A Cluster-based Approach Towards Detecting and Modeling Network Dictionary Attacks

Aliakbar Tajari Siahmarzkooh
Department of Computer Science
University of Tabriz
Tabriz, Iran
Tajari.a@tabrizu.ac.ir

Jaber Karimpour
Department of Computer Science
University of Tabriz
Tabriz, Iran
karimpour@tabrizu.ac.ir

Shahriar Lotfi
Department of Computer Science
University of Tabriz
Tabriz, Iran
Shahriar_lotfi@tabrizu.ac.ir

Abstract—**In this paper, we provide an approach to detect network dictionary attacks using a data set collected as flows based on which a clustered graph is resulted. These flows provide an aggregated view of the network traffic in which the exchanged packets in the network are considered so that more internally connected nodes would be clustered. We show that dictionary attacks could be detected through some parameters namely the number and the weight of clusters in time series and their evolution over the time. Additionally, the Markov model based on the average weight of clusters,will be also created. Finally, by means of our suggested model, we demonstrate that artificial clusters of the flows are created for normal and malicious traffic. The results of the proposed approach on CAIDA 2007 data set suggest a high accuracy for the model and, therefore, it provides a proper method for detecting the dictionary attack.**

Keywords-intrusion detection; Markov chain; grpah clustering; dictionary attack

I. INTRODUCTION

Detecting the intrusion in the flow-based networks as an interesting research is almost a new topic in secure communications. Recently, due to the nature of light weight and scaling, researches have focused on the flow-based approaches. In [1] authors have done a detailed study in the area of intrusion detection for such networks. In the mentioned research, the flow is defined as a set of packets which has similar properties like the same source and destination. Hence, the measurement of the flows indicates an aggregated view of traffic information which reduces the amount of information being analyzed [2]. Furthermore, considering the internal dependency in the network data and relational nature of anomalies, graph-based methods are crucial to detect attacks. In fact, the nature of anomalies is represented in a relational form, and graphs are used to reveal these relations. Graph-based IDSs employ the data dependency and detect attacks by means of some parameters of network traffic.

In [5], authors have designed and developed a system for the flow-based intrusion detection which proved to be suitable for higher efficient networks and resistant against DoS attacks. In [6] authors developed a framework to monitor the network and identify the computer worms and in [7] authors considered the effect of methods of flow-based smart sampling in recognizing the anomalies. Statistical methods to detect intrusion in the network flows were applied in [8] whereas a more general mechanism and defined statistical characteristics of anomaly in the network flows was used in [9].

Many IDSs have employed the communicative graph-based models. In general, the types of anomaly detection methods include the following categories [10]: Anomaly detection in static graphs, Anomalies in simple graphs (without labeling), Anomalies in attributed graphs (with label on nodes or edges), Anomaly detection in dynamic graphs, Feature-based events, Decomposition-based events, Community or clustering-based events and Window-based events.

In the “anomaly detection in static graphs” methods the main task is to trap malicious entities in the network (like nodes, edges and sub-graphs). In the “anomalies in simple graphs (without labeling)” method, the only reachable information in a simple graph is its structure. Therefore, these anomaly detection methods might be used for checking the structure of graph in order to find the patterns and explore the anomalies [11-16]. In the “anomalies in attributed graphs (with label on nodes or edges)” methods there is a rich presentation of the graph for some types of data which might display the nodes and edges as attributes. An example for such graphs includes social networks with the user’s interests. These anomaly detection in the attributed graphs use the mentioned structure as well as convergence of graph attributes to find the patterns [17-21]. The “anomaly detection in dynamic graphs” methods are another type of algorithm to detect anomalies focusing on the sequence of static graphs. More specific, the evolution of graphs has been studied by several research groups recognized as the detection of temporal anomaly pattern, event and point of change which is usually defined as considering a sequence (simple or attributed) of graphs in which the target is to find the timestamp. It is related to a change or event and, consequently, represents the k nodes, k edges or graph parts that contributes in more changes. In the “feature-based events” methods the key idea is that similar graphs probably share certain attributes such as distribution of degree, diameter and eigenvalues [22]. The "decomposition-based events" methods detect such temporary anomalies through tensor decomposition or decomposition of graph tensors as well as interpretation of eigenvectors and singular values chosen appropriately [23]. These methods can fall into two categories based on graph display: matrix or tensors. In the "community or clustering-based events" methods, clustering plays a vital role. The major idea is monitoring the communities or graph clusters over time [24-27]. The "window-based events" methods recognize the anomalies of evolution graphs employing methods that look for behaviors and anomaly patterns in the input graph in time intervals. Particularly, a number of previous works have been used to model normal behavior. In these researches the input graph is compared with the created model in order to detect the behavior as either normal or malicious [28, 29].

