray, "An analysis of covert timing channels," *J. Comput. Secur.*, vol. 1, nos. 3–4, pp. 219–232, Oct. 1992.

[19] Z. T. T. Zhang and Q. Wu, "Bitcoin blockchain based information convert transmission," *Chin. J. Netw. Inf. Secur.*, vol. 7, no. 1, p. 84, 2021. [Online]. Available: http://www.infocomm-journal.com/cjnis/CN/abstract/article_171115.shtml

[20] S. Cabuk, C. E. Brodley, and C. Shields, "IP covert timing channels: Design and detection," in *Proc. 11th ACM Conf. Comput. Commun. Secur.*, Oct. 2004, pp. 178–187.

[21] J. Partala, "Provably secure covert communication on blockchain," *Cryptography*, vol. 2, no. 3, p. 18, Aug. 2018.

[22] S. Song and W. Peng, "BLOCCE+: An improved blockchain-based covert communication approach," *J. Chongqing Univ. Technol. Natural Sci.*, vol. 34, no. 9, pp. 238–244, 2020.

[23] S. Liu et al., "Whispers on Ethereum: Blockchain-based covert data embedding schemes," in *Proc. 2nd ACM Int. Symp. Blockchain Secure Crit. Infrastruct.*, Oct. 2020, pp. 171–179.

[24] J. Tian, G. Gou, C. Liu, Y. Chen, G. Xiong, and Z. Li, "DLchain: A covert channel over blockchain based on dynamic labels," in *Proc. Int. Conf. Inf. Commun. Secur.* Cham, Switzerland: Springer, 2019, pp. 814–830.

[25] H. Cao et al., "Chain-based covert data embedding schemes in blockchain," *IEEE Internet Things J.*, vol. 9, no. 16, pp. 14699–14707, Aug. 2022.

[26] A. A. Giron, J. E. Martina, and R. Custódio, "Steganographic analysis of blockchains," *Sensors*, vol. 21, no. 12, p. 4078, Jun. 2021.

[27] S. Lerner. (2019). *A New Mystery about Satoshi Hidden in the Bitcoin Block-Chain*. [Online]. Available: https://bitslog.com/2013/09/03/new-mystery-about-satoshi/

[28] G. Kappos, H. Yousaf, M. Maller, and S. Meiklejohn, "An empirical analysis of anonymity in Zcash," in *Proc. 27th USENIX Secur. Symp. (USENIX Security)*, 2018, pp. 463–477.

[29] A. Kharraz et al., "Outguard: Detecting in-browser covert cryptocurrency mining in the wild," in *Proc. World Wide Web Conf.*, May 2019, pp. 840–852.

[30] H. N. C. Neto, M. A. Lopez, N. C. Fernandes, and D. M. F. Mattos, "MineCap: Super incremental learning for detecting and blocking cryptocurrency mining on software-defined networking," *Ann. Telecommun.*, vol. 75, nos. 3–4, pp. 121–131, Apr. 2020.

[31] A. Gangwal, S. G. Piazzetta, G. Lain, and M. Conti, "Detecting covert cryptomining using HPC," in *Proc. Int. Conf. Cryptol. Netw. Secur.* Cham, Switzerland: Springer, 2020, pp. 344–364.

[32] Y. Lan, F. Zhang, and T. Haibo, "Using Monero to realize covert communication," *J. Xidian Univ.*, vol. 47, no. 5, pp. 19–27, 2020.

[33] F. Franzoni, I. Abellan, and V. Daza, "Leveraging Bitcoin Testnet for bidirectional botnet command and control systems," in *Proc. Int. Conf. Financial Cryptogr. Data Secur.* Cham, Switzerland: Springer, 2020, pp. 3–19.

[34] Z. Lejun et al., "A covert communication method using special Bitcoin addresses generated by Vanitygen," *Comput., Mater. Continua*, vol. 65, no. 1, pp. 597–616, 2020.

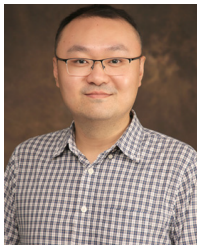
[35] (2020). *Blockcypher's API Documentation*. [Online]. Available: https://www.blockcypher.com/dev/bitcoin/
[36] S. Delgado-Segura, C. Pérez-Sola, G. Navarro-Arribas, and J. Herrera-Joancomartí, "Analysis of the Bitcoin UTXO set," in *Proc. Int. Conf. Financial Cryptography Data Secur.* Cham, Switzerland: Springer, 2018, pp. 78–91.
[37] T. N. Kipf and M. Welling, "Variational graph auto-encoders," 2016, *arXiv:1611.07308*.
[38] F. Chen, Y.-C. Wang, B. Wang, and C.-C.-J. Kuo, "Graph representation learning: A survey," *APSIPA Trans. Signal Inf. Process.*, vol. 9, no. 1, pp. 1–21, 2020.
[39] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," 2013, *arXiv:1312.6114*.
[40] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognit. Lett.*, vol. 27, no. 8, pp. 861–874, Jun. 2006.

