International Journal of Digital Crime and Forensics
Volume 13 • Issue 6

Hidden Service Circuit Reconstruction Attacks Based on Middle Node Traffic Analysis

Yitong Meng, State Key Laboratory of Mathematical Engineering and Advanced Computing, China
https://orcid.org/0000-0003-2736-232X

Jinlong Fei, State Key Laboratory of Mathematical Engineering and Advanced Computing, China

ABSTRACT

Traffic analysis is widely considered an attack posing a threat to anonymity of the communication and may reveal the real identity of the users. In this paper, a novel anonymous circuit reconstruction attack method that correlates the circuit traffic is proposed. This method then reconstructs a complete communication tunnel using the location of middle nodes found between the hidden and client services. The attack process includes independent determination of the location of the malicious nodes. A traffic correlation framework of AutoEncoder + CNN + BiLSTM is established, based on the generative adversarial networks (GAN) model. BiLSTM applies the packet size and packet interval features of bidirectional traffic and combines the reconstruction loss function with the discrimination loss function to achieve correlated traffic evaluation. After balancing the reconstruction loss and discrimination loss scores, the simulation results confirm that the identification performance of the proposed system is higher than the advanced models.

KEYWORDS

Circuit Reconstruction, GAN, Middle Node, Reconstruction Loss Function

1. INTRODUCTION

By increasing the amount of information available on the Internet, the technology for censoring network information is also improving. Currently, anonymous network communication protocols have been developed to meet the needs for privacy data protection. An example is the onion router (Tor)(Dingledine et al., 2004) which is one of the most popular anonymity networks. Tor embeds data in multiple layers of onion encryption effectively preventing attackers from accessing the identity of the communication terminals and the data. This has attracted many attackers and researchers to propose diversified target attacking approaches centering on the traffic analysis (Biryukov et al., 2013; Galteland & Gjøsteen, 2018; Jansen et al., 2018; Kwon et al., 2015). The core objective is to achieve de-anonymization attacks on the communication users by observing the pattern of the encrypted traffic around the onion nodes. Nevertheless, for each attack method, there are several traffic countermeasures such as traffic obfuscation and node protection (Imani et al., 2018; Johnson et al., 2017) which challenge traditional attack methods. In addition to the communication terminals,

DOI: 10.4018/IJDCF.288548

anonymous transmission circuits are also an important part of the communication process. Hence the analysis of transmission circuits creates serious threats compared to attacking the communication terminals. Galteland & Gjøsteen (2018) shows that by analyzing and constructing the transmission circuit any accessing behavior at the communication terminal becomes transparent hence facilitates de-anonymization attacks.

Conventional onion circuits are designed with a cooperative transmission mode among three nodes, where the anonymization effect only involves the communication sender. Subsequently, the proposed hidden service onion circuit can anonymize both the sender and receiver by splicing two sets of regular circuits. In this paper, a novel attack technique of middle node traffic analysis is proposed based on the idea of reconstructing the hidden service communication circuits by controlling the circuit nodes. In this technique, The primary goal is to find the location of the middle node found between the hidden service and the client service. The internal traffic on both sides is then correlated and analyzed and used to accurately reconstruct the complete communication circuits within the network. The initial location determination of the middle node is critical. This is because the middle node makes the circuits transparent, and further evades the existing protection for the critical nodes. Hence knowing the middle node enables attackers to track the entry nodes at both ends of the communication and possess attacking capability using minimal attack resources.

In this paper, referring to the basic conditions derived from the node location determination results, the traffic correlation target is divided into client area traffic and hidden service area traffic. The advantage of this approach is that the transmission features of the unidirectional data within a circuit are independent of and interrelated with each other. In addition, it is shown in the previous works (Guan et al., 2020) that traffic correlation attacks can be performed in a large traffic environment under continuous observation. However, they still have deficiencies such as poor noise immunity, long observation time of the traffic, and high demand for positive and negative sample datasets. Moreover, it is necessary to capture large traffic label data in advance and for over a long time and also use pre-processing models to eliminate the interference of noise in the identification of unsmooth data.

The Generative Adversarial Networks (GAN) (Goodfellow et al., 2014) network can meet the needs of reconstructing positive and negative sample data sets. Its internal network of the is a typical unsupervised model that uses synthetic data generated after repeated game learning to improve the model's judgment on synthetic and real data. The GAN is applied to intrusion detection and image recognition (Akcay et al., 2019; de Araujo-Filho et al., 2020) and provides high robustness and transferability. In particular, the proposed methodology where the packet size and packet interval within the traffic are chosen as the feature values. Convolutional neural network (CNN) models can use convolutional layers to quickly extract features, so the packet size feature vector can be extracted. The Bi-directional Long Short-Term Memory (BiLSTM) uses the chain memory structure to have the ability to learn the characteristic distribution of the time correlation of data packets. Both CNN and BiLSTM train a suitable comparative model of the target loss function. To measure the practical benefits of evaluating the traffic correlation, reconstruction losses are added to compensate for the comparison results of the synthetic traffic with the evaluated traffic.

In summary, the following contributions are made in this paper.

- A novel hidden service circuit reconstruction attack is proposed based on middle node traffic to obtain the key node location identification and correlation evaluation of traffic on both sides. The proposed attack successfully addresses the difficult issue of circuit reconstruction.
- The middle node locations are then trained and classified by using a node-level attack without the influence of the padding mechanism. A combined GAN model of AutoEncoder+CNN+BiLSTM is then constructed to compare the loss values of feature vectors between the reconstructed samples and the real samples. This model further distinguishes the correlated and non-correlated traffic according to the loss values.

- An effective correlation evaluation algorithm is proposed which combines the node location probability with the scores of reconstruction and discrimination losses within the correlated traffic to achieve a comprehensive judgment.
- The systematic experimental evaluation of the attack model indicates that the F1 score of the node location identification is close to 1 point, while the correlation precision and recall of 900 packets are 0.896 and 0.947, respectively, which overperform the advanced correlation models.

2 BACKGROUND

2.1 Hidden Services

The hidden service protocol (Biryukov et al., 2013) was originally designed to protect the real identity of the server and to make the Tor client and server anonymous to each other. The hidden service communication architecture is a multifunctional combination of circuits derived from telescopic circuits. The rendezvous node function circuit is responsible for the secure transmission of the encrypted DATA between the client and the server. This is performed through multiple communication streams attached to the circuit and a fixed 512-byte cell for all load packets in each stream. The rendezvous function circuit is the focus of this paper and the basic composition and mechanism details are shown in Figure 1.

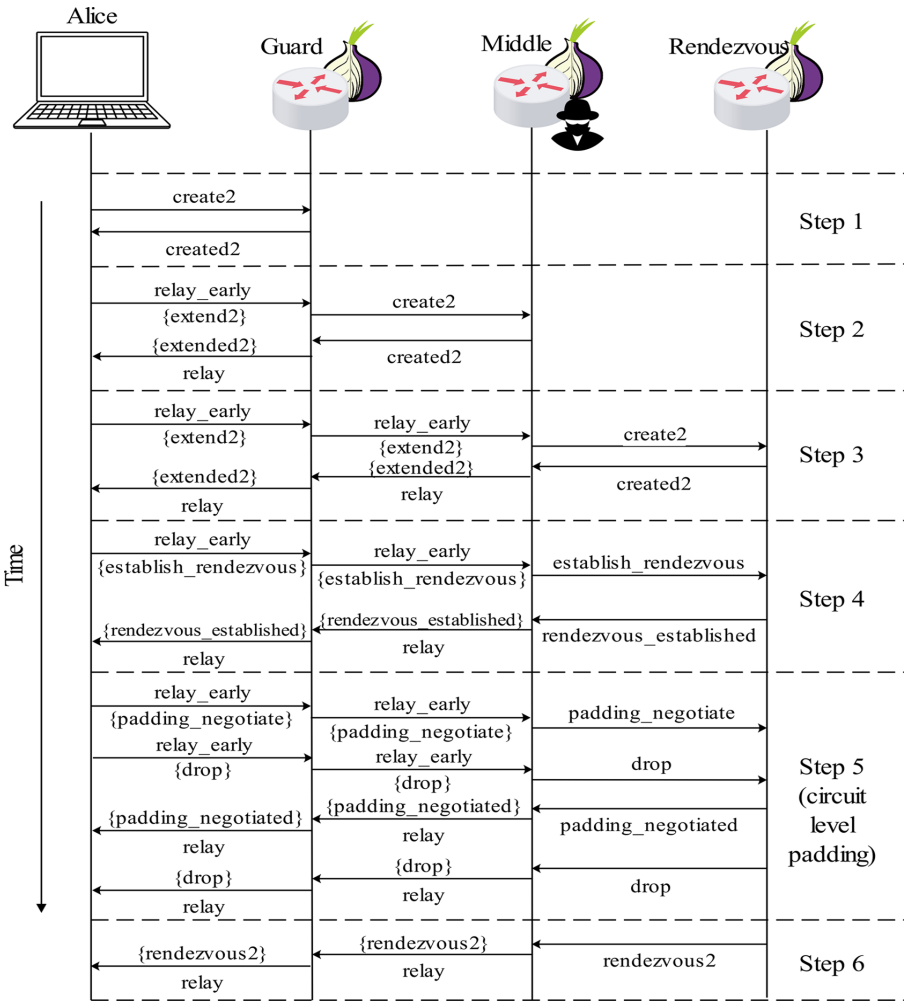
Figure 1. (a) shows the circuit transmission steps from the client proxy to the rendezvous node operated by Alice. Figure 1. (b) also illustrates the circuit transmission steps from the server proxy to the rendezvous node operated by Bob. It is seen that the encapsulation form of the Cell determines the communication location of each relay node in the circuit. A brief analysis of the creation steps (steps 1 to 3) is conducted, where the rendezvous circuit creation is different from the normal circuit creation. However, Tor recognizes the load features exposed by the circuit creation. Starting from Tor-0.4.1.5, attempts have been made to pad the hidden server and client circuits by adding circuit-level padding units at the start of its rendezvous node circuits (step 5). Excluding the server-side padding at this stage is to reduce the circuit overhead. Due to the addition of the padding mechanism, many traditional traffic analysis methods are no longer applicable to the latest version of Tor. Therefore, the related features of circuit structural sequence are avoided, and instead, the node-level features are used to accurately determine the location of a node in a circuit.

2.2 Middle Nodes

The telescopic circuit in Tor only has the neighboring relay nodes that are mutually aware of the forwarding target of the data. Therefore, three nodes are tacitly set as a reasonable observation window. In this paper, a complete reconstruction of the hidden service communication circuit (6 relay nodes) requires a maximum of 3 nodes (including an entry node) and a minimum of 2 middle nodes. For empirical reasons, the two middle nodes are chosen which consume fewer resources. Figure 2 illustrates a reconstruction attack scenario for a complete communication circuit using middle node locations and passive observation of the traffic patterns. The attackers control the middle nodes Client (C-Middle) and Hidden Service (HS-Middle) which are close to the client, Alice, and server, Bob, respectively. The attackers then record and analyze the features of the circuit flow inside the nodes. Note that the node-level attack at that time is passive, thus the original traffic transmission is not disrupted or modified. In cases where several rendezvous circuits are successfully correlated by an attacker-controlled middle node, the previous-hop node can be labeled as a potential entry node for anonymous users and hidden services.

Recently researchers recognized the guarded entry node as a critical location in the circuit and developed several well-established defense schemes (Hayes & Danezis, 2015; Imani et al., 2018) to reduce the possibility of compromising the entry node. The rendezvous circuit has no real exit node and contains multiple internal middle nodes. This effectively removes the limitation of the selection

Figure 1a. The basic communication architecture of the hidden service rendezvous circuit (arrows indicate the load commands): Circuit construction from the client to rendezvous node



of guarded entry nodes. The attackers need to expend considerable resources to control the internal middle nodes to perform a node-level attack. Therefore, the bandwidth consumption required should be further explored first. Using the consensus weight of the nodes $C\left(w\right)$ and the published bandwidth b provided by the consensus document as the base data, analysis shows a linear correlation. As shown in Equation (1), the consensus weight is calculated by multiplying the published bandwidth by the ratio of the actual measured average traffic R to the overall network average traffic Z. It is seen that the weight value fluctuated with the state of the website and that the probability $H_{\hat{G}}$ of controlling the selection of guarded entry node is much higher than that of the middle node, $H_{\hat{M}}$ [Equation (2)].

$$C\left(w\right)=b\frac{R}{Z} \tag{1}$$

Figure 1b. The basic communication architecture of the hidden service rendezvous circuit (arrows indicate the load commands): Circuit construction from the server to rendezvous node

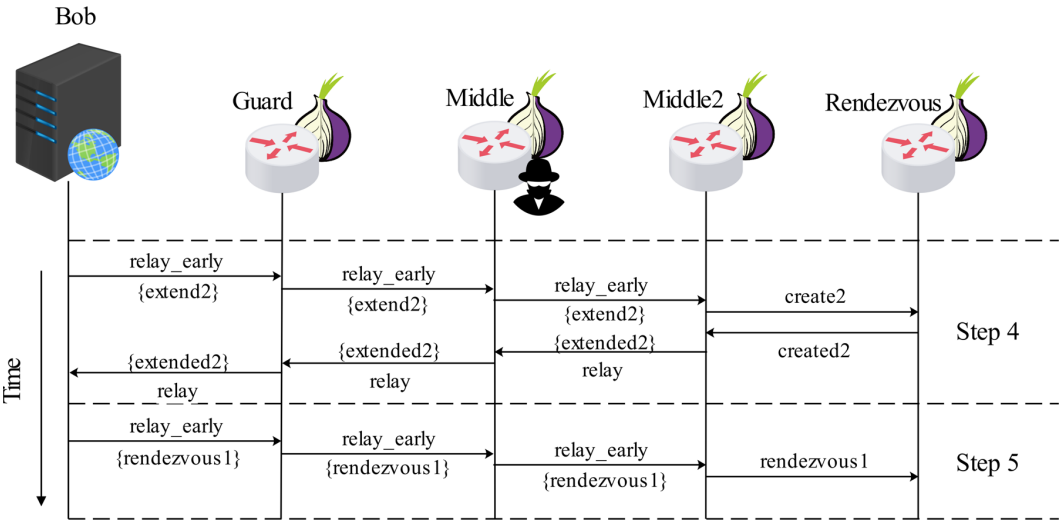
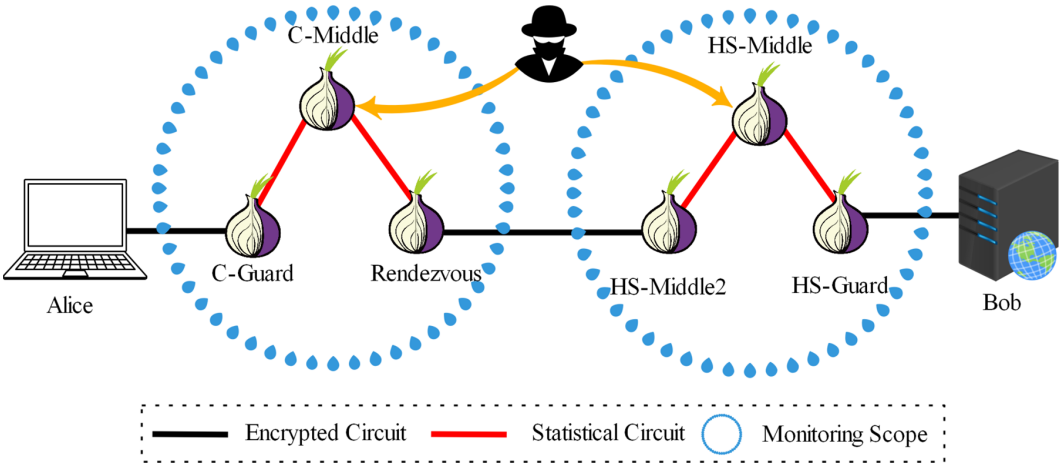


Figure 2. The middle node circuit reconstruction attack scenario



$$H_{\hat{M}} = \frac{C(w)}{C(w)_M} < H_{\hat{G}} = \frac{C(w)}{C(w)_G} \tag{2}$$

To fit the attack scenario, Equation (3) manifests that the attackers continuously correlate the forwarding traffic of the two middle nodes, $\hat{M}_1$ and $\hat{M}_2$. The attackers also record the number of rendezvous communication circuits found in the internal circuit c after r attack attempts. For example, the attackers control the middle nodes with the top 1% weight values. Amongst the current 10,000 rendezvous circuits, the number of circuits observed in 10 attack attempts is nearly 200. This

Table 1. The advanced circuit traffic analysis attack methods

Author	Year	Base model	Attack method	Observation location	Accuracy
Kwon *et al.*	2015	Decisional Tree	Hidden service circuit fingerprint	Entry	98%
Galteland & Gjøsteen	2018	Log-normal distribution	Circuit fingerprint, traffic correlation	Entry, middle, exit	65%
Platzer *et al.*	2020	Linear correlation	Circuit fingerprint	Introduction	96.45%
Biryukov *et al.*	2013	Linear correlation	Traffic correlation	Rendezvous, middle	100%
Jansen *et al.*	2018	Random forests	Circuit fingerprint	Middle	92%
Nasr *et al.*	2018	CNN	Traffic correlation	Entry, exit	96%
Guan *et al.*	2020	Autoencoder	Traffic correlation	Entry, exit	97%
The paper method	2021	SVM+GAN	Hidden service circuit fingerprint, traffic correlation	Middle	94.43%

is much larger than the 73 observed for the guarded entry node under the same conditions. The above results suggest that controlling the middle nodes is beneficial to the circuit reconstruction attacks.

$$P\left(r\right) = c \cdot H_{\hat{M}_1}\left(1 - \left(1 - H_{\hat{M}_2}\right)^r\right) \tag{3}$$

3 RELATED WORK

The summary of the related works is presented in Table 1.

3.1 Circuit Traffic Analysis

The transmission state is blurred using the hierarchical interaction of data. To guarantee the low delay of data interaction, only a fixed cell size is selected in Tor to resist partial traffic analysis. Therefore, some exposed features can still be easily used by the attackers for reliable passive traffic analysis attacks.

Kwon *et al.* (2015) propose a method to reveal the communication state of the telescopic circuits and devise circuit-level fingerprint attacks for hidden services. In their method, the attackers need to extract circuit-level features to meet the attack conditions. To distinguish multi-type communication circuits in the hidden services, the attackers determine four circuit states based on the circuit purposes and special nodes. These circuits are separately identified by the decision tree algorithm with an accuracy higher than 98%.

With the objective of circuit reconstruction, Galteland and Gjøsteen (2018) suggested a method to segment the circuits as per the AS domain and Internet Exchange Point (IXP). In their study, attackers first observe the boundary traffic of each domain and analyze the time difference between the incoming and outgoing traffic in a single domain. Attackers then conduct a random combination of the packets in multiple domains according to the circuit location and reconstruct multiple sets of adjacent telescopic communication circuits. This reveals the anonymous connection between two communication parties. However, the problem that this brings is that it relies heavily on the stability

of the observation time, and the distribution of multiple jurisdictions is also not suitable for realistic circuit reconstruction attack scenarios. This article only needs traffic analysis of two nodes to achieve the basic goal.

Recently, Platzer *et al.* (2020) proposed a kind of introduction circuit attack based on statistical analysis which mainly uses the introduction circuits by controlling multiple key nodes and threatens the real location of the hidden services. The authors take the upcircuit & downcircuit loads during the creation and transmission of the introduction circuit as the basic features to judge the specific circuit location of the hidden services at the malicious nodes. They then allow the malicious client to send the loads with a fixed time pattern along the circuit to track and identify the special traffic from the malicious entry node to determine the real location of the hidden services. However, it is worth noting that neither Platzer nor Kwon's attack modes can adapt to the new version detection of the loading-padding cell mechanism, so this article needs to re-evaluate the effective data load characteristics.

This paper focuses on traffic attacks based on the middle nodes, and the primary objective of relevant studies is to determine whether the traffic acquisition location is the ideal location of communication circuits (Tor internal transmission location). Biryukov *et al.* (2013) examined the attack capability of the middle nodes by studying the defects of the hidden service protocols. They then proposed an effective attack method, where the attacker controls two non-exit nodes. During such attacks, a transmission tunnel is forcedly established between the hidden service and the rendezvous node controlled by the attacker. Passing the transmission traffic through the malicious middle nodes enables the malicious node to identify the special signature generated by the rendezvous node. If the detection conditions are met, the attacker immediately knows that the malicious middle node is in the target circuit. Therefore, the previous-hop node is marked as a potential guard node of the hidden service.

Jansen *et al.* (2018) also show that the attackers can monitor a wide range of targets such as circuit purpose, location, and website fingerprinting identification based on the middle node traffic instead of the network edge traffic. The authors then design the Onionpop classification channel which obtains the purpose and location of the current circuit of nodes and further detects the popularity of the hidden services. In this paper, the determination of the circuit location of middle nodes is also partially based on the Onionpop identification features, but Onionpop is still tested on the old version of Tor. Therefore there is a certain identification gap and necessity for further investigations.

3.2 Deep Learning Analysis

In recent years, deep learning has been widely used in website fingerprinting adversary, protocol traffic classification, traffic correlation analysis, and other mainstream targets in various research fields of Tor. Using deep learning significantly important results were also achieved both theoretically and practically. Compared with the traditional machine learning algorithms, the neural network is more advantageous due to its capability of automatic learning of the correlative features of data vectors by analyzing the potential correlation between input and output. Nevertheless, to obtain high identification accuracy, large amounts of labeled data are required for training and verification of the model. Since the neural network model has the most advanced feature set and complex classifier, the latest deep learning model is selected for comparison with the attack model proposed in this article.

Using the features of the accumulated data packets in large traffic samples, the specific locations of the incoming and outgoing ends in the same path can be inferred during traffic correlation attacks. Zhu *et al.* (2010) verified that the efficiency of correlation attacks largely relies on the number of observed traffic packets. Nasr *et al.* (2018) proposed the DeepCorr deep learning model and showed through experiments that the correlation accuracy of 96% can be maintained for a long time only when 900 packets are observed. This is superior to that of the traditional machine learning attack model under the same conditions. In DeepCorr, the deep learning algorithm is used to break through the barriers of identifying the traffic correlation targets. The model learns the function modes of upcircuit & downcircuit traffic. In addition, the packet size and interval are combined into a two-dimensional

matrix for multi-layer convolution and then the correlation probability of a group of node traffic is provided as the output. Nonetheless, the true positive rate drops from 90% to 50% if the DeepCorr encounters traffic obfuscation noise.

To address the obfuscation noise issue mentioned above, Guan *et al.* (2020) designed the ResTor traffic pre-processing model and conducted noise reduction before recording the upcircuit & downcircuit features of the correlated traffic. This reduces the possibility of packet misplacement and circuit change deviation caused by noise. In ResTor, a stacked autocoder is used to smooth out the cumulative byte sequences and optimize the output vector to make it as close as possible to the input vector. The experiments show that compared with DeepCorr, ResTor configured with the cosine distance algorithm has lower computational complexity and higher identification accuracy. However, ResTor still needs sufficient label data to achieve the target fitting effect.

This article will remove the traditional supervised learning model and use unsupervised learning to overcome the inconvenience of obtaining data notes. The model separately judges the characteristics of data packet size and packet interval time, and better mining hidden features from a practical perspective. Experimental results proves that the dual recognition network architecture is significantly better than DeepCorr's judgment accuracy.

4. LOCATION CORRELATION ATTACK MODEL OF MIDDLE NODES

In this section, a circuit reconstruction scenario is proposed based on the traffic of middle nodes. This scenario consists of the node location attack model and the correlated traffic attack model. Besides, the evaluation method of the circuit correlation is defined to make the attacks applicable to the traffic analysis of the middle nodes.

4.1 Phase I: Node Location Attack

The data forwarding location indicated by the control node in the circuit is analyzed. It is seen that the forwarding load features are weakened due to the protection of the padding mechanism of the hidden circuits. The padding mechanism scans the load marks of the circuit creation and data forwarding phases. Nevertheless, the statistical features such as the load sequence, quantity, and duration verified by Kwon (2015) are not significant enough to serve as the classification indicators. The padding does not disturb nodes to forward the target, load command, and other node-level information. Therefore, it is reasonable to consider node-level information as the classification indicator for determining the node location and improving the reliability of the attack results.

Jansen (2018) recommended the use of circuit purpose classifier and circuit location classifier Onionpop to determine the node location. However, Jansen does not consider the impact of the complex communication environment (circuit padding and data padding) on the two classifiers. Therefore, the node location of each circuit type in this paper is separately labeled, and the feature indicators such as the front & rear hop node types, load commands, and load forwarding amount are continuously used. Furthermore, add the total data transmission time and packet burst frequency, and the classification algorithm is also employed to determine the location information of the nodes.

Table 2 shows the node location division corresponding to each circuit type. A total of k=21 types of node locations are involved. For example, R-HS-M indicates the middle node on the hidden service side of the rendezvous circuit. Bridge nodes are also considered. In cases where a specific node attribute information cannot be determined it is either a network terminal or an undocumented plug-in node of obfs4 or meek bridge. Therefore, the load transmission features of the bridge nodes are described.

Table 2. Circuit classification and node locations

Purpose Position	General	Intro_Client	Intro_HS	Rend_Client	Rend_HS
Exit	G-E	N/A	N/A	N/A	N/A
Guard	G-G	I-C-G	I-HS-G	R-C-G	R-HS-G
Bridge	G-B	I-C-B	I-HS-B	R-C-B	R-HS-B
Middle	G-M	I-C-M	I-HS-M	**R-C-M**	**R-HS-M**
Middle2	N/A	I-C-M2	I-HS-M2*	N/A	R-HS-M2
Introduction	N/A	N/A	I-HS-I	N/A	N/A
Rendezvous	N/A	N/A	N/A	R-C-R	N/A

* For hidden services, besides the introduction nodes created for the first time, a second middle node is added.

4.2 Model Structure

The overall structure of the node location classification model is shown in Fig. 3, where the input training sample is denoted by p and the test sample is indicated by p^*. The basic classifier is determined as the Support Vector Machine (SVM) model (Chang & Lin, 2011). The feature changes are then mapped to a high-dimensional space through the Radial Basis Function (RBF). Subject to the True confidence probability ρ, the sample is determined as positive, otherwise, it is determined as negative. The specific objective function is shown in Equation (5), where *m* indicates the number of samples input into space, $y^{(i)}$ denotes a nonlinear function, w is the hyperplane vertical factor, and $C\sum_{i=1}^{m}\zeta_i$ is the constraint slack variable.

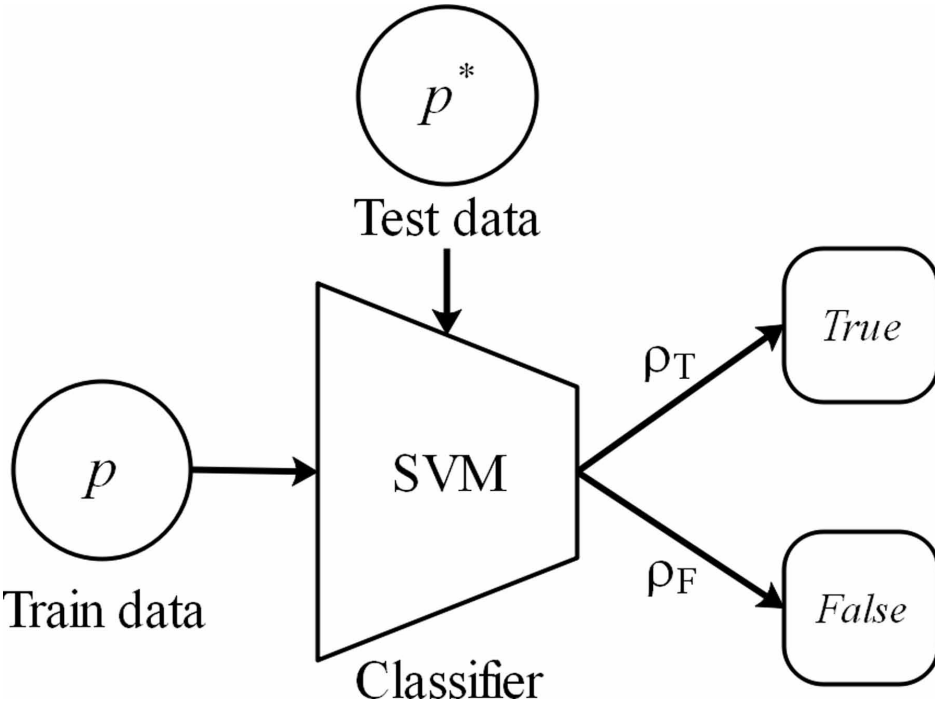
$$\begin{cases} \min \dfrac{1}{2}\left\|w\right\|^2 + C\sum\limits_{i=1}^{m}\zeta_i \\ s.t.\ \ y^{(i)}\left(w^T x^{(i)} + b\right) \geq 1 - \zeta_i \\ \zeta_i \geq 0,\ \ i = 1, 2, \cdots, m \end{cases} \tag{5}$$

Complete input samples covering the node-level information (circuit information and load information) are derived from the monitoring node according to the control protocol. For the circuit information, the corresponding states of each circuit are recorded including the unique circuit identity (ID), the traffic ID, the front & rear hop node marks, and the total traffic transmission time. For the load information, it also record the load quantity and load command sent and received in a particular direction. The packet burst frequency is the number of times when the statistical load interval is longer than 50 ms. Testing shows that the features might become too generalized by the granularity of fine intervals.

4.3 Node Location Training And Testing

Considering that the attackers possess the favorable prerequisites for all node locations, a total of k classifiers are trained to improve the discrimination strength of the model. The one-versus-all (OVA) dichotomy algorithm is then applied for the classifiers, where the target type of each classifier is marked as positive and the rest k-1 types are marked as negative. To balance the positive and negative data within the dataset all the data are also diluted at 1:1 for training. The training performance of

Figure 3. The overall structure of the node location classification model



10-fold cross-validation classifiers is applied during the training, and the optimal result $\rho_{svm_i}^{\max}$ is recorded.

The specific steps of the model identifying the malicious node locations in each traffic are presented in Algorithm 1. The first three steps describe the process in which the test data p^* is first input into the middle node classifier $\left\{SVM_i \middle| i=1,2\right\}$, and it is then sent to the rest of the classifiers when it is not the target location according to the judgment. In addition, if the probability after the experimental evaluation is higher than 90% of the maximum training probability of the classifiers, this type would be determined as the node location; otherwise, the node would be temporarily determined as the middle location. The accurate judgment of the identification of the node locations provides the basic conditions for subsequent traffic evaluation (see, Section 4.7).

4.4 Phase Ii: Middle Node Traffic Correlation

Attackers carry out passive traffic observation at the determined middle node. Nevertheless, in the case of complex encrypted traffic patterns, the attackers are unable to mark all possible correlated and non-correlated traffic data in a real environment. This results in challenges to the learning ability of the model. The GAN (Goodfellow et al., 2014) supports a typical unsupervised learning mode, which generates synthetic traffic using random noise, learns the differences in potential feature vectors between the synthetic traffic and real traffic. The GAN finally possesses the ability to generate data close to real correlated traffic. Accordingly, the model can determine the traffic away from the correlated features as the non-correlated traffic.

A previous study (Nasr et al., 2018) shows that the original features displayed by the communication traffic of hidden services are divided into three categories including packet transmission direction, content size (Size), and packet interval [inter-packet delay (IPD)]. Consequently,

Algorithm 1. Algorithm of Model Evaluation

Input: Classifier $\left\{SVM_i \middle| i=1,2,\cdots,k\right\}$, Test data p^*, Maximum probability of classification $\rho_{svm_i}^{\max}$
Output: Location classification probability ρ
Steps:
1) Input p^* to classifier $\left\{SVM_i \middle| i=1,2\right\}$
2) **if** Classifictaion results $True$ **then**
3) **return** $\rho \leftarrow \rho_{svm_i}$
4) **else**
5) **for** Input p^* to classifier $\left\{SVM_i \middle| i=3,\cdots,k\right\}$ **do**
6) **if** $\rho_{svm_i} \geq 0.9 \cdot \rho_{svm_i}^{\max}$ **then**
7) **return** $\rho \leftarrow \rho_{svm_i}$
8) **break**
9) **else**
10) **return** $\rho \leftarrow \rho_{svm_i}$ $\left\{i=1,2\right\}$
11) **end if**
12) **end for**
13) **end if**

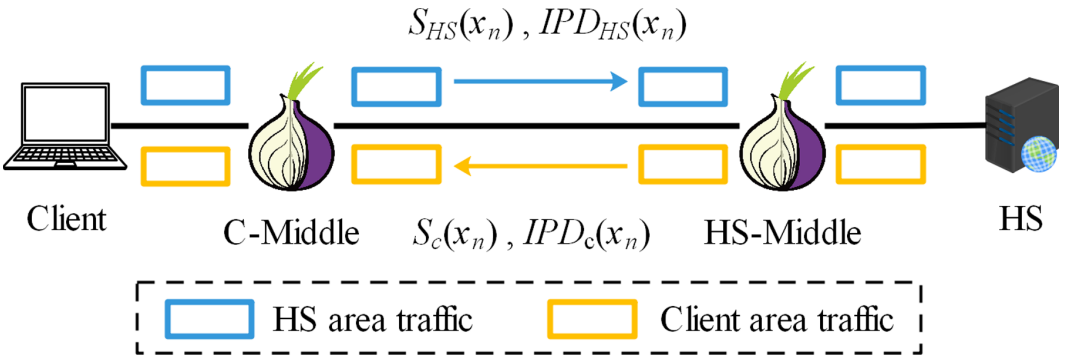
based on the original traffic features, the prior features of the circuit traffic mentioned in Section 4.1 are included in this paper. For example, the response traffic for hidden services is higher than the request traffic on the client-side. As shown in Figure 4, the traffic is tentatively divided into bidirectional combined communication traffic. Different from the previous single-ended traffic, the bidirectional traffic is also divided into the client area traffic C and hidden service area traffic HS according to the transmission target, and the combined traffic $\mathrm{A}\left(x\right)$ is formed. Such traffic pattern depiction strengthened the correlations among the transmission load features within the same communication tunnel.

$$\mathrm{A}\left(x\right)=\begin{bmatrix} S_C\left(x_n\right) & S_{HS}\left(x_n\right) \\ IPD_C\left(x_n\right) & IPD_{HS}\left(x_n\right) \end{bmatrix}$$

4.5 Model Structure

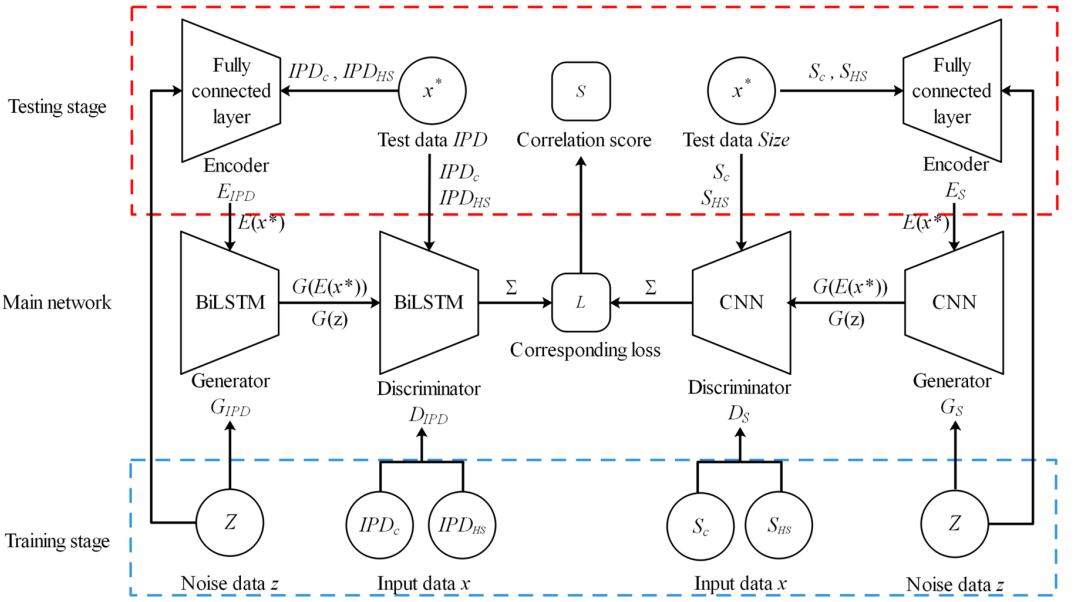
The overall structure of the traffic correlation model of the middle nodes is shown in Figure 5. According to the statistical observation of the traffic sequence, it is seen that the packet interval in the sequence is distributed more randomly than that of the packet size. This results in large fluctuations in the subsequent feature generation and evaluation results. Therefore, a dual-channel feature input structure is proposed to input the packet interval and packet size features into the GAN model in parallel. This is to form the independent target loss value L. The GAN designed for the packet interval is equipped with the BiLSTM model (Ian Goodfellow and Yoshua Bengio and Aaron Courville,

Figure 4. Bidirectional combined communication traffic



2016) as the backbone network. The BiLSTM is superior in the continuous memory of multivariate time series, especially in the extraction of the dependency of bidirectional traffic. Furthermore, the hidden states of the forward and backward transmission at the same location show different results, therefore the BiLSTM is well suited for packet interval sequence evaluation. The other end is a branch to process the packet size for which the CNN model (Ian Goodfellow and Yoshua Bengio and Aaron Courville, 2016) is applied as the backbone network. The traditional CNN is superior in the modeling of the sequence length with good performance in the efficiency of extracting hidden features due to the acceleration feature mapping of the multi-layer convolution. In summary, the advantages of dual channels are as the following: the sensitivity of feature changes of the independent learning does not have a mutual influence; the simulated traffic features generated directly have a high level of accuracy and robustness.

Figure 5. The overall structure of the traffic correlation model



The main structure of the GAN model is comprised of the following three backbone networks:

(1) Synthetic traffic generators G_{IPD} and G_S

The generators obtain the initial vectors from the random noise z and generate an approximation of the $G\left(z_n\right)$ vector for the features of the real traffic using the BiLSTM and CNN networks. The gradient loss function is:

$$L_{G_{IPD,S}}\left(z\right)=\frac{1}{N}\sum_{n=1}^{N}\left(1-\log D\left(G\left(z_n\right)\right)\right) \tag{6}$$

(2) **Correlated traffic discriminators D_{IPD} and D_S**

The discriminators are used to determine whether the *IPD* and *Size* in the generated correlated traffic $G\left(z\right)$ are consistent with the real correlated traffic x. Large input vectors can easily cause gradient expansion of the discriminator parameters and make the results unstable. This issue is addressed by adding the L2 regularized gradient descent penalty factor $\lambda\sum_{n}^{N}\left\|\nabla_{x_n}D\left(x_n\right)\right\|_2^2$ to calculate the penalty value for each discriminator component, where λ indicates the regularized weight decay coefficient.

The combined gradient loss function of the discriminators is

$$L_{D_{IPD,S}}\left(x,z\right)=\frac{1}{N}\sum_{n=1}^{N}\left[\log D\left(x_n\right)+\log\left(1-D\left(G\left(z_n\right)\right)\right)+\lambda\left\|\nabla_{x_n}D\left(x_n\right)\right\|_2^2\right] \tag{7}$$

(3) Autocoders E_{IPD} and E_S

During the pre-processing of the test samples, x^* is subjected to backpropagation through G. This is to obtain the noise vector $z:\ x^*\Rightarrow G\left(x^*\right)\Rightarrow z$. However, the G contains multiple layers of nonlinear transformation neurons which results in a complex solution (Creswell & Bharath, 2019). To simplify the propagation mode of the noise vector z, the autocoder E is added in front of the generator to replace the propagation function G. The autocoders E_{IPD} and E_S at each branch are used to convert the evaluation traffic x^* into a low-dimensional feature vector. The forward propagation route is as follows: $x^*\Rightarrow E\left(x^*\right)\Rightarrow G\left(E\left(x^*\right)\right)$. The L2 regularized Mean Square Error (MSE) algorithm shown in Equation (8) is also applied to calculate the loss function of the input and output gaps. In addition, the generator here serves as the decoder of the hidden vector space and E_{IPD} E_S only indicates the coding process of the input data to avoid confusion with the aforementioned generator G.

$$L_{E_{IPD,S}}\left(z,G\left(E\left(z\right)\right)\right)=\sqrt{\frac{1}{N}\sum_{n=1}^{N}\left[z_n-G\left(E\left(z_n\right)\right)\right]^2} \tag{8}$$

4.6 Correlated Traffic Training And Testing

The alternating iterative method is applied to train the network model on the grounds of the initial idea of GAN. First, select N real training samples $\left\{x_n,1\le n\le N\right\}$ as the correlated samples $p_{data}\left(x\right)$. Use the normal distribution method for random sampling of the traffic noise vectors $p_z\left(z\right)$ $\left\{z_n\sim\mathcal{N}\left(0,1\right),1\le n\le N\right\}$ to form high-dimensional feature vectors. Secondly, $p_{data}\left(x\right)$ is input to the discriminator D, and the Stochastic Gradient Descent (SGD) method is applied to train the corresponding loss objective function. During the iterative training, the parameters are updated as per the minimized binary cross-entropy. The loss weight is also transferred backward to the generator. Thirdly, the feedback of the generator parameters is received to train the generator G, and the random traffic noise $p_z\left(z\right)$ is sampled. The SGD method is then adopted for weight training of the maximized binary cross-entropy. Finally, the optimal generator parameters are used to train the autocoder E. This ensures consistency between the input data and code data.

After training the model, the discriminator D is devised with the optimal weight parameters which is capable of distinguishing the tested traffic x^* and the generated traffic $G\left(z\right)$. The discriminator however lacks the evaluation benefit of G on the tested traffic x^*. The reconstruction of the loss function can make up for the evaluation benefit and the gap between the reconstruction effect $G\left(z\right)$ and the evaluation traffic x^* is described by the objective function after passing through the generator. Note that the noise vector z is output as $E\left(x^*\right)$ by the autocoder E, and the reconstructed loss function is

$$L_R\left(x^*,G\left(E\left(x^*\right)\right)\right)=\sqrt{\frac{1}{N}\sum_{n=1}^{N}\left[x^*{}_n-G\left(E\left(x^*\right)\right)\right]^2} \tag{9}$$

4.7 Circuit Correlation Evaluation

The correlation of the two phases is evaluated and define a correlation score evaluation mode. Equation (10) shows that the score of Phase I is the product of the node probabilities $\rho\left(M\right)$, and the score of Phase II is calculated based on the equilibrium value between the reconstructed loss objective function and the discriminator loss function. The parameter ζ in Equation (11) is the variable adjustment hyper-parameter. The final correlation score is calculated according to Equation (12). The scores of the two phases based on natural logarithms highlight the significance of the traffic correlation with the same node location probability. For example, the target in this paper is a group of observation nodes M_i and M_j, as well as two types of feature vectors $I=\left\{IPD,Size\right\}$. If the observed traffic x^* is input to the correlation model, it is subject to feature mapping through E_{IPD} and E_S. The mapping results are then inputted into the generators G_{IPD} and G_S for the reconstruction of the synthetic traffic. The reconstructed loss value of the synthetic traffic and the observed traffic is calculated, and then the discriminators D_{IPD} and D_S are used to discriminate and calculate the loss of the reconstructed traffic and observed traffic.

According to the regulations on the model, the lower the loss, the higher the traffic similarity. The structure of the correlation model and the evaluation process is shown in Algorithm 2, and the final result is denoted by $\mathrm{Result}[node_i^j]$.

$$S\left(node_i^j\right) = \rho\left(M_i\right) \cdot \rho\left(M_j\right) \tag{10}$$

$$S\left(x^*\right) = \sum_i^I \left(\zeta_i L_D\left(x^*, G\left(E\left(x^*\right)\right)\right) + \left(1-\zeta_i\right) L_R\left(x^*, G\left(E\left(x^*\right)\right)\right)\right) \tag{11}$$

$$S\left(node_i^j, x^*\right) = \frac{\ln S\left(node_i^j\right)}{\ln S\left(x^*\right)} = \log_{S\left(x^*\right)} S\left(node_i^j\right) \tag{12}$$

5. DATASETS

In this section, describes the data collection and processing in different phases of the model, and the internal node location, and real correlated traffic datasets. These provide a basis for the model training and performance evaluation.

5.1 Internal Node Location Datasets

In Phase I, the malicious nodes are used to perform statistical analysis on the appearance of the forwarding traffic. The internal processing operations of the nodes are then collected. To solve the complex filtration of traffic, logs, and other data, the data collection is implemented on the Shadow environmental simulation platform (Jansen & Hopper, 2012). This platform enables the mass customization of the node data. A complete private Tor network framework is built on Shadow that performs and processes the Tor protocol.

The initial goal is to identify the specific location of nodes in the communication circuit (R-C-M or R-HS-M). Therefore, to perform efficient data collection the code of the control end of the Tor protocol (Tor-0.4.2.1) is modified, and the communication traffic is imported into the selected nodes through the control items in Table 3. Real consensus documents are used on Shadow to generate component information. The hidden services are embedded into the Alexa hypercircuit page (Zhuo et al., 2018). By starting the data collection, the clients access the website domain of the hidden services in bulk and request the conventional servers to download the files. After collection, the communication traffic of each circuit type is tracked and recorded according to the stdout output logs in the monitoring nodes. To balance the number of locations of each node and to ensure more than 100,000 instances for each type of circuit, each node location has more than 20,000 instances. The internal dataset also has a total of 360,000 instances.

5.2 Real Correlated Traffic Datasets

The communication states of traffic acquired from the above internal environment are relatively stable. Although the access process of the client-side is randomized, the results of traffic difference judged by the cosine similarity show that there is a small basic deviation degree of traffic in each circuit of the same type. It is believed that the communication traffic should be mixed with noise,

Algorithm 2. Algorithm of Model Construction and Evaluation

Input: Node location probability $\rho\left(M\right)$, Number of iterations k, Batch samples N, Test data x^*, Real samples $\left\{x_n, 1 \le n \le N\right\}$, Random noise $\left\{z_n \sim \mathcal{N}\left(0,1\right), 1 \le n \le N\right\}$

Output: Evaluation results $\mathrm{Result}[node_i^j]$

Steps:

for number of training iterations **do**
for k steps **do**
Sampling from real samples $p_{data}\left(x\right) \Leftarrow \left\{x_n, 1 \le n \le N\right\}$
Generate Discriminator data $D\left(x\right)$
Sampling from random noise $p_z\left(z\right) \Leftarrow \left\{z_n \sim \mathcal{N}\left(0,1\right), 1 \le n \le N\right\}$
Generate Discriminator data $G\left(z\right)$
Compute $L_{D_{IPD,S}}\left(x,z\right)$ minimize loss function $\left(7\right)$ and update parameters
Compute $L_{G_{IPD,S}}\left(z\right)$ maximize loss function $\left(6\right)$ and update parameters
Compute $L_{E_{IPD,S}}\left(z,G(E(z))\right)$ minimize loss function $\left(8\right)$ and update parameters
end for
end for
for Matching Nodes i, j **do**
Mapping of test data $x^* \Rightarrow G\left(E\left(x^*\right)\right)$
Compute $L_R\left(x^*, G\left(E\left(x^*\right)\right)\right)$ loss function $\left(9\right)$
Generate Discriminator data $D\left(x^*\right)$
Compute $L_D\left(x^*, G\left(E\left(x^*\right)\right)\right)$ loss function $\left(7\right)$
Calculate equations $\left(10\right)$, $\left(11\right)$, and $\left(12\right)$ in order
$\mathrm{Result}[node_i^j] \leftarrow S\left(node_i^j, x^*\right)$
return $\mathrm{Result}[node_i^j]$
end for

which is conducive to improving the learning ability of the GAN. Therefore, the correlated traffic is collected in the real node transmission sites.

Figure 6 illustrates the deployment environment for the collection of real middle node data. OnionShare (*OnionShare v2.2.1*, n.d.), a website file sharing and hosting tool, is configured for the servers of hidden services. Shared files of 5-500 MB are randomly placed in the tool, and the services are set to be stopped immediately after the request to display different circuit states between the requests.

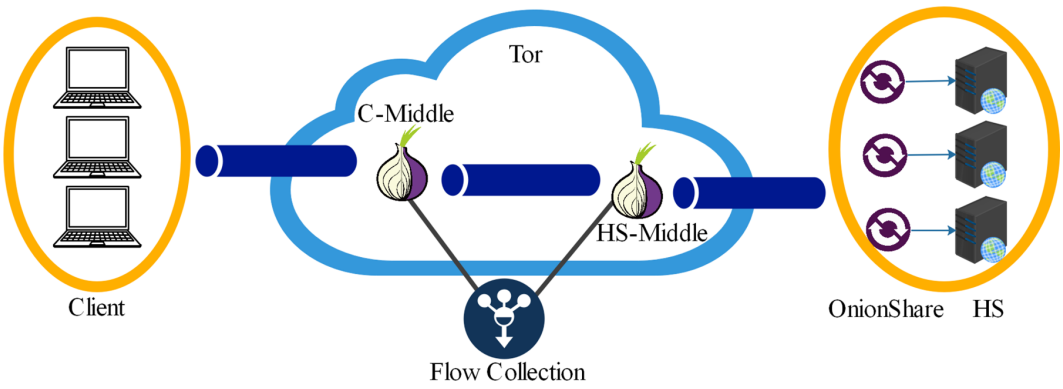
Table 3. Control items of the monitoring nodes

Control item	Description
*HSMiddle1Nodes	The first middle node on the fixed hidden service side
*HSMiddle2Nodes	The second middle node on the fixed hidden service side
*RendezvousNodes	Fixed rendezvous nodes
*IntroductionNodes	Fixed introduction nodes
MiddleNodes	Middle nodes on the fixed client-side
EntryNodes	The setting of guarded entry nodes
ExitNodes	The setting of exit nodes
Bridge	The setting of bridge entry

*Newly added control items of the Tor protocol code.

Here there is an opportunity to collect the long traffic. The two middle nodes in Tor are private control nodes that are also published in the common consensus document.

Figure 6. Data collection scenario of real middle node data



5.3 Data Extraction And Processing

The log information and load data are collected. The features of node locations are extracted according to the description in Section 4.1, where each feature is composed of the correlated information of a group of node ID and circuit ID. The character string is turned to digital features through hot coding. Each type is classified as a libsvm file to provide specific target datasets for the multiple node location targets mentioned above. The target and non-target labels are also added to the dataset before model training, and the 1:1 dichotomy mode is applied to incorporate the imbalance between samples. For example, for training the R-C-M location, the positive sample is labeled as 1, and the negative sample is 0.

The initial sampling is carried out on the correlated traffic features as described in Section 4.4. The statistics of the feature value of each correlated load are carried out and the load-interval and size are subject to normalization within the range of (0,1]. This is to form a standardized data format that is directly inputted to the GAN. Given the consistency of data input dimensions, the first 300 to 900 data are set as the input cutoff used for the packet interval and packet size. The data is also

divided into the training, validation, and testing datasets according to the 7:1:2 ratio. The extraction of the datasets is shown in Table 4.

Table 4. Dataset extraction

Label	Node location	Correlated traffic	Non-correlated traffic
Feature variables	21	4	4
Traffic length	N\A	300-900	300-900
Training size	247463	5000×0000	N\A
Testing size	111685	1000×1000	1000×1000

6. EXPERIMENTAL EVALUATION

This section presents the experimental evaluation results of the proposed model.

6.1 Evaluation Indicators

Here the identification performance of the SVM and GAN models for node locations and correlated traffic are evaluated. The performance measures are the standard evaluation indicators of the deep learning models (Creswell & Bharath, 2019) including precision, recall, and F1 score.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{13}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{14}$$

$$\text{F}_1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{15}$$

Furthermore, the maximum mean difference (MMD) (Li et al., 2015) is used to evaluate the attack model's mastery degree of the distribution of training data features. The kernel trick is used to map the sample vectors of the random noise $G\left(z\right)$ and the test sample x^* to a multi-dimensional feature space. The similarity of distribution between the two parameters is then compared. According to Equation (16), if MMD approaches 0, the features became more similar. The kernel function k enables accelerated calculation using the Gaussian kernel function.

$$\begin{cases} MMD\left(G\left(z_i\right),x_j^*\right)=\dfrac{1}{N\left(N-1\right)}\sum_{i=1}^{N}\sum_{j\neq i}^{N}k\left(G\left(z_i\right),G\left(z_j\right)\right) \\ \qquad\qquad -\dfrac{2}{NM}\sum_{i=1}^{N}\sum_{j=1}^{M}k\left(G\left(z_i\right),x_j^*\right) \\ \qquad\qquad +\dfrac{1}{M\left(M-1\right)}\sum_{i=1}^{M}\sum_{j\neq i}^{M}k\left(x_i^*,x_j^*\right) \\ \qquad\qquad i,j=0,1,2,\cdots,n \\ k\left(G\left(z_i\right),G\left(z_j\right)\right)=\exp\left(-\dfrac{1}{2\sigma}\left|G\left(z_i\right)-G\left(z_j\right)\right|^2\right) \end{cases} \tag{16}$$

Table 5. Hyperparameter adjustment in the GAN model

Network	Details of hyper-parameters	Details of generator hyper-parameters	Details of discriminator hyper-parameters
LSTM	num_epochs:100 batch_size:32 weight:0.5 Activation:softsign	hidden_units: 100 λ :2 Learnin_rate:0.0005 Layers:1	hidden_units: 100 λ :2 Learnin_rate:0.0001 Layers:1
CNN	num_epochs:100 batch_size:128 kernel:800 Learnin_rate:0.0005 Activation:tanh	Window1:(1,30) Window2:(1,30) Stride1: (2,1) Stride2: (2,1) λ :0.8	Window1:(1,30) Window2:(1,30) Stride1: (2,1) Stride2: (4,1) λ :0.8
Encoder	num_epochs:300 batch_size:128 Learnin_rate:0.0005	Dimension:30	---
GAN	dropout:0.1 ζ_{IPD} :0.01-1 ζ_S :0.01-1	G_rounds: 1, E_rounds: 1,	D_rounds: 4,

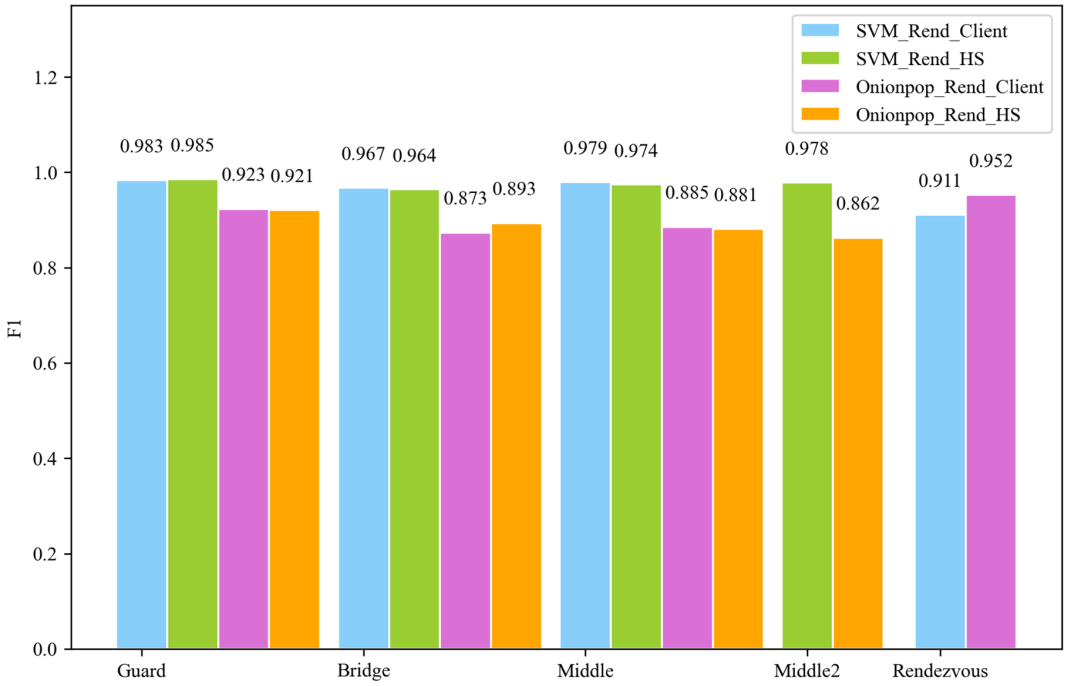
6.2 Parameter Adjustment

The GAN hyper-parameters are presented in Table 5. To maximize the feedback efficiency on the premise of not reducing the identification accuracy, the number of hidden neurons is set to 100. The penalty coefficient is also maintained at 2 on account of the serial feedback features of the LSTM network. The debugging feedback efficiency of the CNN is much higher than that of the LSTM. Therefore, the debugging goal is to maximize the identification accuracy of the packet size, with 2 convolutional layers and the sliding window of 30 as the optimal settings. There are 2 fully connected layers in the Encoder network which input the feature mapping into generators after reducing the dimension to 30. For iterative training of the GAN model, E and G are respectively trained once after training D for four times according to the setting. The dropout of each network is also set to 0.1 to prevent over-fitting of the data during the process. For evaluation score feedback, the parameters ζ_{IPD} and ζ_S are debugged to obtain the optimal proportion of the loss function values.

6.3 Node Location Evaluation

In this section, the identification performance of the node location based on the SVM is evaluated. To meet the circuit correlation target, the evaluation results show the judgment of Rendezvous_Client and Rendezvous_HS in the rendezvous circuit as in Table 2, and the node labels included {Guard, Bridge, Middle, Middle2, Rendezvous}. The testing set of each type of node location contains 2,000 groups of feature vectors. Figure 7 demonstrates the comparison of performance between SVM and Onionpop models. It is seen that for the first 4 types of node labels, the F1 score of the SVM is above 0.9 points and higher than that of Onionpop. The F1 score of Onionpop (0.041 points) is also higher than that of SVM at the Rendezvous point. Note that the identification results of the Middle tag node indicate the effective judgment of the circuit correlation. In addition, due to the asymmetry of the communication load in the transmission channel, the identification result of the middle circuit on the Client-side is superior to that on the HS side. Therefore, simulating the burst features of the load is required to disclose more effective information on the Client side.

Figure 7. The node location judgments



In addition, Algorithm 1 can integrate the traditional single classifier. Therefore to investigate whether other classifiers(k=21) are capable of improving the recall and precision where there is some FP in $\left\{SVM_i \middle| i=1,2\right\}$. As it is seen in the last column of Table 6, the precision of a single middle node is increased to 0.989 by other classifiers. The recall and F1 scores also reach 0.994 and 0.991 points, respectively. The results close to 1 suggest that the identification performance of multi-classifiers is superior to that of a single classifier.

Table 6: Identification performance of multi-classifiers

Indicator	R-C-M	R-HS-M	Other Classifiers
Precision	0.974	0.968	0.989
Recall	0.986	0.981	0.994
F1	0.980	0.974	0.991

6.4 Traffic Correlation Evaluation

To verify whether the GAN model applies to the correlated traffic identification scenarios, investigate the loss values $L_{G_{IPD,S}}(z)$ $L_{D_{IPD,S}}(x,z)$ and $L_{E_{IPD,S}}\left(z,G\left(E\left(z\right)\right)\right)$ generated during model training. A total of 2,000 pairs of correlated traffic with a traffic length of 300 are considered as the training set. Figure 8 shows the visualized loss curves of the packet size (a) and packet interval (b) generated after 100 times of iterative training. In this plot, the shadows and solid lines indicate the extreme value and mean value of multiple training, respectively. As it is seen in Fig. 8, the initial loss value of the autocoder is larger than that of the discriminator and generator. This is because 80-90 times of iterations are required for stable convergence as the decoding vector $G\left(E\left(z\right)\right)$ is generated through the generator and the loss value increased with the changes in feature vectors. Figure 8(a) also presents the changes in the loss value of the packet size. The loss values of the generator and discriminator are mutually entangled and resisted but converged to the optimal value after 40-50 iterations. The advantages of the loss judgment function with added penalty factors are also seen, where the convergence effect of the loss value is improved in the case of the frequent adversary. The loss value of the packet interval is slowly converged and requires 55-65 times of iterations [Figure 8(b)]. The convergence of the generator is slower than that of the discriminator because the training process determines the preferential convergence of the discriminator, whereas the generator only converges after obtaining the parameter feedback.

Investigate the GAN to understand the learning process of the IPD and Size features and use the MMD to evaluate whether the GAN model can learn the basic distribution of the training data features during the adversary. The evaluation results of the first 50 iterations are shown in Figure 9. As it is seen GAN quickly learns the hidden feature vectors of the Size, which is similar to the convergence process of the loss value. The MMD is also decreased to 0.012 after 40 iterations. In terms of the IPD feature learning by the GAN, the MMD is decreased from 0.767 to 0.096 after 30 iterations but remained at 0.092-0.124 in the following 40 iterations. This verifies the initial judgment. The correlation of the IPD also shows a more complex behavior comparing with that of the Size, but the value disturbance caused by the complexity of the backbone model can not be excluded. However, for the current benefit evaluation, Size is more efficient in assisting the GAN model in training and judgment, and the IPD learning ability of the model will be enhanced subsequently.

To evaluate the interaction between the backbone networks of the model, the performance of the autocoder E, generator G, and discriminator D at the training phase is also evaluated. Consider traffic with a length of 300 in a training set and examine the prediction state of the backbone network on the input data. Figure 10(a) shows the process that the true value y_true and the predicted value y_pred of $D(x)$ and $D(G(E(x)))$ are calculated in the discriminator D through the sigmod function. It is seen that the true value is distributed very close to 1, while the predicted value synthesized by the encoder and generator is synchronized with the true value. The synthetic data is also considered to be close to the true value by the discriminator.

The box plots in Figure 10(b) display the scores of the autocoder and generator. Note that only the probability of the fully connected layers is output from the encoder. The logits value of identifying the data vectors is not normalized in x, $E\left(x\right)$ and $G\left(E\left(x\right)\right)$ is used as the judgment basis of the

Figure 8a. Iteration loss values of the training loss: Iteration curve of the packet size training loss

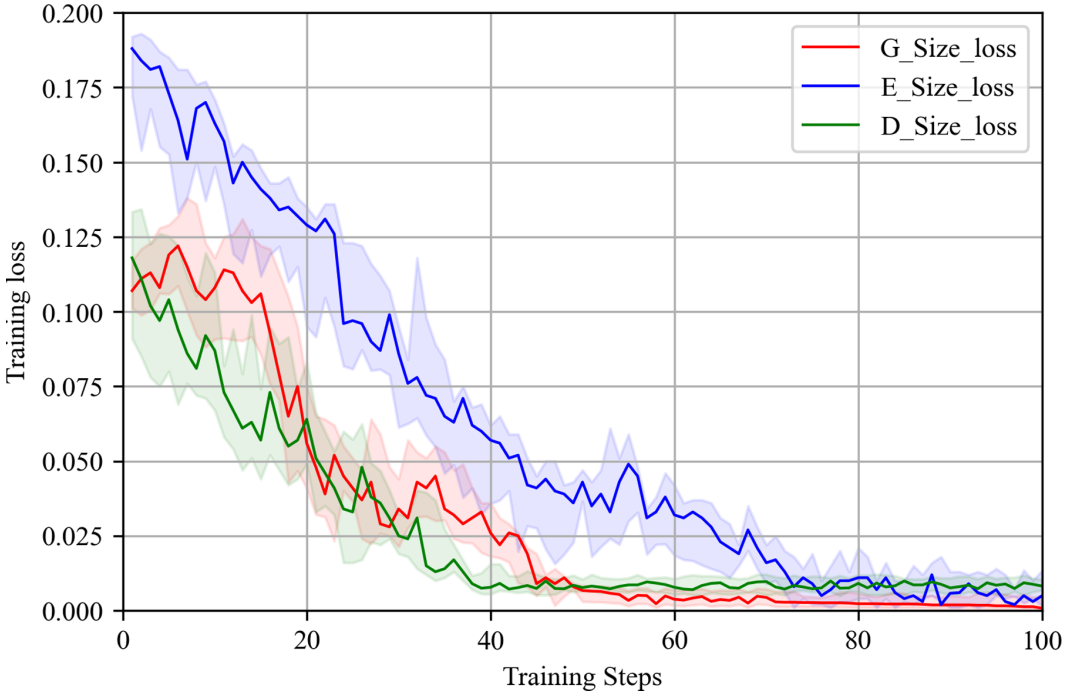


Figure 8b. Iteration loss values of the training loss: Iteration curve of the packet interval training loss

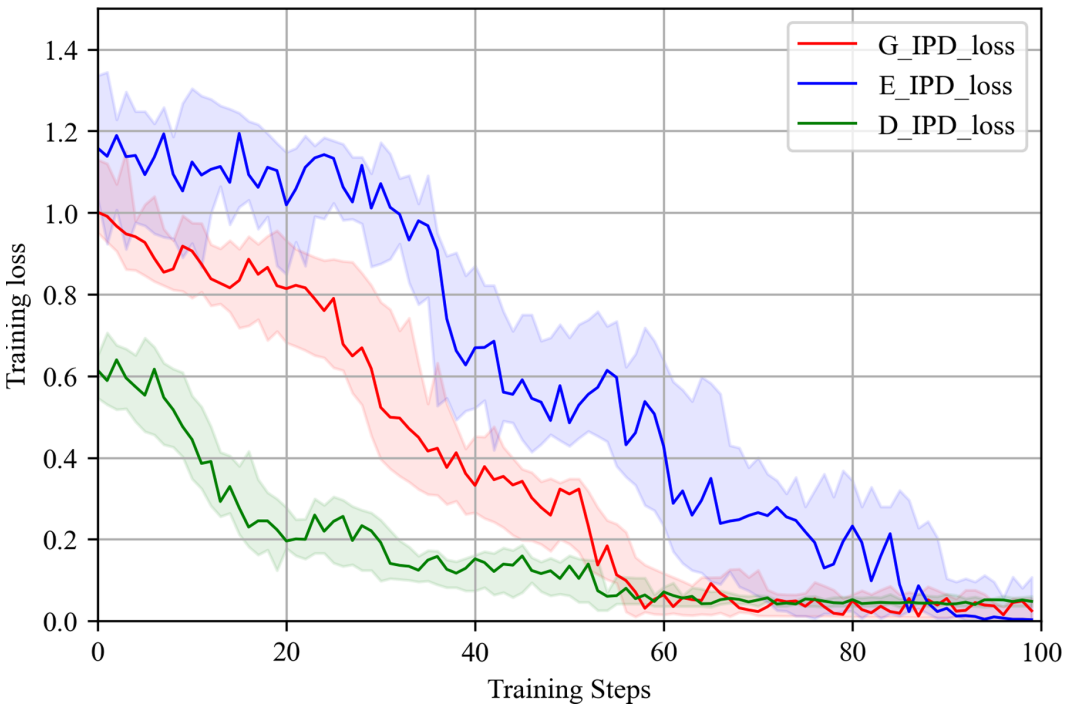
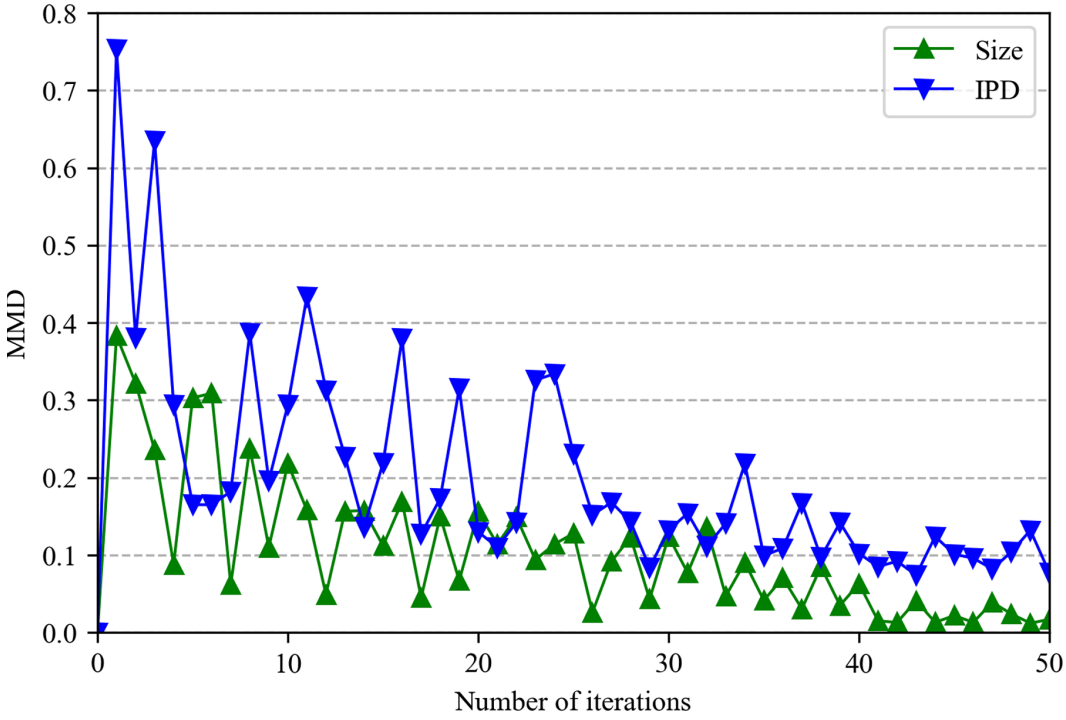


Figure 9. Iterative changes of the MMD in IPD and size



predicted value. The corresponding input data is retained, of which y_encode indicates the coded value of the intermediate process. The results show that the predicted value $G\left(E\left(x\right)\right)$ is highly correlated with the true value x, with a small difference between the mean and the median. In addition, the range of the encoded $E\left(x\right)$ value is narrowed because of the dimension reduction of feature vectors by the encoder. The $G\left(x\right)$ shown on the right side of the figure indicates the testing results after removing the Encode. The range of the predicted value of the generator is expanded and becomes higher than the mean and median of x. The comparisons above suggest that the data transfer effect between the autocoder and the generator is highly efficient and the added encoder is beneficial to the prediction of the input data features by the GAN.

Only use the correlated traffic in the training phase of the GAN. This is because the primary objective of this paper is to effectively distinguish the correlated traffic and non-correlated traffic without prior knowledge of the non-correlated traffic. The evaluation score (Score) described in Equation (11) intuitively interprets the distinction of the model between the two types of targets. In other words, 2,000 pairs of correlated and non-correlated traffic are used for evaluation, with a packet length of 300. Figure 11 is an independently plotted relative frequency distribution diagram of the scores. The training set (green) and testing set (red) of the correlated traffic and the testing set of the non-correlated traffic (blue) are served as the evaluation targets, and the interval frequency is calculated by the mean difference between $S\left(x\right)$ and $S\left(x^*\right)$. The results confirm that for the correlated traffic, the testing scores are close to the training scores, and they are even highly coincident at 0-0.04. For the non-correlated traffic, however, the testing score $S\left(x^*\right)$ is distributed sparsely with the high-frequency score area of 0.10-0.15. These results suggest that there are significant differences

Figure 10a. Score distribution of the true, coded, and predicted values: Scores of the discriminator

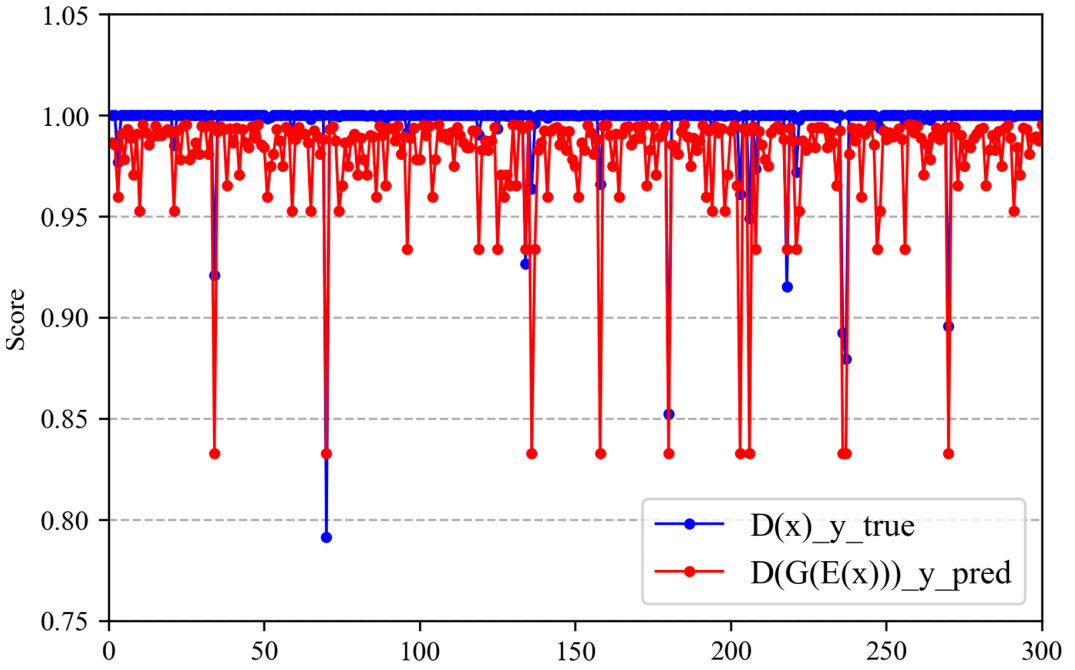
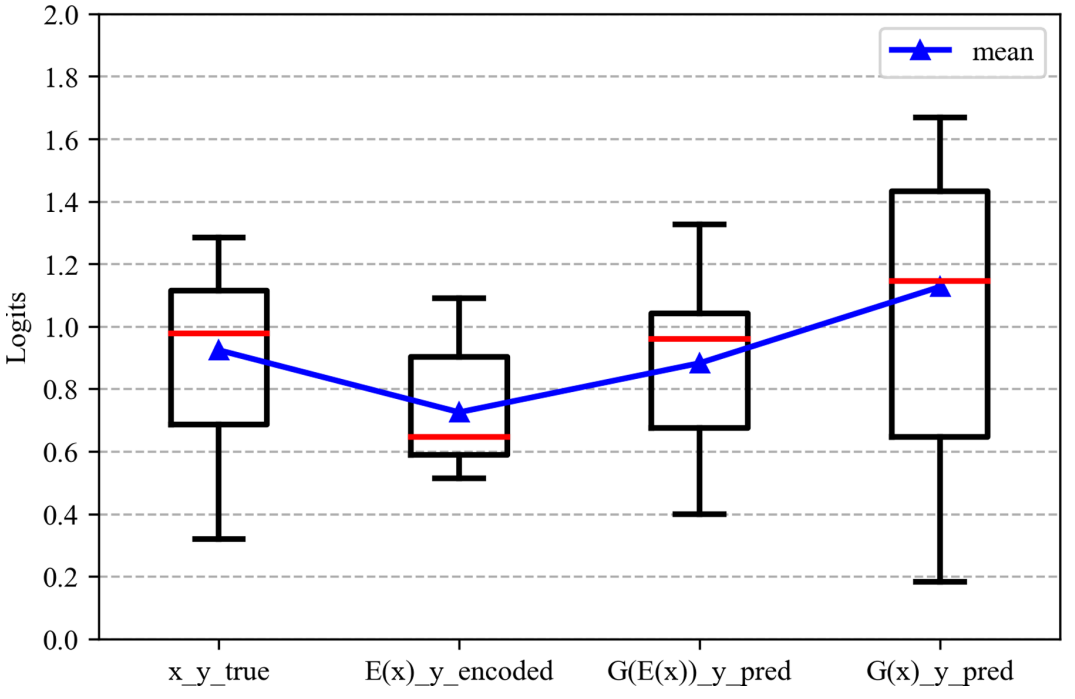


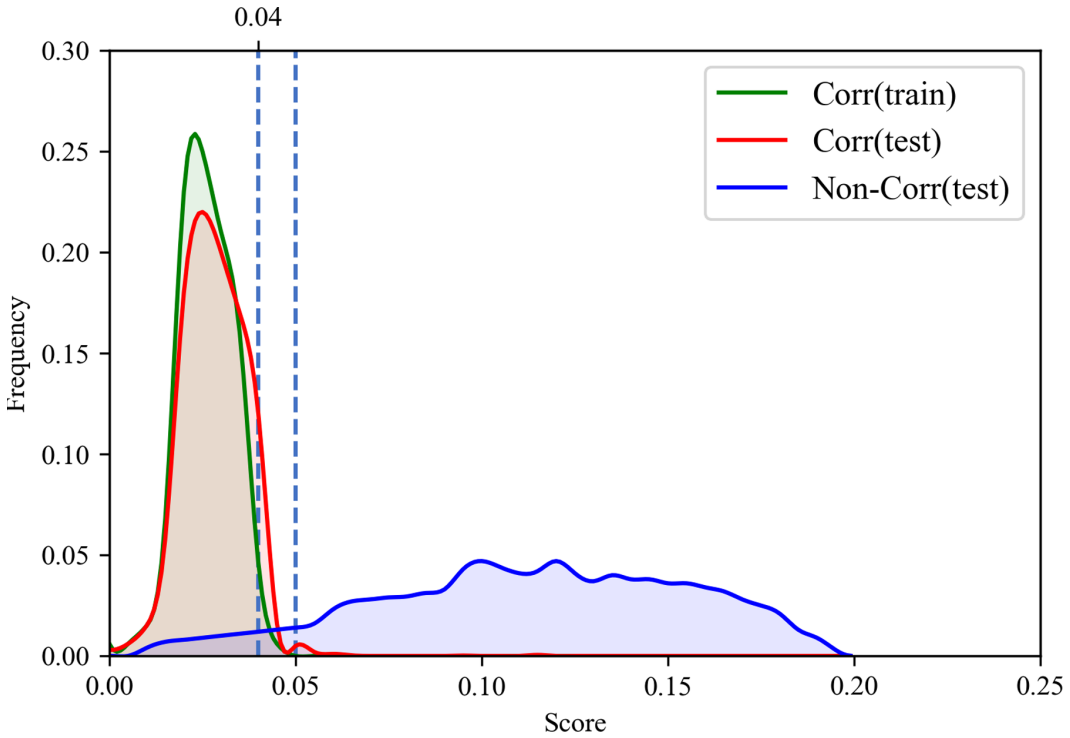
Figure 10b. Score distribution of the true, coded, and predicted values: Scores of the encoder and generator



between the correlated and non-correlated traffic. Therefore, in the model based on the distribution, the testing traffic with a score higher than 0.04-0.05 is intuitively identified as non-correlated traffic.

Note that only 8% of the non-correlated traffic have scores within the normal range of 0.05. However, the evaluation score is better suited for the classification of correlated and non-correlated traffic by balancing the proportions of the reconstruction and the discrimination loss values. Figure 12 illustrates the Precision-Recall curve which indicates the balance between the reconstruction and the discrimination loss values. The area under the curve (AUC) reflects the identification ability of the model and ζ denotes the balance parameter. For ζ=0.13 , the AUC is 0.536 and the optimal parameter is obtained for ζ =0.37. The maximum AUC is also 0.912. If ζ increases to 0.84, the AUC is reduced to 0.712. These suggest that GAN achieves higher classification performance on the targets due to the good performance of the reconstruction loss value.

Figure 11. Frequency distribution of score



6.5 Model Comparison

In this section, the identification performance of the (SVM+GAN) models is compared with the (Onionpop+DeepCorr) and (Onionpop+RAPTOR) combined models. For a fair comparison, the TPR and FPR combined indicator is the product of the identification results of the two models. All the testing data are obtained from the traffic sequences of 300 packets, including 2,000 pairs of correlated and non-correlated traffic. The ROC curves of the model at different thresholds are shown in Figure 13. As it is seen the GAN overperforms the other two advanced combined models. The AUC values also show great differences among different models and are equal to 0.958, 0.794, and 0.599 for the GAN, (Onionpop+DeepCorr) and (RAPTOR), respectively.

