Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2017) 25: 1963 – 1975

doi:10.3906/elk-1601-105

Intrusion detection in network flows based on an optimized clustering criterion

Jaber KARIMPOUR, Shahriar LOTFI, Aliakbar TAJARI SIAHMARZKOOH*
Department of Computer Science, Faculty of Mathematical Sciences, University of Tabriz, Tabriz, Iran

Received: 10.01.2016 • **Accepted/Published Online:** 17.07.2016 • **Final Version:** 29.05.2017

Abstract: Graph-based intrusion detection approaches consider the network as a graph and detect anomalies based on graph metrics. However, most of these approaches succumb to the cluster-based behavior of the anomalies. To resolve this problem in our study, we use flow and graph-clustering concepts to create a data set first. A new criterion related to the average weight of clusters is then defined and a model is proposed to detect attacks based on the above-mentioned criterion. Finally, the model is evaluated using a DARPA data set. Results show that the proposed approach detects the attacks with high accuracy relative to methods described in previous studies.

Key words: Attack, DARPA data set, flow, graph clustering, intrusion detection

1. Introduction

Nowadays, botnets are continuously attacking networks. For network attack detection, network intrusion detection systems were developed. To find the attacks or malicious behavior, these systems analyze the packet contents of the network [1,2]. The packet inspection is a complicated and resource-consuming process that is usually impossible. Thus, it is important to apply some changes to packet contents that analyze approaches. One way that many research studies have considered this is by flow-based intrusion detection. In this approach, the common properties of network packets are analyzed instead of the packet contents. The advantage of flow-based approaches is that only a fraction of network data is analyzed to detect the attacks [3–5].

A flow is defined as packets that have common properties such as source and destination IP, source and destination ports, and protocol types [6]. In addition, aggregated information about the number of packets and bytes is concluded from the flows. Furthermore, due to the internal data dependency in the network traffic and the relational nature of anomalies, graph-based approaches are crucial in detecting attacks. The nature of anomalies is represented in a relational form, and graphs are used to show these relationships. Graph-based approaches employ data dependency and detect attacks by using parameters in the network traffic graph [7–9].

1.1. Motivation

In this paper, the combination of flow-based and graph-based procedures is employed to reveal attacks. First, it should be noted that network packets are collected to create the flows. The flows mentioned are then clustered using a graph-clustering algorithm that is based on a genetic algorithm [10]. Furthermore, the average weight of clusters is calculated in a time series, and some threshold points and time intervals are investigated to achieve the most accurate detection rate of the attack.

*Correspondence: tajari.a@tabrizu.ac.ir

A first observation related to the proposed method is related to the growing number of attacks occurring every day. Security threats and related incidents in the Internet cause several problems for its users [11]. Therefore, the introduction of new intrusion detection methods is of the utmost concern.

A second observation is related to the continuing growth of line speeds. In this situation, Internet traffic grows as line speeds increase. Thus, network monitoring is a resource-consuming process when a huge volume of network traffic is found in today's networks [12].

Finally, a third observation is related to the data communications in the network, which is based on the nature of the relational behavior of anomalies [13,14]. Thus, according to these discussed views, it is necessary to suggest a process based on the relational behavior of the network, which is only obtained by using the flow and graph concepts.

This paper is organized into the following sections: in Section 2, a description of related concepts and research is introduced. Section 3 presents some preliminaries to create an appropriate overview of the network packets. In Section 4, the suggested approach is discussed, and a criterion is defined to detect attacks based on its values. We also provide the corresponding results in this section. Finally, Section 5 contains a conclusion and a description of future projects.

2. Related work

First, in this section, a general overview of intrusion detection approaches is provided, followed by some related works described in more detail. These approaches are categorized in 4 parts as follows:

1) Feature-based approaches: the key idea of these approaches is based on the concept that similar graphs probably share common attributes such as diameter, eigenvalues, and a distribution of degree. Moreover, these methods can be used for checking the structure of a graph in order to find patterns and explore anomalies [15–18].

2) Decomposition-based approaches: in these approaches, tensor decomposition and graph structure are used as an interpretation of eigenvectors and convergence of graph attributes to find the patterns, respectively [19–23].

3) Community-based approaches: in these approaches, the main action is graph clustering. Clustering algorithms are employed to create cluster parameters of data, and the anomalies are recognized based on their values [24–26].

4) Window-based approaches: in this category, the evolutionary behavior of the anomalies in time intervals reveals the patterns. Many recent research studies have also employed these methods in order to detect the behavior of the network and whether it is a normal or malicious case [12,27].

The above-mentioned approaches can now be explained in more detail. We start with the packet content analyzing approach [28,29], which in high-speed networks cannot be useful when considering scalability. One of the approaches for solving this problem is the packet header-based procedure. In this phase, Mahoney et al. [30] proposed an important method based on the packet header such that the anomalies are detected based on the normal values for each packet header. Similarly, Manandhar et al. [31] suggested a new method using traffic data in which the information of the packet header is checked out for anomaly detection. Here, it should be mentioned that to overcome problems related to the scalability of networks, the researchers changed their

viewpoint. by focusing more on the flow concept for intrusion detection. In this approach, the flow data are analyzed instead of the packet contents. In 2010, a method, using time series, was proposed to detect intrusion based on network flows [32].

The other proposed approach that used the flow concept is the work of Hellemons et al. [33]. The study is made up of 2 parts: in the first, an algorithm is proposed for intrusion detection and, in the second part, an initial instance of the IDS is implemented. The authors proposed an algorithm to detect a dictionary attack, which splits attacks into 2 or more phases. The criteria used in the algorithm are packet per flow and the minimum number of flows collected in 1-min time intervals. The threshold points are considered in each phase of the attack; the researchers considered a dictionary attack as occurring in 3 phases: the scan phase, the brute-force phase, and the die-off phase. By applying different threshold points to each phase, they detected the attack in high accuracy mode.

Graph-based intrusion detection is a type of security method that uses the properties of the network instead of packet contents. For the first time, these systems were presented by Staniford-Chen et al. in 1996 [34]. These systems are able to detect an attack by network graph analysis and high scalability attacks such as worms. Moreover, the implementation of this method is related to Ellis et al.'s 2006 study [35]. Some of the graph-based methods can now be described in the following works:

Zhou et al. [36] introduced a method that uses the graph concept for implementing multivariable time series and their relations in each time phase. In 2007, Iliofotou et al. [37] proposed an approach to monitor and analyze network traffic using a traffic dispersion graph. They defined the traffic dispersion graph as the graphic presentation of interactions among groups of nodes. The advantage of using this application is its power to better present the structural relations of the attacks. Another approach, proposed by Le et al. [38], is based on graph theory fundamentals such as degree of nodes, maximum degree, and similarity distance of the graph.

In addition, one of the important fields in graph-based intrusion detection is the graph-based clustering method. In this field, Muniyandi et al. [39] used the decision tree to optimize the k-means algorithm for intrusion detection. Furthermore, in 2012 and 2014, respectively, Mingqiang et al. [40] and Yin et al. [41] also used graph clustering to detect network intrusions. Mingqiang et al. considered 3 sets for the normal, pending, and anomaly clusters and placed the data in N clusters. The clusters were placed in 3 types of defined clusters based on the rate of normal and anomaly data that had already been specified. The existing clusters in the pending clusters were placed in one of the normal or anomaly groups using the local deviation coefficient. Yin et al. optimized this approach and used the local deviation factor instead of the local deviation coefficient.

Table 1 shows a general view of the previous intrusion detection methods.

Table 1. Anomaly detection methods.

Method	Data type	Attack	Proposed system	Accuracy
Graph in time series	Flow-based	DDoS	Graph-based	94.2%
Dispersion graph	Flow-based	DDoS	Graph-based	100%
Using flow concept	Flow-based	Dictionary	Flow-based	99%
Graph clustering and local deviation coefficient	Packet-based	DoS, Scan	Graph-based	95.3%
Graph clustering and local deviation factor	Packet-based	DoS, Scan	Graph-based	97.2%
Packet heard analyzing	Packet-based	DoS, Scan	Packet-based	95.4%

We should mention that the flow and graph-clustering concepts in this paper are used to detect an attack in the network such that the nodes, the edges, and the weight of edges indicate the IPs, the flows, and the number of flows in the graph, respectively. Additionally, based on the average weight of clusters that are reached from the graph-clustering algorithm and comparing it in several time intervals and threshold points, the anomaly points can be detected. We expect that our approach will detect attacks with high accuracy since it considers the data dependency in the network flows that is such an important parameter in identifying the behavior of attacks.

3. Preliminaries

According to the studies discussed in the previous section, increasing the accuracy of the intrusion detection approach faces restrictions in detecting some types of attacks [34]. Due to the disadvantages of the methods discussed in previous works, an optimized approach is provided here that will enable detection rates to increase and false alarm rates to decrease. The flow and graph-clustering concepts are used in our study to reveal attacks with high accuracy. For more illustration, the corresponding steps are listed in Figure 1.

1. Collecting packets of network traffic
2. Extracting packet header
3. Creating flows based on common properties of the packets
4. Creating graphs and performing clustering algorithm on graphs in several time intervals
5. Calculating number and weight of clusters in normal and anomaly state
6. Detecting attack based on the above parameters
7. Creating Markov model based on average weight of clusters
8. Calculating detection rate of intrusion detection and accuracy of model

Figure 1. Steps of the proposed approach.