Zhenyu Guo is currently pursuing the Ph.D. degree with the School of Computer Science, Beijing Institute of Technology, Beijing, China. His current research interests include the development of algorithms for deep learning with application to cyberspace security.

Xin Li received the Ph.D. degree in computer science from Hong Kong Baptist University in 2009. She is currently an Associate Professor with the School of Computer Science, Beijing Institute of Technology, China. Her research interests include the development of algorithms for representation learning, reasoning under uncertainty, and machine learning with applications to graph data mining, recommender systems, and robotics.

Jiamou Liu has been a Senior Lecturer with The University of Auckland, New Zealand, since 2016. Prior to his current role, he was a Senior Lecturer with the Auckland University of Technology (AUT) and a Research Associate with Leipzig University. His expertise lies in artificial intelligence, where he has made contributions toward graph data mining, machine learning, and multi-agent systems. He has published over 120 published papers across venues, such as AAAI, IJCAI, WWW, and SIGIR.

Zijian Zhang (Senior Member, IEEE) received the Ph.D. degree from the School of Computer Science and Technology, Beijing Institute of Technology. He was a Visiting Scholar with the Department of Computer Science and Engineering, State University of New York at Buffalo, in 2015. He is currently an Associate Professor with the School of Cyberspace Science and Technology, Beijing Institute of Technology. His research interests include the design of authentication and key agreement protocol and analysis of entity behavior and preference.



Meng Li (Senior Member, IEEE) received the Ph.D. degree in computer science and technology from the School of Computer Science and Technology, Beijing Institute of Technology (BIT), China, in 2019. He is currently an Associate Professor and the Dean Assistant with the School of Computer Science and Information Engineering, Hefei University of Technology (HFUT), China. He is also a Post-Doctoral Researcher with the Department of Mathematics and HIT Center, University of Padua, Italy, where he is with the Security and PRIvacy Through Zeal (SPRITZ) Research Group led by Prof. Mauro Conti. He was sponsored by the ERCIM 'Alain Bensoussan' Fellowship Programme from October 2020 to March 2021 to conduct Post-Doctoral Research supervised by Prof. Fabio Martinelli at CNR, Italy. He was also sponsored by the China Scholarship Council (CSC) from September 2017 to August 2018 for a joint Ph.D. study supervised by Prof. Xiaodong Lin in the Broadband Communications Research (BBCR) Laboratory with the University of Waterloo and Wilfrid Laurier University, Canada. His research interests include security, privacy, fairness, applied cryptography, cloud computing, edge computing, blockchain, and vehicular networks. In this area, he has published more than 60 papers in international peer-reviewed transactions, journals, and conferences, including IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, IEEE/ACM TRANSACTIONS ON NETWORKING, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, *ACM Transactions on Database Systems*, IEEE TRANSACTIONS ON SERVICES COMPUTING, IEEE TRANSACTIONS ON SMART GRID, IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT, IEEE TRANSACTIONS ON NETWORK SCIENCE AND ENGINEERING, IEEE TRANSACTIONS ON GREEN COMMUNICATIONS AND NETWORKING, COMST, MobiCom, ICICS, SecureComm, TrustCom, and IPCCC. He is an Associate Editor of IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT, and IEEE INTERNET OF THINGS JOURNAL.

Jingjing Hu received the Ph.D. degree in computer science from the Beijing Institute of Technology, Beijing, China. She is currently an Associate Professor with the School of Computer Science, Beijing Institute of Technology. Her research interests include service computing, web intelligence, and information security.

Liehuang Zhu (Senior Member, IEEE) received the M.S. degree in computer science from Wuhan University, Wuhan, China, in 2001, and the Ph.D. degree in computer science from the Beijing Institute of Technology, Beijing, China, in 2004. He is currently a Full Professor with the School of Cyberspace Science and Technology, Beijing Institute of Technology. His research interests include data security and privacy protection, blockchain applications, and AI security. He has authored more than 150 journal and conference papers in these areas. He has served as the Program Co-Chair of MSN 2017, IWWS 2018, and INTRUST 2014. He received the Best Paper Award from the IEEE/ACM IWQoS 2017, the IEEE TrustCom 2018, and the IEEE IPCCC 2014. He is an Associate Editor of IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, *IEEE Network*, and IEEE INTERNET OF THINGS JOURNAL. He was a Guest Editor of the Special Issue of IEEE WIRELESS COMMUNICATIONS and IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS.