symmetry

MDPI

Article

PeerAmbush: Multi-Layer Perceptron to Detect Peer-to-Peer Botnet

Arkan Hammoodi Hasan Kabla [1], Achmad Husni Thamrin [2], Mohammed Anbar [1], Selvakumar Manickam [1] and Shankar Karuppayah [1,*]

[1] National Advanced IPv6 Centre (Nav6), Universiti Sains Malaysia, Pulau Pinang 11800, Malaysia
[2] Graduate School of Media and Governance, Keio University, Tokyo 108-8345, Japan
* Correspondence: kshankar@usm.my

Abstract: Due to emerging internet technologies that mostly depend on the decentralization concept, such as cryptocurrencies, cyber attackers also use the decentralization concept to develop P2P botnets. P2P botnets are considered one of the most serious and challenging threats to internet infrastructure security. Consequently, several open issues still need to be addressed, such as improving botnet intrusion detection systems, because botnet detection is essentially a confrontational problem. This paper presents PeerAmbush, a novel approach for detecting P2P botnets using, for the first time, one of the most effective deep learning techniques, which is the Multi-Layer Perceptron, with certain parameter settings to detect this type of botnet, unlike most current research, which is entirely based on machine learning techniques. The reason for employing machine learning/deep learning techniques, besides data analysis, is because the bots under the same botnet have a symmetrical behavior, and that makes them recognizable compared to benign network traffic. The PeerAmbush also takes the challenge of detecting P2P botnets with fewer selected features compared to the existing related works by proposing a novel feature engineering method based on Best First Union (BFU). The proposed approach showed considerable results, with a very high detection accuracy of 99.9%, with no FPR. The experimental results showed that PeerAmbush is a promising approach, and we look forward to building on it to develop better security defenses.

Keywords: P2P networks; P2P botnets; intrusion detection systems; feature engineering; Multi-Layer Perceptron; deep learning

Citation: Kabla, A.H.H.; Thamrin, A.H.; Anbar, M.; Manickam, S.; Karuppayah, S. PeerAmbush: Multi-Layer Perceptron to Detect Peer-to-Peer Botnet. *Symmetry* **2022**, *14*, 2483. https://doi.org/10.3390/sym14122483

Academic Editor: Marcin Michalak

Received: 28 October 2022
Accepted: 15 November 2022
Published: 23 November 2022

1. Introduction

Typically, a bot is a compromised machine remotely controlled by a botmaster. A network of such infected machines under the command of a botmaster is called a botnet [1]. The compromised end-hosts are exploited in order to steal data or to launch Distributed Denial of Service (DDoS) attacks [2]. In their different scenarios, botnets have recently led to more threats to internet infrastructure security. Botnets are the reason behind many malware attacks, such as cryptocurrency mining, click fraud, and DDoS attacks [3]. The peculiarity of the botnet is its ability to launch large-scale, stealthy, and highly coordinated attacks [1]. Thus, taking down the botnet or even detecting the botnet is very challenging.

Traditionally, a botnet starts when the attacker, called the botmaster, infects many machines, called bots, over the network with various viruses, worms, and trojan horses and then commands and controls them remotely to coordinate a large-scale attack [1]. For that, we can summarize the main three components of botnets: Bot (master/operator), Command-and-Control mechanism (C2), and Malware (malicious software) [4]. Intuitively, the larger the botnet, the more challenging it is to counter. However, there is centralization (master) in that attack, where security guards could target it to weaken the whole thread. However, there is the issue of decentralized botnets, where there is no centralization to counter. It is important to consider that botmasters also evolve their mechanisms for command-and-control purposes. Consequently, some botnets utilize the concept of avoiding the single point of failure where the non-centralization leads to the attack [5].

Therefore, it is clearly necessary to categorize the botnet into centralized and decentralized botnets [6]. Examples of centralized botnets are HTTP-based and IRC-based botnets [7]. The HTTP-based botnets use HTTP messages to hide their commands, whereas the IRC-based botnets are considered the most widespread botnets, and both HTTP-based and IRC-based botnets have a simple structure. As previously explained, HTTP-based and IRC-based suffer from a centralized nature, i.e., they are under the risk of a single point of failure.

In contrast, decentralized botnets such as Peer-to-Peer (P2P) botnets operate in a distributed style where there is no central server to lead the command-and-control to launch an attack [4,8]. Reasonably, due to the distributed nature of P2P botnets, these botnets are comparatively harder to counter or detect. The idea of P2P botnets is that the attackers leverage the features of the P2P network in order to construct a robust net of bots, avoiding the risk of a single point of failure and giving permissions to peers to act as client and server simultaneously [9,10].

Examples of P2P botnets are Nugache [11], Zeus GameOver [12], Slapper [13], Storm [14], ZeroAccess, DDG [15], and the very sophisticated FritzFrog [16], etc. These botnets have a bad history, with many victims, leading to large financial losses [17]. So far, there are two P2P botnets: P2P botnets that either use specific P2P protocol for a special purpose or adopt the public P2P protocols, i.e., parasite botnets [18]. When P2P botnets utilize a specifically built private P2P protocol, these botnets are easier to detect, whereas the parasite P2P botnets adopt existing protocols and become stealthy and harder to detect [18].

Recently, P2P networks have become viral because of the ease of sharing resources. For example, cryptocurrencies are based on the decentralization concept, such as Ethereum, which is wholly based on a P2P network [19]. For that, attackers also aim to leverage the same characteristics of P2P networks, such as scalability, resilience, and efficiency, to attack important organizations, banks, institutions, companies, etc. Therefore, we take this challenge to benefit one of the most effective security defenses, the Intrusion Detection System (IDS), to build a Deep Learning (DL) model to detect P2P botnets. Machine Learning (ML) and DL techniques have proven their effectiveness in learning-based anomaly detection [20]. Some efficient ML techniques have shown considerable results in classifying network traffic, and this is one of the reasons behind employing these techniques to detect botnets, because P2P bots have symmetrical activities in the network. Thus, the symmetry in bots' behaviors makes them detectable compared to benign network traffic.

However, we found very few papers that attempted to detect the botnets using Deep Learning techniques. In contrast, DL techniques have not yet been completely leveraged in detecting P2P botnets. This is another challenge in developing the efficiency of DL techniques in this matter. Finally, this paper proposes a novel DL-based approach to detect P2P botnets using Multi-Layer Perceptron (MLP) as a DL classifier.

The rest of this paper is organized as follows. Section 2 exclusively reviews the works relevant to P2P botnet detection using ML and DL techniques and then critically discusses them. Section 3 presents the proposed methodology. Section 4 shows the implementation and experimental results and compares them with the existing works. Finally, Section 5 concludes our work and suggests future avenues.

2. Relevant Research and Critical Analysis

This section discusses the state-of-the-art of IDS solutions against P2P botnets using ML and DL techniques. Respectfully, this paper shows the findings and limitations of the relevant research in order to highlight the open issues, which may facilitate other researchers to work on the existing gaps. This section critically analyses the gaps in relevant research to develop a solution that contributes to the current body of knowledge and research.

2.1. Relevant Research

In the last decade, there has been a growing interest in detecting and preventing botnet techniques. These techniques include monitoring the botnet and learning how a bot infects

benign machines, but the challenge starts with detecting whether this machine is infected or not. Once a machine is detected as infected, many post-procedures must be taken to avoid exploiting this machine to enlarge the botnet. There have been many approaches developed to detect botnets. Based on a set of factors, these approaches can be classified as either signature-based or anomaly-based IDSs. In addition, there are three classes of botnet detection system based on location: host-based, network-based, and hybrid-based detection systems [21]. This paper covers only research that works on detecting P2P botnets using ML and DL techniques and then critically discusses the relevant research.

PeerRush was proposed to mine unwanted traffic in the P2P network [22]. Rahbarinia is meant to detect Storm, Zeus, and Waledac P2P botnets. The proposed solution performed cross-validation and showed a low misclassification rate of 0.68% and a low false positive of 0.1%.

Garg et al. [23] applied several ML techniques, such as nearest neighbor, Naive Bayes, and J48, to detect P2P botnets. In this experiment, the classifiers nearest neighbor and J48 performed better than other classifiers.

Jiang and Shao [24] proposed an approach to detect P2P botnets using an unsupervised ML technique. The proposed approach focuses more on the characteristics of command-and-control traffic. The author applied a clustering technique to distinguish between benign and P2P botnet flows.

Liao and Chang [25] proposed a methodology to distinguish between legitimate P2P traffic and P2P botnet traffic using the packet size. The author discovered that P2P botnets frequently try to update connection information for other bots rather than staying idle. Moreover, P2P botnets usually transmit data with a minimum rate of connection. The author applied Bayesian networks, Naïve Bayes, and J48 as classifiers to classify the network traffic. The J48 technique showed the highest detection rate compared to other classifiers.

Zhao and Traore [26] proposed an ML-based approach to detect P2P botnets by classifying the captured fast flux network flows. The authors applied a decision tree as a classifier to detect P2P botnets.

Alauthaman et al. [6] proposed a method to detect P2P botnets using a multilayer feed-forward neural network as well as decision trees. The regression tree is applied as a feature selection technique. Thereafter, the selected features feed the feed-forward neural network training. The proposed method achieved a high detection accuracy of 99% and a low FPR of 0.7%.

Yang and Wang [27] implemented an ML technique to detect P2P botnet for DDoS attacks. The authors also proposed a feature extraction method using the graph symmetry concept. This paper took the packets as subdivisions of the signal, and the time interval and data packet were the corresponding two-dimensional features.

Yin proposed a node-based detection approach to detect P2P botnet characteristics [28]. Yin focused more on the network characteristics of an individual node by examining the node flows in order to extract the significant features. The author then utilized an ML classifier to detect the bots.

Priyanka presented a two-tier detection scheme to detect parasite P2P botnets in their waiting stage [17]. The authors considered only two essential behaviors: the search requests' intensity and the long-living peers. However, that does not reflect the whole network of the compromised machines.

Table 1 summarizes the findings, limitations, and other details of all the relevant research.

Table 1. Summary of relevant research.

Article	Technique	Performance Metrics	Findings	Limitations
[22]	SVM	Accuracy, FPR, TPR	• Proposing an approach that achieved low misclassification in detecting three types of P2P botnet.	• Small lab dataset that consists of only 11 hosts; • The used hosts were running limited P2P applications, mostly Skype traffic.
[23]	Nearest NeighborNaive BayesJ48	TP, FP, Time	• The authors experimented with different ML techniques in detecting the P2P botnet and compared their abilities in classifying this kind of botnet.	• Detection of legitimate traffic is very weak.
[24]	Hierarchical clustering dendrogram	FPR, Detection Rate	• Detecting P2P botnets by discovering flow dependencies.	• The proposed approach will fail to detect the botnets that have irregularity in their traffic flow, such as Storm, because this method was built based on the similarity of botnet traffic.
[25]	Bayesian networksNaïve BayesJ48	Accuracy, FP, FN	• Proposing a methodology to detect P2P botnets using ML techniques and achieving a high detection rate.	• Research was conducted only for the LAN environment; • No details were mentioned regarding the used dataset; • Compared to other solutions, this methodology is considered complex because it deals with whole features and costs the model time and resources.
[26]	Decision Tree	TP, TN	• Proposing a P2P detecting system by identifying the malicious fast-flux networks.	• It is based on low Time to Live; once the TTL reaches zero, then packets are discarded, which leads to losing some of the network information; • Limited record of the used dataset; • The major evaluation metrics are missed.
[6]	Neural Network	Accuracy, FPR	• Based on multilayer NN, the proposed method achieved a high detection rate of 99%.	• The used dataset (ISOT) consists of only DNS malicious traffic.
[27]	K-NearestREP TreeSVM	Accuracy, Recall, FPR	• Proposing a new feature extraction method using the graphic symmetry concept to detect the P2P botnet.	• The used dataset (ISOT) consists of only DNS malicious traffic.
[28]	Decision Tree	Accuracy, Precision, FPR, TPR	• Proposing an approach based on ML classifier to detect the P2P botnets through the node level; • The proposed approach achieved a high detection rate.	• Storage overhead and major computational are required to process the constant flows at the node without even feature engineering; • The author used sampling, which does not detect the same number of botnets as detected using constant flow monitoring.
[17]	MultiBoostABDecisionStump	Accuracy, FPR, TPR	• Detecting parasite P2P botnets using machine learning classifiers.	• No feature engineering was performed; this approach costs resources and time; • The authors used the same dataset of [22], which is small and limited in terms of the traffic type.

2.2. Critical Analysis

Critically, we analyze the relevant research respectfully. We did our best to exhaustively cover all the works that work on detecting P2P botnets using ML and DL techniques through the top scientific research repositories, namely: ScienceDirect, WoS, Springer- Link, Wiley, IEEE, and several flagship conferences in the domain of security according to the CORE2021 ranking, which are ACM, IEEE SP, NDSS, and Usenix-Security. For integrity,

some interesting works detected P2P botnets and achieved a high detection rate. However, we still derived some concerns from the existing good works. In general, we cluster the concerns into three, as follows.

2.2.1. Dataset Concern

As cyber security researchers, we know that compiling a real network dataset is difficult for many reasons, such as anonymity and privacy considerations. For this reason, we found that most of the existing datasets are simulated datasets. The argument is not about whether it is a real network dataset or a simulated one, but rather it is about the way the dataset is constructed. Constructing a dataset for IDS experiments is quite important because the dataset is the crucial norm to evaluate the effectiveness of the proposed IDs. For example, there is no problem if the proposed IDS was wisely and smoothly built, but if we look back at the used dataset to evaluate this IDS and we find it was rashly constructed, we no longer trust it. Some issues in the data reflected in the final evaluation, such as the imbalanced dataset, finally led to an overfitting problem [29]. This paper will not discuss other problems, such as small-size datasets, unknown-source datasets, and unavailability, etc.

The problem we found in most of the existing works is that the used datasets were incomplete, or they did not include real network features. It is agreed that the dataset must contain attack traffic mixed with background traffic in order to make the proposed model/approach/system learn more about both normal and abnormal traffic. For example, some existing works evaluated their experiments using the CTU-13 dataset, such as [30], and after analyzing this dataset, we found that it does not contain background traffic. For that, we conducted a simple assessment survey of the existing datasets that has P2P botnet traffic by checking Google data [31], Mendeley data [32], and Kaggle [33]. Based on a checklist inspired by Susan McGregor [34], we chose a dataset to evaluate our proposed approach. The quality assessment included checking for certain points, such as: is it of known pedigree? Is it complete? Is it high-volume? Is it consistent? Is it dimensionality structured? The dataset was also assessed regarding the data fit via validity, reliability, and representativeness. Table 2 summarizes the existing datasets that contain botnet traffic.

Table 2. Summary of the botnet datasets.

Dataset	Description	Limitation
DCNDS [35]	• Project dataset, P2P botnet; • Detection using enhanced peer hunter.	• No PCAP files were provided; • No background flows.
CTU-13 [36]	• Includes 13 scenarios of different botnet samples, such as the P2P botnet; • Many protocols were considered, such as ICMP, TCP, and DNS.	• No background flows.
VHS-22 [37]	• Mixed flows of botnets from other datasets such as ISOT, CICIDS, CTU-13, and MTA with legitimate traffic;	• Only a CSV file was provided.
MTA-KDD-19 [38]	• Malware Traffic Analysis Knowledge Dataset.	• Only a CSV file was provided; • Small dataset.
TrendMicro [39]	• CTF Wildcard botnet dataset 400; • Contains only the following features: timestamp, source, destination, port, bytes.	• Only five features were provided (limited images of the network); • Only a CSV file was provided.
P2P-BDS [21]	• Based on the article: Peer-2-Peer botnet detection system.	• No longer reachable.
ISOR [40]	• Based on the article [40].	• No longer reachable.
ISOT [41]	• Botnet dataset.	• Contains only traffic passed from/to DNS.

It should be noted that we did not cover the IoT and Android botnet datasets. Moreover, we did not consider the datasets that were only provided as CVS files because the CSV file reflects a limited network traffic image. Therefore, we listed providing only CSV

files as a limitation in our work. Another concern in the datasets is that most of the relevant works did not show whether the used dataset was balanced. Ignoring such norms might show good experimental results, but they would be fake. Overall, we had to construct a new and solid dataset (Section 3.1) to avoid the existing concerns.

2.2.2. Feature Engineering Concern

First, most of the authors of the relevant research did not engineer features where they handled all the features, and they had to then feed that large volume to the models/approaches/systems. Consequently, that cost resources and time and increased the complexity. Second, even authors who had feature engineering did not evaluate the utility of the proposed feature engineering methods before and after implementing those methods. Finally, some matters must be considered by the feature engineering method, such as the simplicity, applicability, and efficiency of the proposed solution. In this paper, we propose a novel, efficient, and applicable feature selection method to reduce the data dimensionality and speed up the whole data processing (Section 3.3). Moreover, the feature selection facilitates the targeted attack detection for the predictors in the next stage (detection stage) [42].

2.2.3. Detection Concern

All the utilized classifiers in the relevant works are ML techniques. Although some works achieved good detection rates, they still cannot prove their effectiveness because of the different data preparation, and major evaluation metrics were missed. In addition, some researchers repeated applying the same ML techniques, i.e., the existing works did not apply many different classifiers to understand each technique's performance and cover more probabilities. For example, [26–28] applied Decision Tree as a classifier to detect the P2P botnet, whereas [22] and [27] applied SVM as a classifier, and [23] and [25] applied Naïve Bayes as a classifier. Noticeably, there are many repetitions of applying the same ML classifiers. Furthermore, DL techniques have not been completely leveraged in detecting P2P botnets. Another concern is that not all of the authors mentioned the testing approach utilized in their experiments, whether it was Percentage-split or Cross-validation. For that, our proposed approach is based on the most effective DL technique, the MLP, and it is tested by both testing approaches: Percentage-split and Cross-validation.

3. PeerAmbush

In this paper, we propose a novel approach, PeerAmbush, to detect one of the most dangerous attacks, P2P botnets using the DL technique. PeerAmbush addresses some of the derived limitations of relevant research, as earlier discussed in Section 2.2, regarding the used datasets to evaluate the proposed solution, feature engineering influence, and detection performance. PeerAmbush consists of five stages: Data Construction, Data Preparation, Feature Engineering, MLP-based P2P Botnet Detection, and Evaluation Results. Each stage includes some substages before feeding the output as input to the following stage. Figure 1 shows an overview of PeerAmbush.

The key contributions of our proposed approach are as follows:

- Constructing a new dataset that includes P2P botnet traffic and background flow;
- Proposing a novel feature engineering method based on mathematical union theory to select the most significant features: Best First Union (BFU);
- Adapting the MLP as a classifier to detect P2P botnets.

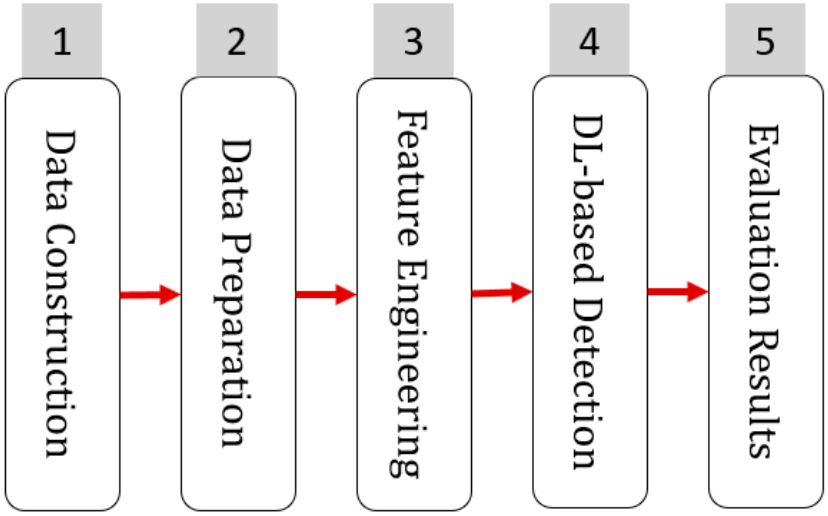

Figure 1. Overview of PeerAmbush.

3.1. Data Construction

As mentioned previously, generating a new dataset is challenging due to privacy matters. For this reason, we constructed a new dataset based on the reliable existing ones. One of the norms to assess the dataset quality was whether it was of a known pedigree. After that, we selected the CTU-13 dataset [36] for three reasons: (i) it is reliable, and many researchers have used this dataset to evaluate their solutions, (ii) it contains the P2P botnet scenario, and (iii) it was provided as PCAP files not CSV files, and that gives more understanding of the network traffic and the behavior of P2P botnets. However, this dataset was missing benign network flows. Hence, we merged the CTU-13 dataset with a recent network background flow collected from the HIKARI dataset [43]. As a result, the newly constructed dataset contains both the abnormal behavior of the P2P botnet and the normal flow of the network to let the trained model learn about both the normal and abnormal and then avoid the problem of overfitting. Figure 2 simplifies the process of this stage.

3.1.1. CTU-13 Dataset

This dataset is considered one of the most reliable datasets in the IDS community. CTU-13 has a large capture of real botnet traffic, including 13 different scenarios of different botnet types. This dataset contains many protocols, such as ICMP, TCP, DNS, etc. One of its useful features is that it was provided as PCAP files, allowing a realistic and deeper understanding of the network traffic. The PCAP files of the CTU-13 dataset also contain other types of information, such as NetFlow, Weblogs, etc. [36].

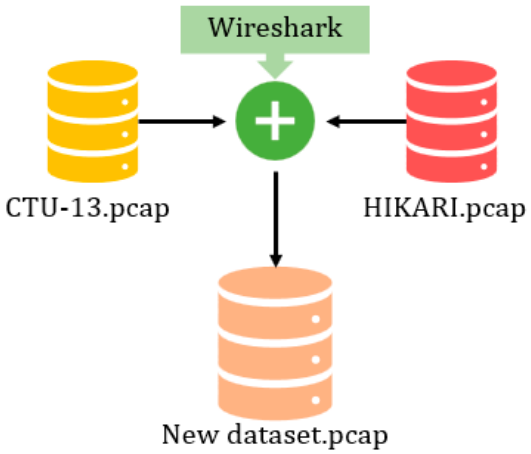

Figure 2. Data Construction.

3.1.2. HIKARI Dataset

This dataset was recently captured by Ferriyan [43], and it was decided to merge it with the CTU-13 to complete what was missing in the first dataset. The HIKARI dataset completely captures the network traffic, such as communication between hosts, broadcast messages, and domain lookup queries. We chose the ground-truth from this dataset because it provides realistic benign traffic from a real production network, not synthetic traffic, as found in some datasets [43].

3.2. Data Preparation

Data preparation means preparing the selected dataset for the next stages by some substages to make it fit the purpose of this research and be seen as readable by the next methods, i.e., the feature engineering method and the detection stage. The first procedure is filtration because, as mentioned previously, the CTU-13 dataset contains 13 different scenarios, and this approach concerns only P2P botnets (Scenario no. 12) [36]. Consequently, we exclude the other scenarios and keep the P2P botnet scenario as well as the ground-truth from the HIKARI dataset. After filtration, we label the filtered dataset to import it into a supervised learning-based model. We convert the labelled dataset into numeric data to make them readable by the next methods/algorithms (Numericalization). Finally, we normalize the dataset before taking it as input for the third stage. Figure 3 shows the process of the data preparation stage.

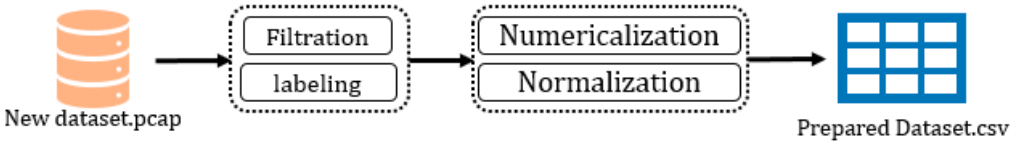

Figure 3. Process of data preparation.

3.3. Feature Engineering

Recently, researchers have widely applied feature engineering because it plays a vital role in their proposed solutions. There are many feature engineering methods, such as feature selection. Feature selection reduces the data dimensionality, saving time and reducing the proposed solutions' complexity [29]. Unfortunately, most of the relevant works had no feature engineering methods. Accordingly, those proposed solutions were more complex and sources/time-consuming.

In this paper, we propose a novel feature engineering method based on mathematical union theory to select the most significant features that reflect the influence of whole features and improves the predictor performance. We name the proposed method Best First Union (BFU). BFU starts with a feature evaluator to select the highest important features as the best first ones via two different methods: CFS Subset Evaluation and Consistency Subset Evaluation. The next subsections explain the two methods (Sections 3.3.1 and 3.3.2).

Each method evaluates the features differently and eventually provides a final shortlist of features. The resulting two shortlists will then be united to generate one shortlist that includes the most significant features and leverage two different evaluators. Later, only the selected features will go through the detection stage as inputs. Figure 4 simplifies the process of our novel feature engineering method, BFU.

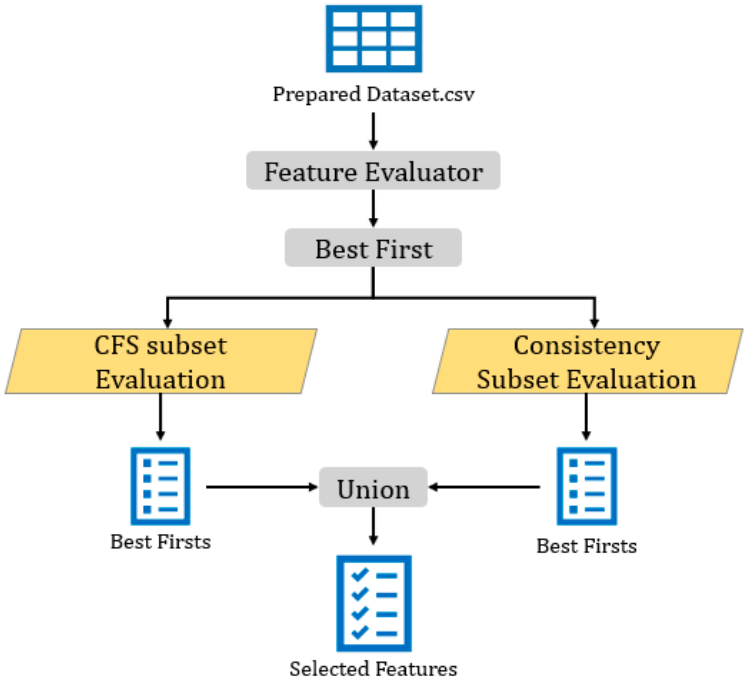

Figure 4. The process of BFU.

3.3.1. CFS Subset Evaluation

This method works on evaluating the worth of features by grouping them into subsets and then considering the individual predictive ability of each feature along with the degree of redundancy among features. Consequently, the features highly correlated with the class during low intercorrelation are preferred as the best first [44].

3.3.2. Consistency Subset Evaluation

This method works on evaluating the worth of features by grouping them into subsets and then measuring the level of consistency in the class values. Each subset has a consistency, and that consistency can never be lower than that of the full dataset [45].

3.4. MLP-Based P2P Botnet Detection

According to the relevant research, the behavior of P2P botnets is distinguishable from normal network traffic. Detecting the P2P botnets can be modelled as a multi-class classification task. We earlier prepared our dataset through some substages, such as labelling. Accordingly, we adopted a typical supervised deep learning technique, MLP, as a classifier to detect the P2P botnet. There are two reasons behind choosing this DL technique in this work: (i) there is no work yet that has applied DL techniques to detect P2P botnets, and (ii) this technique has proved its effectiveness in detection systems, i.e., there are some researchers who applied this technique to detect different types of attacks, such as [46–49], and their experiments show the high efficiency of MLP in network intrusion detection systems.

3.4.1. Multi-Layer Perceptron (MLP)

Multi-Layer Perceptron is a deep learning technique; it is considered one of the most efficient neural network techniques for classification in IDS [50]. MLP is a feed-forward and fully connected neural network. Figure 5 shows a simple hypothetical example of the architectural design of MLP.

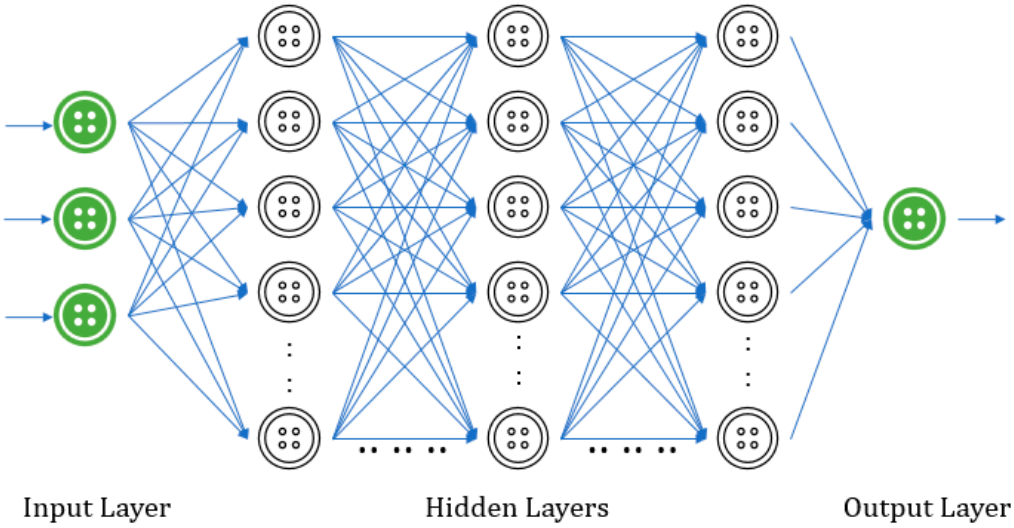

Figure 5. A hypothetical example of the architectural design of MLP.

The MLP takes the numeric and normalized values of the selected features, as prepared in the previous stages.

The most relevant works that applied MLP as a classifier did not mention the parameter settings. Some of them merely increases the number of hidden layers, which might cause overfitting, i.e., increasing the hidden layers may be illusory once it performs well on the training dataset. However, testing the trained model with a new dataset may show a disappointing performance. The number of hidden layers should be tunned along with the number of nodes and the dataset volume. Setting a number of hidden layers and nodes can reflect the performance quality and trade-off, minimizing the total error due to bias and variance. Often, the complex model leads to overfitting, while the simple ones fail to catch the relationship between the input and the output [51].

In this paper, we utilize both testing approaches: Percentage-split, and Cross-validation, as well as providing the parameter settings of MLP.

3.4.2. Percentage-Split

In this testing approach, the dataset is split into 80% and 20% of the dataset. The first one is used to train the MLP, while the second one is to test the effectiveness of MLP in detecting the intrusion with a new dataset (20%).

3.4.3. Cross-Validation

In this testing approach, the dataset is divided into 10 folds of cross-validation, where nine of them are to train the MLP and one fold is to test the effectiveness of MLP in detecting the intrusion. Figure 6 shows the difference between the two testing approaches.

In this paper, all the major metrics are used to evaluate the performance of employing the MLP to detect the P2P botnet, such as Detection Accuracy, FPR, Precision, Recall, and F-Score. Providing all the major metrics reflects the quality and effectiveness of the proposed approach.

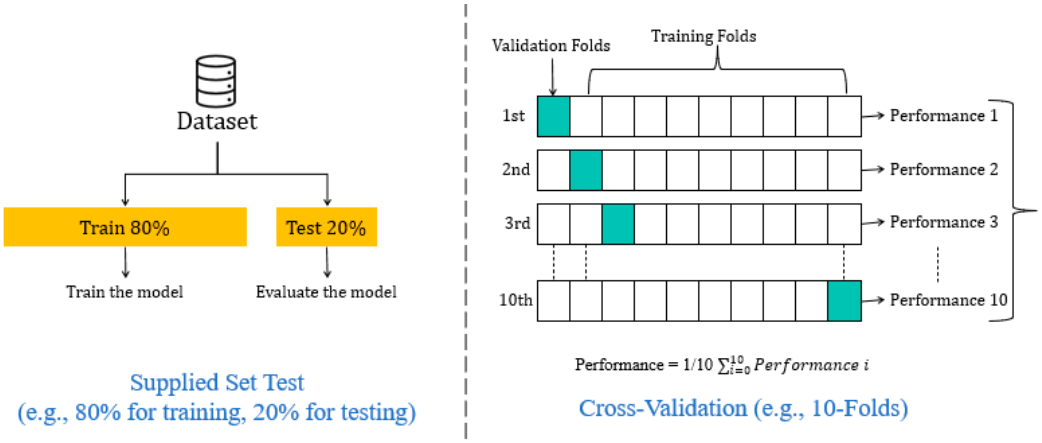

Figure 6. The two testing approaches.

3.5. Evaluation Metrics

Several key metrics were used to evaluate the performance of different models/approaches/systems. The evaluation metrics are similar to an accurate reading of the performance of the proposed solution. However, some standard metrics should be calculated, especially for comparison purposes, such as Accuracy, False Positive Rate, Precision, etc. This paper considers all the major metrics to evaluate PeerAmbush compared to other works. True Positive (*TP*) is the percentage of correctly predicted attacks, while True Negative (*TN*) is the percentage of correctly predicted normal instances of traffic. In comparison, False Positive (*FP*) and False Negative (*FN*) are the percentages of normal instances when they are predicted as an attack and the percentage of attacks that are predicted as normal instances, respectively [20,52]. The False Positive Rate is another major metric to evaluate the proposed approach; it is the percentage of normal instances when they are predicted as attacks. Equation (1) calculates the FPR, Equation (2) calculates the Precision, Equation (3) calculates the Recall, and Equation (4) calculates the F-Score, as follows [53].

$$\text{FPR} = \frac{FP}{TN + FP} \tag{1}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{3}$$

$$\text{F} - \text{Score} = \frac{Precision * Recall}{Precision + Recall} * 2 \tag{4}$$

4. Implementation and Experimental Results

This section describes the design, implementation, and experimental results of each stage in PeerAmbush through three subsections. PeerAmbush as a security solution against P2P botnets was thoroughly explained in the previous section. However, this section shows the output of each stage, starting with constructing a new dataset and ending with the evaluation results. Figure 7 shows the roadmap of PeerAmbush in detail.

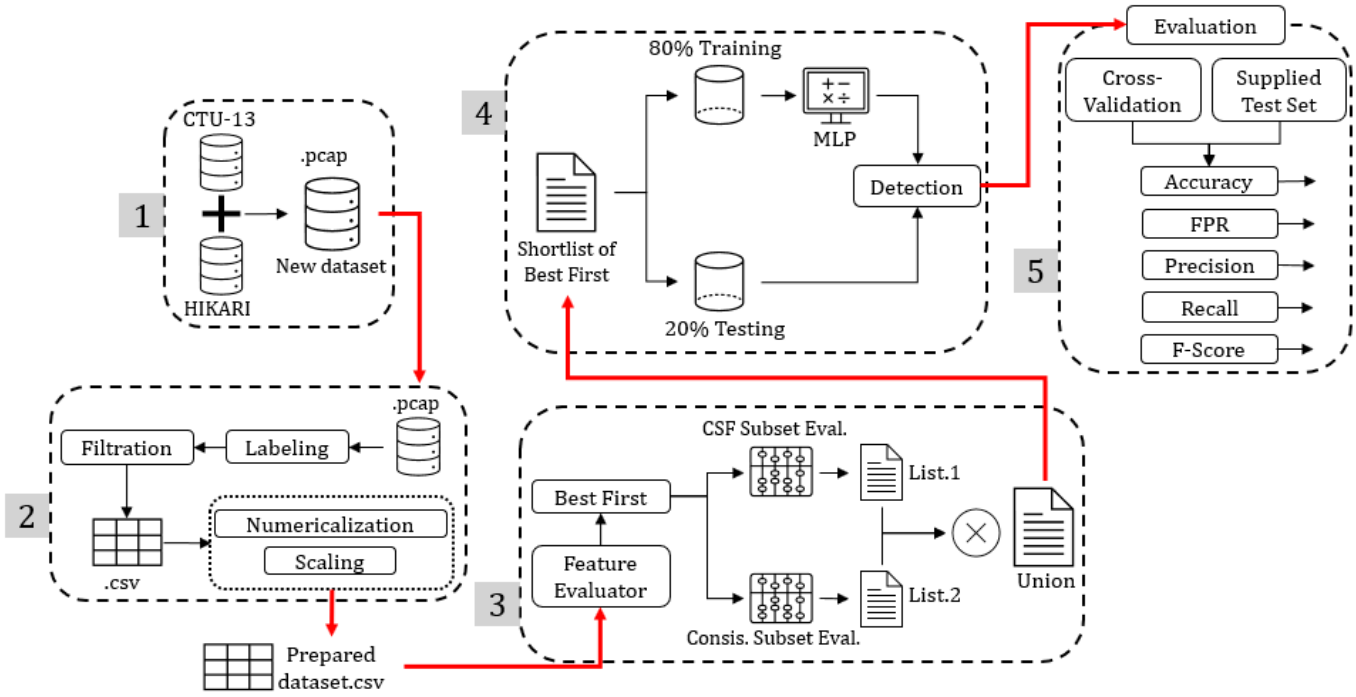

Figure 7. The roadmap of PeerAmbush.

4.1. Data Construction and Preparation (Stages 1–2)

As previously discussed in Section 2.2.1, there is a need to construct a new dataset that contains both the traffic of P2P botnets and benign network traffic. The concern was that Scenario no. 12 was selected for further analysis in our work, and we found that there were flows from the botmaster to the infected machines, and vice. In addition, this scenario also contains flows from 'infected machines' to 'non-infected' machines. Suppose we experimentally block the IP addresses of the botmaster and the infected machines. In that case, no traffic lasts, i.e., there is no normal traffic among only benign nodes (between non-infected machines). Therefore, for further understanding, we need to differentiate between the flows of infected and non-infected machines by their traffic. For additional information regarding CTU-13, Table 3 shows the IP addresses of the botmaster and the infected machines, and the dataset can then be found through [36].

Table 3. Types of flow from scenario no. 12 of the CTU-13 botnet dataset.

IP Address	Machine Role	Flow Direction
147.32.84.165	Botmaster	To/From infected and non-infected machines
147.32.84.165	Bot	To/From other infected machines and rarely to benign machines
147.32.84.191	Bot	To/From botmaster and rarely to non-infected machines
147.32.84.192	Bot	To/From botmaster and rarely to non-infected machines

Noticeably, we can see in Table 3 that there was no traffic flow from non-infected to non-infected machines. As a complement measure, we extracted the background of normal network traffic from HIKARI [43]. The number of extracted packets from CTU-13 is 352,266 packets. In contrast, the number of complement packets from the HIKARI dataset is 533,848 packets. Accordingly, the number of packets of the newly constructed dataset is 886,114 packets. The new dataset is labelled into multiclass; botmaster flow, infected machines flow (bots), and benign flow (i.e., multiclass labelling). After labelling, filtration keeps only the flows of scenario no.12 and drops the rest (scenarios 1–11 and 13). Finally, before converting the dataset to numeric data and applying the normalization, we checked

whether the dataset was balanced or not before proceeding to the next stage, and it was balanced. Table 4 describes the newly constructed dataset.

Table 4. Description of dataset.

Total number of records	886,114
Category	Multi-class
Classes	Botmaster, Bot, Normal
Number of botmaster/bots records	352,266
Number of normal records	533,848
Number of features	30

4.2. Feature Engineering (Stage 3)

The prepared dataset has 30 features. In our novel feature engineering method (BFU), all features are evaluated by a feature evaluator using two different methods (Sections 3.3.1 and 3.3.2) to select the best first. Each method then outcomes in a different shortlist representing the best first. Mathematically, A represents the best first list of CFS Subset Evaluation, and B represents the best first list of Consistency Subset Evaluation. Equation (5) calculates A union B, where x represents the feature [54].

$$A \cup B = \{x: x \in A \text{ or } x \in B\} \quad (5)$$

Consider the two lists of A and B, such that the number of features in the union of A and B can be calculated as follows in Equation (6) [54].

$$n(A \cup B) = n(A) + n(B) - n(A \cap B) \quad (6)$$

where $n(A \cup B)$ represents the total number of features in $A \cup B$; it is called the cardinality of a final list of $A \cup B$. Whereas $n(A)$ represents the number of features in A; it is called the cardinality of list A. Similarly, $n(B)$ represents the number of features in B, called the cardinality of list B. Moreover, $n(A \cap B)$ represents the number of features common to both A and B; it is called the cardinality of list $A \cap B$, i.e., A intersection B.

Table 5 summarizes the evaluation methods (Best First Evaluators).

Table 5. Summary of evaluation methods.

	Feature Evaluator-Method	
	CFS Subset Evaluation	**Consistency Subset Evaluation**
Search method	Best First	Best First
Search direction	Forward	Forward
No. of subset evaluated	171	178
Merit of best subset found	0.866	1
No. of features selected	3	2
Selected features	Source, Time to live, Epoch time	Source, Version

The above table shows that four features compose the final shortlist of features. Only these four features are considered to feed the detection stage. Comparatively, the BFU has achieved fewer selected features compared to the relevant research, as explained in Table 6. Selecting only four features represents approximately 12.5% of the whole original dataset composed of 30 features. Accordingly, this feature selection saves time and resources for the detection system, and indirectly that makes the process less complex and smoother. Furthermore, the next section (P2P botnet detection) shows the positive influence of using the BFU method compared to not using the BFU method, i.e., using the full dataset.

Table 6. Number of features in each work of the relevant works.

Article	[22]	[23]	[6]	[27]	[28]	[17,24–26]	BFU
No. of selected features	8	17	10	7	13	-	4

4.3. Evaluation Results of MLP-Based P2P Botnet Detection

This section shows the parameter settings and experimental results of PeerAmbush to detect P2P botnets. In addition, this section provides a comparison with respect to the relevant works. As mentioned previously, we tested PeerAmbush using two testing approaches: Percentage-split and Cross-validation. Furthermore, the parameter settings that we set to MLP are as follows. The number of training instances utilized in one iteration is 100 (officially called the patch size). There are ten hidden layers in our neural network. Experimentally, we slightly increased/decreased the number of hidden layers and the nodes in each layer until we achieved the highest detection rate, considering also the time taken to build a model and, as mentioned previously, that it is not recommended to keep increasing the number of hidden layers to avoid the overfitting problem [51]. Furthermore, we set 0.5 as the learning rate for updating the weights of nodes. The momentum that is applied to weight updates is 0.2. Last but not least, the training time is measured by the number of epochs to train through, and it is 500 epochs. Our parameter settings achieved better results compared to the relevant research. PeerAmbush achieved a high detection accuracy of 99.9%, and no FPR. Meanwhile, the default parameter settings achieved a detection accuracy of 96.5%, with higher false positive alarms compared to our parameter settings. Table 7 shows the parameter settings of MLP.

Table 7. The Parameter settings of MLP.

Parameter	Value
Batch size	100
Hidden Layers	10
Learning Rate	0.5
Momentum	0.2
Training Time	500

Table 8 summarizes the experimental results of our proposed PeerAmbush approach compared to the most recent works (last five years). No relevant works have yet leveraged DL techniques to detect P2P botnets. Consequently, there was a vital need to employ efficient techniques such as MLP to detect one of the most serious threats, P2P botnets. In addition, none of the relevant works have tested the proposed solution with the two different testing approaches to show its effectiveness. Moreover, most of the relevant works did not provide many of the major evaluation metrics, which is what causes us to doubt the solution. Thus, we provide all the major evaluation metrics, which show the superiority of PeerAmbush in Accuracy, FPR, Precision, Recall, and F-Score using two different testing approaches: Percentage-split and Cross-validation.

Table 9 comparatively shows the performance of MLP with our parameter settings and the performance of the best ML techniques that have been applied by the relevant research using the same dataset (our newly constructed dataset).

To conclude, some ML techniques did well in detecting P2P botnets in terms of detection accuracy, as shown in the relevant research. However, IDSs are not only based on intrusion detection accuracy, and there are some matters that should be considered to improve the overall performance, such as time and complexity. In this work, we proposed a novel feature engineering methods to select the most significant features. The proposed feature selection method eventually produced only four features to the predictor, and that contributed to reducing the data dimensionality and then reducing the process complexity.

Experimentally, we also used the full-features dataset to show the positive influence of our feature engineering method (BFU, Stage 3). Comparatively, the results of using the BFU method are better for many reasons: higher detection accuracy, lower FPR, higher Precision and Recall, and the time taken to build a model is less than the time taken when we use the full dataset. Table 10 comparatively shows the results of using the BFU method and without using the BFU method.

Table 8. The experimental results of PeerAmbush compared to the relevant works.

Article	Technique	Testing Approach	ACC (%)	FPR	Precision	Recall	F-Score	Others
[6]	NN	Cross-validation	99.0	0.75	-	-	-	-
[27]	K-Nearest	Cross-validation	76.5 79.1	0.06	-	0.82 0.85	-	-
	REP Tree	Cross-validation	96.1 97	0.01 0.02	-	0.96 0.97	-	-
	SVM	Cross-validation	88 89	0.06 0.05	-	0.82 0.91	-	-
Peer-Ambush	MLP	Percentage-split	99.9	0.0	1.0	1.0	1.0	TPR = 1.0 ROC area = 1.0
		Cross-validation	99.9	0.0	1.0	1.0	1.0	TPR = 1.0 ROC area = 1.0

ACC: Detection Accuracy, FPR: False Positive Rate, TPR: True Positive Rate, SVM: Support Vector Machine, MLP: Multi-Layer Perceptron, NN: Neural Network.

Table 9. The performance of MLP compared the used ML techniques using the same dataset.

Technique	Accuracy (%)	FPR	Recall
DecisionStump	82.1	0.040	0.82
AdaBoostAB	95.4	0.009	1.0
SVM	88.8	0.06	0.88
Neural Network	91.81	0.016	0.91
MLP	99.9	0.001	1.0

Table 10. Comparison between the results with BFU and without BFU.

	Using BFU	Without Using BFU
Number of features	4	30
Detection Accuracy (%0)	99.9	96.5
FPR	0.001	0.007
Precision	1.0	0.96
Recall	1.0	0.96

The above table shows the power of using the BFU. There are some features that may mislead or affect the classification process. For that, feature engineering selected only the most significant features that reflected the worth of the whole set and also benefited the predictor for better classification.

In general, this approach achieved higher detection accuracy and no FPR with fewer selected features compared to the relevant works. Last but not least, this paper shows the performance of one of the most effective DL classifiers, which is the MLP with a certain parameter setting to detect P2P botnets. Finally, the experimental results are promising to build and improve new IDSs to detect the P2P botnets.

5. Conclusions and Future Work

In conclusion, we proposed PeerAmbush as a novel DL-based approach to detect one of the most serious attacks, the P2P botnets. The proposed approach addresses some of the limitations in the relevant research, such as the dataset issues and feature engineering matter, by constructing a new dataset and proposing a novel feature engineering method, respectively. The novel feature engineering method is based on Best First Union, though we named it BFU, and this method selected only four features as the best to detect P2P botnets. Therefore, PeerAmbush employs the MLP as a DL classifier to classify the network traffic because the relevant works have not yet leveraged the DL techniques, and to benefit the effectiveness of DL in this case. The proposed approach consists of five main stages; each stage has substages to solve a certain issue and prepare the dataset for the next stage. PeerAmbush showed impressive results compared to the relevant research in terms of detection accuracy, FPR, Precision, Recall, and F-Score by only using four features as the fewest number of selected features compared to the relevant works. In the future, we look forward to specifically detecting more new P2P botnet types, such as DDG P2P botnet or the very sophisticated FritzFrog P2P botnet. Henceforward, we will also work on employing more DL techniques to detect more types of P2P botnet. We also plan to complete this work by improving a prevention technique against the detected traffic.

Author Contributions: Conceptualization, A.H.H.K. and A.H.T.; methodology, A.H.H.K.; software, A.H.H.K.; validation, A.H.H.K., A.H.T. and S.K.; investigation, A.H.T. and S.K.; data curation, A.H.H.K.; writing—original draft preparation, A.H.H.K.; writing—review and editing, A.H.T. and S.K.; supervision, A.H.T., M.A., S.M. and S.K.; project administration, A.H.T. and S.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Ministry of Higher Education Malaysia's Fundamental Research Grant Scheme under Grant FRGS/1/2021/ICT07/USM/03/1.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Karuppayah, S. *Advanced Monitoring in P2P Botnets A Dual Perspective*; Springer: Singapore, 2018; ISBN 9789811090493.
2. Karuppayah, S.; Manickam, S.; Böck, L.; Grube, T.; Mühlhäuser, M.; Fischer, M. SensorBuster: On Identifying Sensor Nodes in P2P Botnets. In Proceedings of the 12th International Conference on Availability, Reliability and Security, Reggio Calabria Italy, 29 August–1 September 2017; pp. 1–6. [CrossRef]
3. Lo, W.W.; Layeghy, S.; Sarhan, M.; Portmann, M. XG-BoT: An Explainable Deep Graph Neural Network for Botnet Detection and Forensics. *arXiv* **2022**, arXiv:2207.09088. Available online: https://arxiv.org/abs/2207.09088 (accessed on 13 October 2022).
4. Karuppayah, S.; Roos, S.; Rossow, C.; Muhlhauser, M.; Fischer, M. Zeus Milker: Circumventing the P2P Zeus Neighbor List Restriction Mechanism. In Proceedings of the 2015 IEEE 35th International Conference on Distributed Computing Systems, Columbus, OH, USA, 29 June–2 July 2015; pp. 619–629. [CrossRef]
5. Su, S.C.; Chen, Y.R.; Tsai, S.C.; Lin, Y.B. Detecting P2P Botnet in Software Defined Networks. *Secur. Commun. Netw.* **2018**, *2018*, 4723862. [CrossRef]
6. Alauthaman, M.; Aslam, N.; Zhang, L.; Alasem, R.; Hossain, M.A. A P2P Botnet Detection Scheme Based on Decision Tree and Adaptive Multilayer Neural Networks. *Neural Comput. Appl.* **2018**, *29*, 991–1004. [CrossRef] [PubMed]
7. Khan, R.U.; Zhang, X.; Kumar, R.; Sharif, A.; Golilarz, N.A.; Alazab, M. An Adaptive Multi-Layer Botnet Detection Technique Using Machine Learning Classifiers. *Appl. Sci.* **2019**, *9*, 2375. [CrossRef]
8. Zhang, J.; Perdisci, R.; Lee, W.; Luo, X.; Sarfraz, U. Building a Scalable System for Stealthy Peer to Peer Botnet Detection. *IEEE Trans. Inf. Forensics Secur.* **2014**, *2*, 6–10.
9. Karuppayah, S.; Vasilomanolakis, E.; Haas, S.; Muhlhauser, M.; Fischer, M. BoobyTrap: On Autonomously Detecting and Characterizing Crawlers in P2P Botnets. In Proceedings of the 2016 IEEE International Conference on Communications (ICC), Kuala Lumpur, Malaysia, 22–27 May 2016. [CrossRef]
10. Karuppayah, S.; Fischer, M.; Rossow, C.; Muhlhauser, M. On Advanced Monitoring in Resilient and Unstructured P2P Botnets. In Proceedings of the 2014 IEEE International Conference on Communications (ICC), Sydney, NSW, Australia, 10–14 June 2014; pp. 871–877. [CrossRef]
11. Stover, S.; Dittrich, D.; Hernandez, J.; Dietrich, S. Analysis of the Storm and Nugache Trojans: P2P Is Here. *USENIX Login* **2007**, *32*, 18–27.

12. Andriesse, D.; Rossow, C.; Stone-Gross, B.; Plohmann, D.; Bos, H. Highly Resilient Peer-to-Peer Botnets Are Here: An Analysis of Gameover Zeus. In Proceedings of the 2013 8th International Conference on Malicious and Unwanted Software: "The Americas" (MALWARE), Fajardo, PR, USA, 22–24 October 2013; pp. 116–123. [CrossRef]
13. Arce, I.; Levy, E. An Analysis of the Slapper Worm. *IEEE Secur. Priv.* **2003**, *1*, 82–87. [CrossRef]
14. Yen, T.F.; Reiter, M.K. Are Your Hosts Trading or Plotting? Telling P2P File-Sharing and Bots Apart. In Proceedings of the 2010 IEEE 30th International Conference on Distributed Computing Systems, Genoa, Italy, 21–25 June 2010; pp. 241–252. [CrossRef]
15. Tara Seals Unique P2P Architecture Gives DDG Botnet "Unstoppable" Status | Threatpost. Available online: https://threatpost.com/p2p-ddg-botnet-unstoppable/154650/ (accessed on 8 November 2022).
16. Jake Aurand FritzFrog P2P Botnet Attacking Healthcare, Education and Government Sectors—Binary Defense. Available online: https://www.binarydefense.com/threat_watch/fritzfrog-p2p-botnet-attacking-healthcare-education-and-government-sectors/ (accessed on 8 November 2022).
17. Priyanka; Dave, M. PeerFox: Detecting Parasite P2P Botnets in Their Waiting Stage. In Proceedings of the 2015 International Conference on Signal Processing, Computing and Control (ISPCC), Waknaghat, India, 24–26 September 2015; pp. 350–355. [CrossRef]
18. Rodríguez-Gómez, R.A.; Maciá-Fernández, G.; García-Teodoro, P.; Steiner, M.; Balzarotti, D. Resource Monitoring for the Detection of Parasite P2P Botnets. *Comput. Netw.* **2014**, *70*, 302–311. [CrossRef]
19. Kabla, A.H.H.; Anbar, M.; Manickam, S.; Alamiedy, T.A.; Cruspe, P.B.; Al-Ani, A.K.; Karupayah, S. Applicability of Intrusion Detection System on Ethereum Attacks: A Comprehensive Review. *IEEE Access* **2022**, *10*, 71632–71655. [CrossRef]
20. Alamiedy, T.A.; Anbar, M.F.; Belaton, B. Ensemble Feature Selection Approach for Detecting Denial of Service Attacks in RPL Networks. *Int. J. Eng. Res.* **2021**, *V7*, 21. [CrossRef]
21. Kaur, N.; Behal, S. P2P-BDS: Peer-2-Peer Botnet Detection System. *IOSR J. Comput. Eng.* **2014**, *16*, 28–33. [CrossRef]
22. Rahbarinia, B.; Perdisci, R.; Lanzi, A.; Li, K. Peer Rush: Mining for Unwanted P2P Traffic. *J. Inf. Secur. Appl.* **2014**, *19*, 194–208. [CrossRef]
23. Garg, S.; Singh, A.K.; Sarje, A.K.; Peddoju, S.K. Behaviour Analysis of Machine Learning Algorithms for Detecting P2P Botnets. In Proceedings of the 2013 15th International Conference on Advanced Computing Technologies (ICACT), Rajampet, India, 21–22 September 2013; pp. 1–3. [CrossRef]
24. Jiang, H.; Shao, X. Detecting P2P Botnets by Discovering Flow Dependency in C&C Traffic. *Peer-to-Peer Netw. Appl.* **2014**, *7*, 320–331. [CrossRef]
25. Liao, W.H.; Chang, C.C. Peer to Peer Botnet Detection Using Data Mining Scheme. In Proceedings of the 2010 International Conference on Internet Technology and Applications, Wuhan, China, 20–22 August 2010; pp. 1–3. [CrossRef]
26. Zhao, D.; Traore, I. P2P Botnet Detection through Malicious Fast Flux Network Identification. In Proceedings of the 2012 Seventh International Conference on P2P, Parallel, Grid, Cloud and Internet Computing, Victoria, BC, Canada, 12–14 November 2012; pp. 170–175. [CrossRef]
27. Yang, Z.; Wang, B. A Feature Extraction Method for P2P Botnet Detection Using Graphic Symmetry Concept. *Symmetry* **2019**, *11*, 326. [CrossRef]
28. Yin, C. Towards Accurate Node-Based Detection of P2P Botnets. *Sci. World J.* **2014**, *2014*, 425491. [CrossRef]
29. Kuhn, M.; Johnson, K. *Feature Engineering and Selection: A Practical Approach for Predictive Models*; CRC Press, Taylor & Francis Group: Boca Raton, FL, USA, 2019; ISBN 9781351609470.
30. Xing, Y.; Shu, H.; Kang, F.; Zhao, H. Peertrap: An Unstructured P2P Botnet Detection Framework Based on SAW Community Discovery. *Wirel. Commun. Mob. Comput.* **2022**, *2022*, 1–18. [CrossRef]
31. Dataset Search. Available online: https://datasetsearch.research.google.com/ (accessed on 12 October 2022).
32. Mendeley Data. Available online: https://data.mendeley.com/ (accessed on 12 October 2022).
33. Kaggle. Available online: https://www.kaggle.com/datasets (accessed on 12 October 2022).
34. McGregor, S.E. *Practical Python Data Wrangling & Data Quality*; O'Reilly Media, Inc.: Sevastopol, CA, USA, 2022; ISBN 9781119130536.
35. Karuppayah, S.; Jaisan, A. DCNDS Project Dataset—P2P Botnet Detection Using Enhanced Peer Hunter. 2021. Available online: https://doi.org/10.5281/ZENODO.5554851 (accessed on 12 October 2022).
36. CTU University The CTU-13 Dataset. Available online: https://www.stratosphereips.org/datasets-ctu13 (accessed on 12 October 2022).
37. VHS-22 | Kaggle. Available online: https://www.kaggle.com/datasets/h2020simargl/vhs-22-network-traffic-dataset (accessed on 12 October 2022).
38. MTA-KDD-19 | Kaggle. Available online: https://www.kaggle.com/datasets/mathurinache/mtakdd19 (accessed on 12 October 2022).
39. 2019 Trendmicro CTF Wildcard 400 | Kaggle. Available online: https://www.kaggle.com/datasets/hawkcurry/2019-trendmicro-ctf-wildcard-400 (accessed on 12 October 2022).
40. Joshi, A.; Chaudhary, M.S. Study of P2P Botnet. *IOSR J. Comput. Eng.* **2014**, *16*, 35–42. [CrossRef]
41. IMPACT—ISOT Botnet Dataset. Available online: https://www.impactcybertrust.org/dataset_view?idDataset=1281 (accessed on 12 October 2022).
42. Nargesian, F.; Samulowitz, H.; Khurana, U.; Khalil, E.B.; Turaga, D. Learning Feature Engineering for Classification. In Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, Melbourne, Australia, 19–25 August 2017; pp. 2529–2535. [CrossRef]

43. Ferriyan, A.; Thamrin, A.H.; Takeda, K.; Murai, J. Generating Network Intrusion Detection Dataset Based on Real and Encrypted Synthetic Attack Traffic. *Appl. Sci.* **2021**, *11*, 7868. [CrossRef]
44. Hall, M.A. Correlation-Based Feature Subset Selection for Machine Learning. Available online: https://weka.sourceforge.io/doc.dev/weka/attributeSelection/CfsSubsetEval.html (accessed on 13 October 2022).
45. Liu, H.; Setiono, R. A Probabilistic Approach to Feature Selection—A Filter Solution. Available online: https://weka.sourceforge.io/doc.stable/weka/attributeSelection/ConsistencySubsetEval.html (accessed on 13 October 2022).
46. Mohammed, A.J.; Arif, M.H.; Ali, A.A. A Multilayer Perceptron Artificial Neural Network Approach for Improving the Accuracy of Intrusion Detection Systems. *IAES Int. J. Artif. Intell.* **2020**, *9*, 609–615. [CrossRef]
47. Huang, J.; Liu, J. Intrusion Detection System Based on Multi-Layer Perceptron Neural Network and Decision Tree. In Proceedings of the 2012 IEEE Fifth International Conference on Advanced Computational Intelligence (ICACI), Nanjing, China, 18–20 October 2012; pp. 188–190. [CrossRef]
48. Rosay, A.; Riou, K.; Carlier, F.; Leroux, P. Multi-Layer Perceptron for Network Intrusion Detection: From a Study on Two Recent Data Sets to Deployment on Automotive Processor. *Ann. Telecommun. Telecommun.* **2022**, *77*, 371–394. [CrossRef]
49. Florencio, F.D.A.; Moreno, E.D.; Macedo, H.; Salgueiro, R.J.P.D.B.; Do Nascimento, F.B.; Santos, F.A.O. Intrusion Detection via Multilayer Perceptron Using a Low Power Device. In Proceedings of the Euro American Conference on Telematics and Information Systems, Fortaleza, Brazil, 12–15 November 2018. [CrossRef]
50. Catania, C.A.; Garino, C.G. Automatic Network Intrusion Detection: Current Techniques and Open Issues. *Comput. Electr. Eng.* **2012**, *38*, 1062–1072. [CrossRef]
51. Hornik, K. Approximation Capabilities of Multilayer Feedforward Networks. *Neural Netw.* **1991**, *4*, 251–257. [CrossRef]
52. Elejla, O.E.; Anbar, M.; Belaton, B.; Hamouda, S. Labeled Flow-Based Dataset of ICMPv6-Based DDoS Attacks. *Neural Comput. Appl.* **2019**, *31*, 3629–3646. [CrossRef]
53. Hu, T.; Liu, X.; Chen, T.; Zhang, X.; Huang, X.; Niu, W.; Lu, J.; Zhou, K.; Liu, Y. Transaction-Based Classification and Detection Approach for Ethereum Smart Contract. *Inf. Process. Manag.* **2021**, *58*, 102462. [CrossRef]
54. Cantone, D.; Omodeo, E.; Policriti, A. *Set Theory for Computing*; Springer: New York, NY, USA, 2001. [CrossRef]