As shown in Figure 1, some data preprocessing should be done on the network traffic that is collected as a packet-based data set. A packet's headers are extracted from the packets and then, according to their common properties, the flows are simulated in time intervals. The flows are then clustered using the graph-clustering algorithm. In addition, the final model is created based on the new criterion to detect attacks in the time series. Finally, the detection rate and false alarm rates are calculated by applying some time intervals and threshold points on values of the criterion. All of these steps are explained in more detail in the next subsections.

3.1. Packet header extraction

As mentioned above, the common properties of the packets should be extracted to simulate the flows. Thus, extracting the packet's headers that include information about the packets is required. The information is about source and destination IP addresses, source and destination ports, and protocol type. Some other information is available in the packet headers; however, the headers are not usable for the flow simulation and other processes. As an example of a packet's headers, Figure 2 represents the properties of packets so that each row in this figure indicates the extracted properties of one of the headers. The columns show the packet number, source and destination IP, protocol type, the time the packets were sent, and source and destination port numbers. In the next step, the flows should be simulated according to the categorized packets from which their headers were extracted.

Packet number	Source IP	Destination IP	Protocol	Time	Source Port	Destination Port
1	172.16.100.21	172.16.27.16	TCP	08:00:03	1022	25
2	172.16.100.21	172.16.27.16	TCP	08:00:03	1022	25
3	172.16.100.21	172.16.27.16	TCP	08:00:05	1022	25
4	192.168.84.15	172.168.43.98	UDP	08:00:05	1024	1022
5	192.168.83.2	192.168.64.26	UDP	08:00:07	1098	655
6	172.168.14.43	192.168.64.26	UDP	08:00:07	1014	109
7	172.168.14.43	172.168.109.1	UDP	08:00:09	1037	126
8	172.168.109.1	172.16.100.9	UDP	08:00:10	1024	243
9	172.16.100.9	192.168.3.3	TCP	08:00:10	1035	244
10	192.168.42.12	172.168.1.45	UDP	08:00:11	1012	359

Figure 2. Example of a packet's headers.

3.2. Flow simulation

Based on the definition of a flow in the first section of this study [6], packets that have common properties are collected at the same time intervals and are shown as the flows. A 7-tuple is used to show the simulated flows:

(Src_IP, Dst_IP, Packt, Time, Src_Port, Dst_Port, Prot)

All of the packets that have the same values of mentioned parameters are included in the same flow. As an example of the flow properties, Figure 3 is presented in such a way that each row shows the mapped source and destination IP, number of packets in the flow, the time the flow was sent, source and destination port numbers, and the protocol type. For example, 8 packets pass through the network from IP 2 to IP 3 at 08:03:12 under TCP protocol from port number 1036 to 1012.

3.3. Graph creation using the flows

As already known, a set of flows is provided at each time interval. These flows show several graphs at various time points t_1, t_2,..., t_n such that the flows are the edges of the graphs and the number of flows between 2 nodes represents the weight of the edge. Therefore, the steps of graph creation from the flows are as follows: initially, some of the time intervals are created (t_1, t_2, ..., t_n) and the flows are included at each time interval. Then the graphs are created using the nodes, flows, and their weights. The final graph is provided in different time bins using the adjacency matrix of the contained flows and nodes.

3.4. Graph clustering

After the graph creation process, a genetic-based, graph-clustering approach is introduced to create the best clusters of flows. The steps of this algorithm are as follows:

Source IP	Destination IP	Number of Packets	Time	Source Port	Destination Port	Protocol
2	3	8	08:03:12	1036	1012	TCP
1	4	27	08:03:15	1015	115	TCP
2	3	16	08:03:19	1025	165	UDP
1	2	9	08:03:26	1023	114	UDP
1	4	4	08:03:27	104	109	UDP
3	5	7	08:03:28	104	108	UDP
8	16	1	08:03:29	102	1024	TCP
10	3	12	08:03:30	1022	1012	TCP
9	2	10	08:03:33	1024	67	TCP
1	16	5	08:03:36	1022	87	UDP

Figure 3. Example of flows.

Step 1: an initial population is created such that each chromosome represents a set of random numbers that are related to the cluster's number. It should be emphasized that the length of each chromosome is equal to the number of graph nodes.

Step 2: the fitness function is calculated based on internal and external communications. Interval flows are related to the flows that are in the same cluster, and external ones are the flows between clusters that are not part of any cluster.

In order to achieve the best fitness function, the 2 parts of the communications process should be considered. This is illustrated in Eq. (1), which indicates the number of connections between the nodes in a cluster. It is then clear that the best clustering is attained for the high values of A_i in this equation. In this expression, μ_i and N_i show the number of edges and nodes in the ith cluster, respectively. Therefore, N_i^2 represents the total number of edges in the ith cluster, leading A_i to show the probability of the internal communication in the clusters.

$$A_i = \frac{\mu_i}{N_i^2} \tag{1}$$

In addition, to show the external communications, it is necessary to define a parameter so that the probability of the previously mentioned criterion is calculated. One could evaluate the number of connections between the clusters from Eq. (2). To attain the best clustering, lower values of E_{ij} are needed since these values occur when the dependency among the clusters is low. As a result of this, independent clusters are created, meaning that the best clustering has been implemented. ε_{ij} and $2N_iN_j$ represent the number of edges from the ith to the jth cluster and the maximum number of edges between 2 clusters, respectively.

$$E_{ij} = \begin{cases} 0 & \text{if} \quad i = j \\ \frac{\varepsilon_{ij}}{2N_iN_j} & \text{if} \quad i \neq j \end{cases} \tag{2}$$

Hence, E_{ij} shows the degree of dependency from the ith to the jth cluster. The final objective function is

achieved based on Eq. (3). In this equation, k represents the number of nodes in the graph, and the probability of internal connections cancels out the external ones.

$$MQ = \begin{cases} \frac{\sum_{i=1}^{k} A_i}{k} - \frac{\sum_{i,j=1}^{k} E_{ij}}{\frac{k(k-1)}{2}} & \forall k > 1 \\ A_i & k = 1 \end{cases} \tag{3}$$

Step 3: after calculating the fitness function, the selection process of the genetic algorithm is performed to select the best population.

Step 4: a 2-point crossover and swap mutation are applied to the population in order to add other individuals.

Step 5: the next population generation is created using the replacement action of the genetic algorithm.

After performing all of above-mentioned preliminaries, a cluster-based data set is available such that some criteria are defined, and an anomaly can be detected based on their values in the time series. An example of the proposed data set is provided in Figure 4. Some important properties of the cluster-based data set such as time interval, the number of clusters, the number of cluster nodes, internal and external packets, and internal and external flows are calculated.

Time interval	Cluster's number	Number of nodes	Internal packets	External packets	Internal flows	External packets
1	1	3	14	3	8	**2**
1	2	6	13	2	6	**1**
1	3	2	7	5	3	**2**
1	4	5	8	6	2	**3**
2	1	4	13	4	5	**4**
2	2	7	16	2	9	**1**
3	1	4	8	5	4	**1**
3	2	3	5	6	2	**2**
3	3	3	11	3	4	**1**
3	4	4	17	7	3	**3**
3	5	5	22	12	10	**5**
3	6	3	8	7	2	**2**
4	1	2	9	4	6	**2**
5	1	7	10	8	3	**3**

Figure 4. Part of a cluster-based data set.

4. Proposed approach

In this section, a criterion is proposed to detect attacks based on its values in the time series. This criterion should have different values in the normal and anomaly conditions of the network. Based on this fact, in this paper, the flow and cluster concepts are used to define the best criterion as the behavior of the network can be analyzed by the combination of these concepts. Also, after creating the flows and then performing the graph-clustering algorithm on the flow-based data set, a cluster-based one is created. Therefore, the criterion should consider the internal and external of properties of the clusters and the number of clusters in the emerged data set. The proposed criterion is deduced from Eq. (4), such that AveW, W_i, $ExtW_i$, and N represent the average weight of clusters, total weight and external weight of the ith cluster, and the number of clusters, respectively. Accordingly, we believe that in the attack condition, the value of the proposed criterion changes over time. To

prove this claim, a DARPA data set is used to evaluate the proposed approach. Many recent papers [42–46] used DARPA data sets that included packet-based network traffic. These data consist of 7 weeks of network traffic and 5 types of attacks: DoS, scan, local access, user to root, and data. Table 2 shows the number and types of attacks in each categorized attack. A brief description of these attacks is provided in Table 2.

Table 2. Various attack descriptions.

Attack type	Description
DoS	Denial of service; an attempt to make a network resource unavailable to its intended users: temporarily interrupt services of a host connected to the Internet
Scan	A process that sends client requests to a range of server port addresses on a host to find an active port
Local access	The attacker has an account on the system in question and can use that account to attempt unauthorized tasks
User to root	Attackers access a user account on the system and are able to exploit some vulnerability to gain root access to the system
Data	Attackers involve someone performing an action that they may be able to do on a given computer system, but that they are not allowed to do according to policy

$$AveW = \frac{\sum_{i=1}^{N}(W_i) - (\sum_{i=1}^{N} ExtW_i)/N}{N} \tag{4}$$

According to the proposed criterion and the mentioned data set, the final model of attack detection can be created. More specifically, the cluster-based data are given as input to the model, and then the proposed criterion is calculated in the time series so that the normal and anomaly points are detected based on defined threshold points. Furthermore, different threshold points are investigated in the time series to identify the best one. The best threshold point is extracted from the detection rates of the suggested way during several time intervals. The threshold point and time interval parameters are explained in more detail in the next subsection.

4.1. Time interval and threshold point selection

The 2 parameters discussed above, i.e. the time interval and threshold point, are important criteria for arriving at the maximum detection rate in the proposed approach. The time interval is used to separate the data set into equal time bins. This is done to ensure that each step is performed and, therefore, that the criterion is calculated. Moreover, the threshold points are also constant values that identify the attack and the normal cases. If the value of the criterion is less (more) than the threshold point then the network is in the normal (anomaly) case.

Accordingly, the network traffic is separated into equal intervals of 20, 40, 60, and 90 s, and the steps of our method are implemented in each time bin. The time series of the criterion are then created. To be more specific, the time series of 20 s includes t_1, t_2,..., t_n intervals in a way that t_i represents the ith 20 s of the network traffic. It should be mentioned that all of these intervals are tested in the final model.

For each time series, 4 threshold points are considered to identify their normal and anomaly points. The threshold points are tested at every time interval, and the evaluation rates are correspondingly determined for them. Tables 3–6 show the results for different values of the mentioned parameters. Table 3 represents the results of a 20-s time window. In such a case, the threshold points are 20, 35, 50, and 65 s. In each row of this table, 5 evaluation rates are calculated: true and false positive, true and false negative, and the detection rate. These parameters are defined as follow:

Table 3. True and false rates in a 20-s time window.

Threshold point	False positive	True positive	False positive	True positive	Detection rate
> 20	0.153	0.990	0.010	0.847	0.918
> 35	0.043	0.961	0.039	0.957	0.959
> 50	0.036	0.972	0.028	0.964	0.968
> 65	0.078	0.928	0.072	0.922	0.925

Table 4. True and false rates in a 40-s time window.

Threshold point	False positive	True positive	False positive	True positive	Detection rate
> 30	0.088	0.905	0.095	0.912	0.908
> 45	0.041	0.970	0.030	0.959	0.964
> 60	0.074	0.945	0.055	0.926	0.935
> 80	0.079	0.938	0.062	0.921	0.929

Table 5. True and false rates in a 60-s time window.

Threshold point	False positive	True positive	False positive	True positive	Detection rate
> 40	0.055	0.910	0.090	0.945	0.927
> 60	0.032	0.944	0.056	0.968	0.956
> 80	0.017	0.958	0.042	0.983	0.970
> 100	0.024	0.907	0.093	0.976	0.941

Table 6. True and false rates in a 90-s time window.

Threshold point	False positive	True positive	False positive	True positive	Detection rate
> 50	0.172	0.990	0.010	0.828	0.909
> 70	0.046	0.955	0.045	0.954	0.954
> 90	0.067	0.988	0.012	0.933	0.960
> 120	0.058	0.937	0.063	0.942	0.939

True positive: an attack has occurred, and an alarm is produced.

False positive: no attack has occurred, but an alarm is produced.

True negative: no attack has occurred, and no alarm is produced.

False negative: an attack has occurred, but no alarm is produced for it.

Detection rate: calculated by dividing the number of detected attacks into the number of total attacks.

As shown in Table 3, the threshold point at 50 has a higher detection rate relative to other points. If the detection rate is considered as the main evaluation parameter, then it can be concluded that 50 is the best threshold point for a 20-s interval. However, the threshold point at 20 has the highest true positive rate, and the threshold at 50 has the lowest false positive rate. These are good point parameters for these threshold points; however, as was discussed previously, the essential parameter is the detection rate. As a result, 50 is identified as the best threshold point for the 20-s interval.

The same analysis is deduced from other time windows and threshold points. Table 4 is related to the 40-s time interval at threshold points of 30, 45, 60, and 80. The best threshold point for this time bin is 45 since it has the maximum detection rate value. In Table 5, the time interval is 60 s, and the threshold points

are 40, 60, 80, and 100. Here, the threshold point of 80 is the best one. Table 6 also shows the time interval at 90 s with 50, 70, 90, and 120 considered threshold points. The best threshold point here is 90.

As a final investigation, the results of Tables 3–6 are shown in Table 7. This table selects the best threshold point in each time interval and then includes all of the evaluation values of the mentioned parameters. As shown, the 60-s time interval at the threshold point of 80 has the highest detection rate relative to other time intervals and threshold points. Therefore, the values of the parameters (time interval = 60, threshold point = 80) are considered the best values for testing the network traffic in both normal and anomaly cases.

Table 7. Comparison of the 4 best time intervals.

Time interval	Threshold point	False positive	True positive	False positive	True positive	Detection rate
20	> 50	0.036	0.972	0.028	0.964	0.968
40	> 45	0.041	0.970	0.030	0.959	0.964
60	> 80	0.017	0.958	0.042	0.983	0.970
90	> 90	0.067	0.988	0.012	0.933	0.960

In Table 8, the detection rate of all possible attack types in the data set is represented in the best condition of the mentioned parameters. As shown in this table, the maximum detection is achieved in DoS and user to root attacks.

Table 8. Detection rate of types of attacks in the 60-s time window and at a threshold of 80.

Attack type	DoS	Scan	Local access	User to root	Data	Total
Detection rate	0.882	0.625	0.529	0.928	0.527	0.970

4.2. Comparison

In this section, our approach is compared to procedures that do not use graph-clustering and flow concepts. As mentioned in the final model, the proposed criterion is related to the clustered data set. The evaluation parameters are calculated for each case: before and after clustering. Table 9 shows this comparison using the mentioned parameters. As shown in this table, using the clustering approach leads to higher detection rates relative to the approaches that do not use the clustering algorithm such that the detection rate after clustering is 0.970. However, it is 0.877 in the best condition without clustering.

Table 9. Comparison of the intrusion detection method before and after the clustering method.

Method	False positive	True positive	False negative	True negative	Detection rate
After clustering	0.017	0.958	0.042	0.983	0.970
Before clustering	0.089	0.485	0.515	0.911	0.877

In Table 10, the proposed approach is compared to other intrusion detection systems. The evaluation parameter for comparisons is the number of 100% detected attacks, i.e. the detection rate is 100%. As shown in this table, the number of detected attacks in the data set is 43, while it is 10 and 32 for both the Cisco and Snort systems. This indicates that the performance of the proposed approach in detecting attacks.

To be more specific, Snort is a rule-based intrusion detection system. It combines the advantages of signature- and anomaly-based inspection methods to reveal attacks. Snort can accurately detect threats at high speeds. With nearly 4 million downloads and hundreds of thousands of registered users, it is the most widely

Table 10. Comparison of attacks detection.

Method	DoS	Scan	Local access	User to root	Data	Total
Our approach	15	5	9	13	1	43
Cisco	3	1	5	1	0	10
Snort	9	4	13	5	1	32

deployed IDS in the world. Cisco IDS is also a network-based intrusion detection system that uses a database to control intrusion alarms in which a sensor platform monitors the network and a director platform provides a single GUI management interface for users.

4.3. Conclusion and future work

In this paper, a new approach has been proposed to detect attacks using a graph-clustering algorithm. A new criterion is defined based on the internal and external weight of clusters. We have determined that attacks are detected by the different values of the proposed criterion. In addition, the threshold points and time intervals are considered to evaluate the suggested procedure to achieve the maximum detection rate. Results from the DARPA data set indicate that the proposed model detects the attacks in the network flows with high accuracy. For future consideration, we recommend creating probability models such as Markov models, which are based on the mentioned parameters, to achieve high detection rate accuracy. Also, the definition of new criteria can be an appropriate option to provide high performance intrusion detection approaches.

References

[1] Halme LR, Bauer RK. AINT misbehaving-A taxonomy of anti-intrusion techniques. Comput Secur 1995; 14: 163-172.

[2] Lunt TF. A survey of intrusion detection techniques. Comput Secur 1993; 12: 405-418.

[3] Barford P, Plonka D. Characteristics of network traffic flow anomalies. In: Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement; 1–2 November 2001; Burlingame, CA, USA. pp. 69-73.

[4] Chapple MJ, Wright TE, Winding RM. Flow anomaly detection in firewalled networks. In: Securecomm and Workshops; 28–31 August 2006; Baltimore, MD, USA. pp. 1-6.

[5] Sperotto A, Schaffrath G, Sadre R, Morariu C, Pras A, Stiller B. An overview of IP flow-based intrusion detection. IEEE Commun 2010; 12: 343-356.

[6] Hofstede R, Bartos V, Sperotto A, Pras A. Towards real-time intrusion detection for NetFlow and IPFIX. In: 9th International Conference on Network and Service Management (CNSM) 2013; 14–18 October; Zurich, Switzerland. pp. 227-234.

[7] Akoglu L, McGlohon M, Faloutsos C. OddBall: Spotting anomalies in weighted graphs. In: 14th Pacific-Asia Conference on Knowledge Discovery and Data Mining; 28 February 2010; Hyderabad, India. pp. 410-421.

[8] Akoglu L, Tong H, Koutra D. Graph based anomaly detection and description: a survey. Data Min Knowl Disc 2015; 29: 626-688.

[9] Ellis D, Aiken J, McLeod A, Keppler D, Amman P. Graph-Based Worm Detection on Operational Enterprise Networks. McLean, VA, USA: MITRE Corporation, 2006.

[10] Doval, D, Mancoridis S, Mitchell BS. Automatic clustering of software systems using a genetic algorithm. In: 1999 International Conference on Software Tools and Engineering Practice; 25–28 July 1999; Pittsburgh, PA, USA. pp. 73-81.

[11] Hoque N, Bhattacharyya DK, Kalita JK. FFSc: a novel measure for low-rate and high-rate DDoS attack detection using multivariate data analysis. In: 8th International Conference on Communication Systems and Networks; 5–10 January 2016; Bangalore, India. New York, NY, USA: IEEE. pp. 1-2.

[12] Peel L, Clauset A. Detecting change points in the large-scale structure of evolving networks. In: Twenty-Ninth AAAI Conference on Artificial Intelligence; 25–29 January 2015; Austin, TX, USA. pp. 2914-2920.

[13] Perozzi B, Akoglu L, Sanchez P. L, Muller E. Focused clustering and outlier detection in large attributed graphs. In: 20th ACM Special Interest Group on Knowledge Discovery and Data Mining; 24–27 August 2014; New York, NY, USA. pp. 1346-1355.

[14] Karagiannis T, Papagiannaki K, Faloutsos M. BLINC: Multilevel traffic classification in the dark. In: Proceedings of the 2005 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications; 22–26 August 2005; New York, NY, USA. pp. 229-240.

[15] Henderson K, Gallagher B, Eliassi-Rad T, Tong H, Basu S, Akoglu L, Koutra D, Faloutsos C, Li L. RolX: structural role extraction & mining in large graphs. In: 18th ACM International Conference on Knowledge Discovery and Data Mining; 12–16 August 2012; Beijing, China. pp. 1231-1239.

[16] Ding Q, Katenka N, Barford P, Kolaczyk ED, Crovella M. Intrusion as (anti) social communication: characterization and detection. In: 18th ACM International Conference on Knowledge Discovery and Data Mining; 12–16 August 2012; Beijing, China. pp. 886-894.

[17] Henderson K, Eliassi-Rad T, Faloutsos C, Akoglu L, Li L, Maruhashi K, Prakash BA, Tong, H. Metric forensics: A multi-level approach for mining volatile graphs. In: 16th ACM International Conference on Knowledge Discovery and Data Mining; 24–28 July 2010; Washington, DC, USA. pp. 163-172.

[18] Bonacich P, Lloyd P. Eigenvector-like measures of centrality for asymmetric relations. Soc Networks 2001; 23: 191-201.

[19] Ide T, Kashima H. Eigenspace-based anomaly detection in computer systems. In: 10th ACM International Conference on Knowledge Discovery and Data Mining; 22–25 August 2004; Seattle, WA, USA. pp. 440-449.

[20] Liu C, Yan X, Yu H, Han J, Yu PS. Mining behavior graphs for backtrace of noncrashing bugs. In: 5th SIAM International Conference on Data Mining; 10–11 April 2005; Newport Beach, CA, USA. pp. 286-297.

[21] Gunnemann S, Farber I, Boden B, Seidl T. Subspace clustering meets dense subgraph mining: a synthesis of two paradigms. In: 10th IEEE International Conference on Data Mining; 13–17 December 2010; Sydney, Australia. pp. 845–850.

[22] Xu X, Yuruk N, Feng Z, Schweiger T. Scan: a structural clustering algorithm for networks. In: 13th ACM International Conference on Knowledge Discovery and Data Mining; 12–15 August 2007; San Jose, CA, USA. pp. 824-833.

[23] Chakrabarti D. Autopart: parameter-free graph partitioning and outlier detection. In: 8th European Conference on Principles and Practice of Knowledge Discovery in Databases; 20–24 September 2004; Pisa, Italy. pp. 112-124.

[24] Tantipathananandh C, Berger-Wolf T. Constant-factor approximation algorithms for identifying dynamic communities. In: 15th ACM International Conference on Knowledge Discovery and Data Mining; 28–30 June 2009; Paris, France. pp. 827-836.

[25] Kuramochi M, Karypis G. Frequent subgraph discovery. In: 2001 IEEE International Conference on Data Mining; 29–30 November 2001; San Jose, CA, USA. New York, NY, USA: IEEE. pp. 313-320.

[26] Chakrabarti S. Dynamic personalized page rank in entity-relation graphs. In: 16th International Conference on World Wide Web; 08–12 May 2007; Alberta, Canada. pp. 571-580.

[27] Mongiovi M, Bogdanov P, Ranca R, Singh AK, Papalexakis EE, Faloutsos C. Netspot: Spotting significant anomalous regions on dynamic networks. In: 13th SIAM International Conference on Data Mining; 2–4 April 2013; Austin, TX, USA. pp. 1-9.

[28] Martin R. Snort: Lightweight intrusion detection for networks. In: Proceedings of LISA'99': 13th Systems Administration Conference; 7–12 November 1999; Seattle, WA, USA. pp. 229-238.

[29] Wang K, Stolfo SJ. Anomalous payload-based network intrusion detection. In: Jonsson E, Valdes A, Almgren M, editors. Recent Advances in Intrusion Detection. Berlin, Germany: Springer, 2004. pp. 203-222.

[30] Mahoney MV, Chan PK. PHAD: Packet Header Anomaly Detection for Identifying Hostile Network Traffic. Technical Report CS 2001-04. Melbourne, FL, USA: Florida Institute of Technology, 2001.

[31] Manandhar P, Aung Z. Towards practical anomaly-based intrusion detection by outlier mining on TCP packets. In: Proceedings of 25th International Conference on Database and Expert Systems Applications; 1–4 September 2014; Munich, Germany. pp. 164-173.

[32] Sperotto A, Schaffrath G, Sadre R, Morariu C, Pras A, Stiller B. An overview of IP flow-based intrusion detection. IEEE Commun 2010; 12: 343-356.

[33] Hellemons L, Hendriks L, Hofstede R, Sperotto A, Sadre R, Pras, A. SSHCure: a flow-based SSH intrusion detection system. In: Proceedings of the 6th International Conference on Autonomous Infrastructure, Management and Security; 4–8 June 2012; Luxembourg, Luxembourg. pp. 86-97.

[34] Staniford-Chen S, Cheung S, Crawford R, Dilger M, Frank J, Hoagland J, Levitt K, Wee C, Yip R, Zerkle D. GrIDS: A graph based intrusion detection system for large networks. In: Proceedings of the 19th National Information Systems Security Conference; 22–25 October 1996; Baltimore, MD, USA. pp. 361-370.

[35] Ellis DR, Aiken JG, McLeod AM, Keppler DR, Amman PG. Graph-Based Worm Detection on Operational Enterprise Networks. Technical Report. McLean, VA, USA: George Mason University, 2006.

[36] Zhou Y, Hu G, He W. Using graph to detect network traffic anomaly. In: Communications, Circuits and Systems; 23–25 July 2009; Milpitas, CA, USA. pp. 341-345.

[37] Iliofotou M, Pappu P, Faloutsos M, Mitzenmacher M, Singh S, Varghese G. Network traffic analysis using traffic dispersion graphs (TDGs): techniques and hardware implementation. In: Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement; 24–26 October 2007; San Diego, CA, USA. pp. 315-320.

[38] Le DQ, Jeong T, Hong JWK. Traffic dispersion graph based anomaly detection. In: Proceedings of the Second Symposium on Information and Communication Technology; 13–14 October 2011; Hanoi, Vietnam. pp. 36-41.

[39] Muniyandi AP, Rajeswari R, Rajaram R. Network anomaly detection by cascading k-means clustering and c4.5 decision tree algorithm. Proc Eng 2012; 30:174-182.

[40] Mingqiang Z, Hui H, Qian W. A graph-based clustering algorithm for anomaly intrusion detection. In: 7th International Conference on Computer Science & Education; 14–17 July 2012; Melbourne, Australia. pp. 1311-1314.

[41] Yin S, Chen Z, Kim S. LDFGB algorithm for anomaly intrusion detection. In: Linawati, Mahendra MS, Neuhold EJ, Tjoa AM, You I, editors. Information and Communication Technology. Berlin, Germany: Springer, 2012. pp. 396-404.

[42] Moorthy M, Rajeswari M. Virtual host based intrusion detection system for cloud. Int J Eng Tech 2013; 5: 5023-5029.

[43] Huang T, Zhu Y, Zhang Q, Zhu Y, Wang D, Qiu M, Liu L. An LOF-based adaptive anomaly detection scheme for cloud computing. In: 37th Annual Computer Software and Applications Conference Workshops; 22–26 July 2013; Kyoto, Japan. pp. 206-211.

[44] Madhu BR, Pratik PJ. Data mining based CIDS: Cloud Intrusion Detection System for Masquerade Attacks [DCIDSM]. In: Computing, Fourth International Conference on Communications and Networking Technologies; 4–6 July 2013; Tiruchengode, India. pp. 1-5.

[45] Vaid C, Verma HK. Anomaly-based IDS implementation in cloud environment using BOAT algorithm. In: 3rd International Conference on Reliability, Infocom Technologies and Optimization (ICRITO) (Trends and Future Directions); 2–4 September 2015; Noida, India. pp. 1-6.

[46] Chou H, Wang SD. An adaptive network intrusion detection approach for the cloud environment. In: International Carnahan Conference on Security Technology; 21–24 September 2015; Taoyuan, Taiwan. pp. 1-6.