In this paper, a combination of flow-based and graph-based approaches is used to detect the attack and to model the behavior of the network using graph clustering concepts and time series related to the associated parameters. The overview of the proposed approach is the following: 1-collect the packets of network traffic, 2-extract packet headers, 3-create flows based on the common properties of the packets, 4-calculate the number and weight of clusters in the normal and abnormal state, 6-detect the attack considering the previous parameters, 7-create the Markov model based on the average weight of clusters, 8-calculate the detection rate of intrusion detection and the accuracy of the model. It should be stressed that in this paper, the graph clustering algorithm in the network flows is used in order to achieve some appropriate parameters to detect the attacks whereas the cluster-like behavior of the produced flows of attacks has not been considered in the above mentioned works.

II. PROPOSED SOLUTION

In this section, we propose a new strategy for attack detection. Particularly, we create a Markov model for normal state as well as dictionary attack mode in the network traffic. The proposed solution needs to assume the network and transmitted packets as a directed weighted graph. Consequently, in this graph, nodes, edges and weights represent IPs, flows transmitted between the nodes and number of flows transmitted in each connection, respectively. Time series are shown to be appropriate tools for intrusion detection and then a discussion is made upon how the attacks could be detected based on the number of clusters and their weights. It should be pointed that the results introduced in this work are achieved by analyzing the behavior of packets of network graph.

A. Analysis strategy

The presented analysis in this section expresses two key concepts of the proposed solution for intrusion detection: a) We believe that the attacks or, in more general, the anomalies can be detected with regard to the evolution of clusters obtained from implementation of genetic-based graph clustering algorithm and also consideration of their weights (the number of flows among nodes) over time. In addition, time series of network traffic are analyzed in a sequential mode to determine what has happened. Here, graph clustering parameters such as the number of clusters and their weights could be utilized to create time series. It should be noted that in [30], a clustering method based on genetic algorithms was presented in which the software system is assumed as a directed graph and, then, the available graph turn out to be clustered by the genetic algorithm [30, 31]. b) We suggest an approach using graph clustering to detect the attack so that each edge of the graph represents a flow between two IP addresses in the network which carries some packets. Here, we believe that separate analysis of parameters which are obtained from the clustering on multiple separate pieces of traffic can assist in intrusion detection, since each separated piece in individual does not reveal the real behavior of traffic, however, multiple separated pieces will prove to do.

Subsequently, the mentioned genetic-based algorithm consists of the following steps:

1. At first, a set of chromosomes are created in which the encoding is random number generation so that the length of a chromosome depends on the number of graph nodes encoded with the random numbers starting from 1 to the number of nodes (n).

2. The objective function which consists of two parts is calculated as follows: the internal communication of clusters indicates the number of connections between the nodes within a cluster as given by $A_i = \mu_i / N_i^2$ where μ_i and N_i represent the number of edges and nodes in i-th cluster, respectively and that N_i^2 indicates the total number of edges in i-th cluster. The communications of clusters represents the number of connections among the clusters for which the low values lead to the low dependency among the clusters and, thus, an appropriate clustering is achieved and is given by:

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

where, E_{ij} shows the dependency degree between i-th and j-th cluster. The number of edges from i-th to j-th cluster is indicated by and $2N_iN_j$ is the maximum number of edges between two clusters. The objective function is obtained from:

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

where k is the number of clusters. The above equation is used for evaluation of unweight graphs in a way that if the graph is a weighted one, we change the parameters of objective function using some new definitions such that here μi would be the sum of weight of edges in i-th cluster and shows the total weight of edges between i-th and j-th cluster in normalized state.

3. Using selection function, selection process is performed.

4. Crossover and mutation operations are performed.

5. Utilizing replacement action, new population and thus next generation are created.

Furthermore, the introduced algorithm must run until a better clustering wouldn't be obtained. Generally speaking, if the number of iterations doesn't exceed a maximum, the algorithm goes to step 2; else, the process must be stopped and the best founded answer should be displayed. The algorithm iterates a fixed number of times. Then, because the upper bound of function (the maximum fitness value is possible) is often not found, we limit the number of generations in order to guarantee the termination of search process.

B. Data set

We use the CAIDA 2007 data set which is comprised of packet-based traffic [4]. Now, flow-based data is needed to evaluate our approach and, therefore, we have to do the following steps to convert the packet-based data into flows:

1. Each packet is read from the traffic.

2. Common properties (source IP address, destination IP, protocol, sending time, source and destination port) are extracted from the packets.

3. According to the flow definition, extracted headers are classified into one second intervals.

4. Packets with common properties are placed in the same flow, the number of packets is also calculated.

Finally, there are 1-second flows which can be utilized on different time series.

C. Network traffic

In this section, we test the normal and malicious traffic that can be observed on graph level. We analyze the time series of clusters, their weights, and then highlight the changes which exist in the malicious network.

C.1. Graph

The time series related to the number of clusters and their weights are shown in Figures 1 and 2. In some parts of the diagram in Figure 1, the peak of nearly 190 is observed that indicates a lot of exchanged packets in the network. By considering the time series of number of clusters in the graph (Figure 1), we can see the peaks in the diagram that are observed in some periods of time. In other words, such peak points in the graph can represent some anomalies but not all of them. Therefore, we have to derive a new parameter to detect and model the anomalies based on these two parameters. Significantly, knowing that simultaneous observing of the number of clusters and their weights allow us to distinguish between the normal traffic and the malicious one. For example, a peak point in the time series of the weights of clusters can be created by both of the file transfer and a scan however, if the peak occurs in the time series of the number of clusters, we can say that an IP scan creates a peak point. Since during a scan, the edges come into the more number of nodes, more separate clusters are generated and therefore, the number of clusters increases.

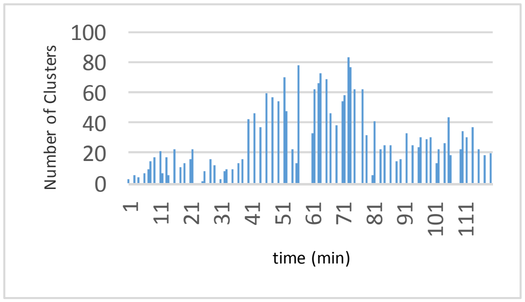

Fig. 1. Time series of number of clusters in 120 minutes.

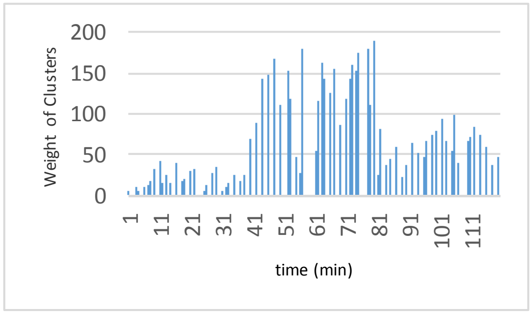

Fig. 2. Time series of weight of clusters in 120 minutes.