Figure 12. Precision-recall curve of score

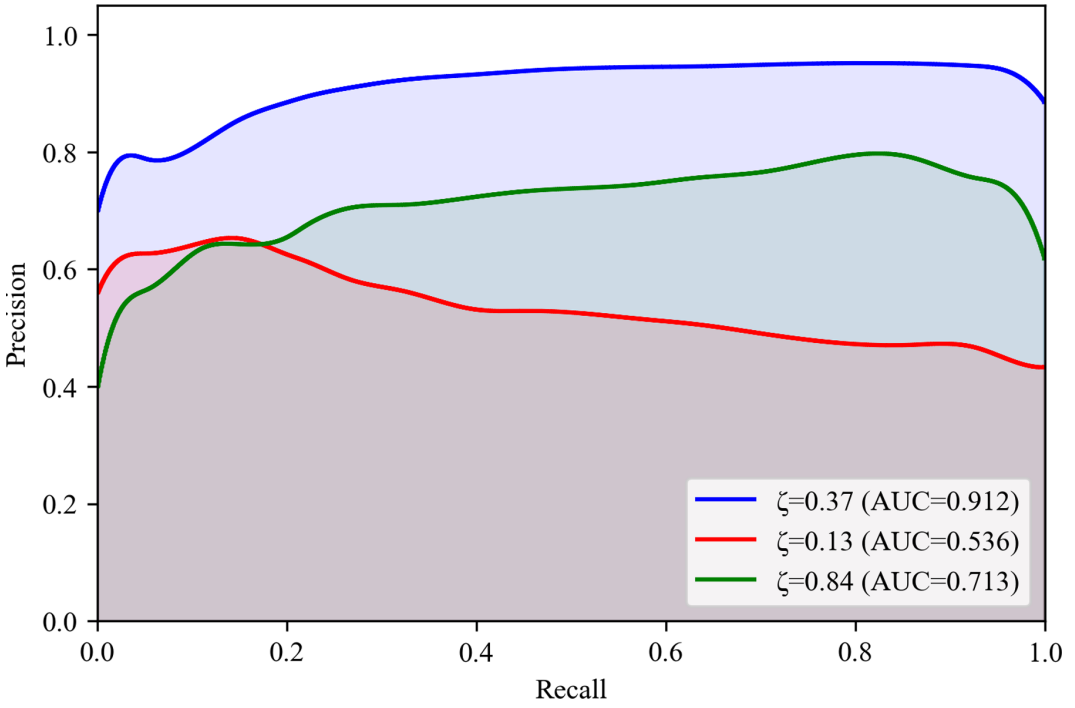
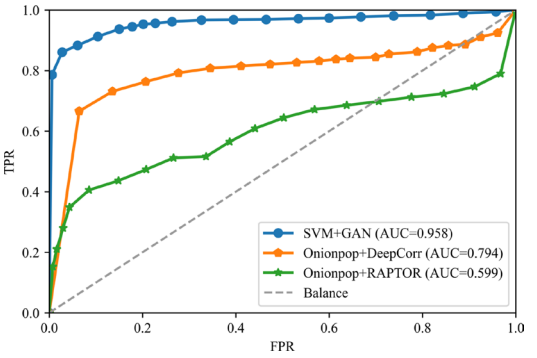


Figure 13. The ROC curves of the models with the identification traffic length of 300



As it was also stated in previous studies (Nasr et al., 2018; Sun et al., 2015) the packet length has a significant impact on the traffic correlation model. To verify the testing results of the packets in different lengths, the 2,000 pairs of additional long traffic with a packet length of 900 are added to the testing set. According to the results presented in Table 7, by increasing the packet length, the Precision of GAN is increased from 0.896 to 0.955, its Recall increased from 0.947 to 0.975, and its F1 score is 0.965 points. These suggest that the increase in packet length improves the identification precision of the proposed model. In contrast, although the F1 score of DeepCorr is increased from 0.728 points to 0.899 points, the result is not better than that of GAN. In practice, if the observed traffic is not limited, any model would show strong competitiveness.

Table 7. The identification traffic length of the model

Traffic length	Attack model	Precision	Recall	F1
300	GAN	0.896	0.947	0.921
	DeepCorr	0.773	0.688	0.728
	RAPTOR	0.353	0.325	0.338
900	GAN	0.955	0.975	0.965
	DeepCorr	0.916	0.884	0.899
	RAPTOR	0.432	0.412	0.421

As it is seen the processing efficiency of the models played a crucial role in scenarios with large traffic. It is also noted that the attackers determine the node location first and then identify the observed traffic due to the attack process of middle nodes. Therefore, the processing needs to be in serial. Nevertheless, the internal operation of the GAN supports parallel processing of the packet size and packet interval. The following experiments are carried out on a system with an Intel Core i7-9700K CPU @ 3.70GHz processor and NVIDIA GeForce RTX 2080ti. Table 8 lists the processing speed and F1 score of the combined attack model on single traffic.

The results in Table 8 shows that the total processing time is increased with the increase of the traffic length to varying degrees. The (Onionpop+RAPTOR) model is the quickest in processing 300 packets which is 3.33 ms faster than that of the (SVM+GAN) model. Its F1 score is however only one-third of the (SVM+GAN) model. The efficiency of the (SVM+GAN) model is also lower than that of the (Onionpop+DeepCorr) model. By comparison, the longer the traffic to be processed, the larger the increase of the processing time. The (SVM+GAN) model requires 12.06 ms to process the traffic with a length of 900. The major cause of this is that the LSTM backbone network used in the IPD reduces the judgment efficiency when processing long traffic. Moreover, since the set model of Algorithm 1 can only determine the location after multiple times of judgment, the mean processing time of the node location (4.74 ms) is longer than that of the Onionpop (3.37 ms). In summary, if the attackers select 300 packets of the observed traffic, the efficiency and precision of the circuit correlation are guaranteed at the same time.

Table 8. Processing efficiency of the attack models applied on a single node traffic

Attack model	*time (ms)	Traffic length	F1
SVM + GAN	7.36 (4.74+2.62)	300	0.912
	8.88 (4.74+4.14)	500	0.937
	12.06 (4.74+7.32)	900	0.959
Onionpop + DeepCorr	5.33 (3.37+1.96)	300	0.703
	5.98 (3.37+2.61)	500	0.788
	7.25 (3.37+3.88)	900	0.867
Onionpop + RAPTOR	4.03 (3.37+0.66)	300	0.298
	4.59 (3.37+1.22)	500	0.346
	5.22 (3.37+1.85)	900	0.406
*Total time: Mean identification time of the node location + mean classification time of the traffic correlation			

7. CONCLUSION

In this paper, a circuit reconstruction attack was proposed based on the middle node traffic. The problems of difficult correlation were also solved according to the node location identification and traffic correlation identification steps. In considered cases where the node location applied to the padding mechanism and the overall identification precision was improved by the effect of multi-classifiers. Further, the framework of AutoEncoder+CNN+BiLSTM is proposed to learn the basic features of the correlated traffic. After performing dimension reduction and mapping of the sample data, the reconstruction loss function of the generator and the discrimination loss function of the discriminator are obtained. The experimental results showed that the SVM+GAN model proposed in this paper accurately distinguishes the correlated and non-correlated traffic and overperforms the existing attack models. In future studies, the learning and identification abilities of the LSTM and CNN backbone networks will be optimized to satisfy the identification precision and computational efficiency of attackers in scenarios with large traffic.

ACKNOWLEDGMENT

This project is funded by the National Key Scientific Research Project (No. 2019QY1302 and 2019QY1305).

REFERENCES

Akcay, S., Atapour-Abarghouei, A., & Breckon, T. P. (2019). GANomaly: Semi-supervised Anomaly Detection via Adversarial Training. *Lecture Notes in Computer Science, 11363*, 622–637. 10.1007/978-3-030-20893-6_39

Biryukov, A., Pustogarov, I., & Weinmann, R. P. (2013). Trawling for tor hidden services: Detection, measurement, deanonymization. *Proceedings - IEEE Symposium on Security and Privacy*, 80–94. doi:10.1109/SP.2013.15

Chang, C. C., & Lin, C. J. (2011). LIBSVM: A Library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, *2*(3), 1–39. https://doi.org/10.1145/1961189.1961199

Creswell, A., & Bharath, A. A. (2019). Inverting the Generator of a Generative Adversarial Network. *IEEE Transactions on Neural Networks and Learning Systems*, *30*(7), 1967–1974. https://doi.org/10.1109/TNNLS.2018.2875194

de Araujo-Filho, P. F., Kaddoum, G., Campelo, D. R., Santos, A. G., Macedo, D., & Zanchettin, C. (2020). Intrusion Detection for Cyber-Physical Systems using Generative Adversarial Networks in Fog Environment. *IEEE Internet of Things Journal, 4662*(C), 1–1. 10.1109/jiot.2020.3024800

Dingledine, R., Mathewson, N., & Syverson, P. (2004). Tor: The second-generation onion router. *Proceedings of the 13th USENIX Security Symposium*.

Galteland, H., & Gjøsteen, K. (2018). Adversaries monitoring Tor traffic crossing their jurisdictional border and reconstructing Tor circuits. *ArXiv*.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative Adversarial Networks. *Communications of the ACM*, *63*(11), 139–144. https://doi.org/10.1145/3422622

Guan, Z., Xiong, G., Li, Z., & Gou, G. (2020). ResTor: A Pre-Processing Model for Removing the Noise Pattern in Flow Correlation. *Proceedings - IEEE Symposium on Computers and Communications,* 1–6. 10.1109/ISCC50000.2020.9219544

Hayes, J., & Danezis, G. (2015). Guard Sets for Onion Routing. *Proceedings on Privacy Enhancing Technologies*, *2015*(2), 65–80. https://doi.org/10.1515/popets-2015-0017

Ian Goodfellow and Yoshua Bengio and Aaron Courville. (2016). *Deep Learning*. MIT Press. http://www.deeplearningbook.org

Imani, M., Barton, A., & Wright, M. (2018). Guard Sets in Tor using AS Relationships. *Proceedings on Privacy Enhancing Technologies*, *2018*(1), 145–165. https://doi.org/10.1515/popets-2018-0008

Jaggard, A. D., & Syverson, P. (2017). Onions in the crosshairs: When the man really is out to get you. *WPES 2017 - Proceedings of the 2017 Workshop on Privacy in the Electronic Society, Co-Located with CCS 2017,* 141–151. 10.1145/3139550.3139553

Jansen, R., & Hopper, N. (2012). Shadow: Running Tor in a Box for Accurate and Efficient Experimentation. *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, 1–22.

Jansen, R., Juarez, M., Galvez, R., Elahi, T., & Diaz, C. (2018). Inside Job: Applying Traffic Analysis to Measure Tor from Within. *Proceedings 2018 Network and Distributed System Security Symposium*. doi:10.14722/ndss.2018.23261

Johnson, A., Jansen, R., Hopper, N., Segal, A., & Syverson, P. (2017). PeerFlow: Secure Load Balancing in Tor. *Proceedings on Privacy Enhancing Technologies*, *2017*(2), 74–94. https://doi.org/10.1515/popets-2017-0017

Kwon, A., Alsabah, M., Lazar, D., Dacier, M., & Devadas, S. (2015). Circuit Fingerprinting Attacks : Passive Deanonymization of Tor Hidden Services. *Proceedings of the 24th USENIX Security Symposium*, 287–302. https://www.usenix.org/system/files/conference/usenixsecurity15/sec15-paper-kwon.pdf

Li, Y., Swersky, K., & Zemel, R. (2015). Generative moment matching networks. *32nd International Conference on Machine Learning, ICML 2015*, *3*, 1718–1727.

Nasr, M., Bahramali, A., & Houmansadr, A. (2018). DeepCorr: Strong Flow Correlation Attacks on Tor Using Deep Learning. *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, 1962–1976. 10.1145/3243734.3243824

OnionShare v2.2.1. (n.d.). https://onionshare.org/

Platzer, F., Schäfer, M., & Steinebach, M. (2020). Critical traffic analysis on the tor network. *ACM International Conference Proceeding Series*. 10.1145/3407023.3409180

Sun, Y., Edmundson, A., Vanbever, L., Li, O., Rexford, J., Chiang, M., & Mittal, P. (2015). Raptor: Routing attacks on privacy in tor. *Proceedings of the 24th USENIX Security Symposium*, 271–286.

Zhu, Y., Fu, X., Graham, B., Bettati, R., & Zhao, W. (2010). Correlation-Based Traffic Analysis Attacks on Anonymity Networks. *IEEE Transactions on Parallel and Distributed Systems*, *21*(7), 954–967. https://doi.org/10.1109/TPDS.2009.146

Zhuo, Z., Zhang, Y., Zhang, Z., Zhang, X., & Zhang, J. (2018). Website Fingerprinting Attack on Anonymity Networks Based on Profile Hidden Markov Model. *IEEE Transactions on Information Forensics and Security*, *13*(5), 1081–1095. https://doi.org/10.1109/TIFS.2017.2762825

Yitong Meng received his master's degree in 2017 and is currently a PhD student. His research interests include network security, encrypted traffic analysis, and anonymous networks.

Jinlong Fei is an associate professor at the State Key Laboratory of Mathematical Engineering and Advanced Computing. His research interests include network security, privacy protection, and encrypted traffic analysis.