C.2. Normal traffic versus malicious

The aim of the analysis in this section is to prove that the mentioned peaks really relates to an attack. In order to identify the network behavior in the attack time, it is needed to compare it with another time frame reflecting in a network without any attack. Table I represents the number of active hosts in the normal and malicious time frames which represents that the number of sources and destinations are approximately equal values in a normal time frame. On the contrary, in the malicious time frame, a large number of destinations against the number of sources are observed leading to many clusters created based on the clustering algorithm. The number of destinations in malicious condition indicates that almost all of IP addresses in the network are scanned, while increasing the number of sources is related to scanning task because of some destinations responding to it. Table II indicates that the low percentage of number of IPs is assigned to clusters from 1 to 3 and other percentages are assigned to other clusters. However, 77% of total weight of clusters is assigned to these three clusters. Hence, the unbalanced allocation of weights to the clusters represents the malicious behavior of the network traffic.

D. Overview on intrusion detection method

In previous sections, we observed that the cluster-based time series such as the number of clusters and their weights could be appropriate solution to detect intrusion. We saw that the time series allow one to analyze data and keep the relation between events. The graphs are then appropriate to create time series based on the number of clusters and their weights in time intervals. In addition, the anomaly analysis provides us to consider two parameters: "number of clusters" and "weight of clusters" to detect the intrusion in general form. Furthermore, our analysis indicates that the time series related to the number of clusters and their weights in the graph produce such information on the amount of data transferred in the network.

TABLE I. NUMBER OF SOURCES AND DESTINATIONS WITHIN NORMAL AND MALICIOUS TIME FRAME

	Malicious		Normal	
	Source	*Destination*	*Source*	*Destination*
Number	5	83	46	54

TABLE II. PERCENTAGE OF RESOURCES ASSOCIATED WITH CLUSTERS

	Number of IPs	Weight of clusters
Cluster 1	4%	33%
Cluster 2	6%	22%
Cluster 3	6%	22%
Total	16%	77%
Other	84%	23%

E. Markov model creation

In the previous sections, we indicated that the time series obtained from some parameters of graph clustering algorithm can be a powerful tool to detect attacks. Our suggestion is to create a model for normal and anomaly conditions of the network based on the parameters mentioned in the graph clustering. As an example we model dictionary attack using Markov chain. A Markov chain is a sequence of random variables such that the probability of going to the next state depends only on the current state:

$$Pr(X_{n+1}=x|X_1=x_1, X_2=x_2, \dots, X_n=x_n)=Pr(X_{n+1}=x|X_n=x_n)$$

Markov chains are explained by a sequence of nodes, such that the edges between nodes are labeled by the probabilities of going from one state at time n to another state at time n+1. Also, transition matrix of time n through time n+1 is used to represent the same information.

The proposed approach is based on the following factors: We use the models to describe the specific properties such as the number of clusters and their evolution over the time. To do this, we need a framework for modeling so that Markov chains turn out to be proper tools for sequential data modeling. Now, in order to obtain correct information about the network traffic, it is necessary to know whether the attack is progressing in time series or the network is secure. Therefore, we model the attack traffic and normal traffic, separately. We note that it is supposed that the time series for a complete network are combination of normal and anomalous activities of the network.

F. Some graph properties of network traffic

The introduced method models the time series of dictionary attack traffic and normal traffic. In order to proceed, the first step is to understand what the traffic is like in its extensive form. At first, in what follows, we will explain how a dictionary attack performs if there only exists information related to the number and weights of clusters. Afterwards, an example of network time series will be presented that includes normal and attack traffic.

F.1. Dictionary attack

Dictionary attack is one of the most common attacks in cyberspace. In the mentioned data set a dictionary attack is observed. Here, as an example of such an attack we investigate the traffic generated by a host that has performed dictionary attack against the network. Figures 3 and 4 depict the clusters and their weight during five minutes, respectively. The available values represent the normal and anomalous network traffic. During the attack, the number and weights of clusters are varying. In time series related to the number of clusters, it can be seen that the number of clusters is increasing in the range of 90 to 250 seconds that reaches up to 50 clusters. Then, the number of clusters per second drops suddenly and reaches to 15 clusters in normal state. Finally, the attack activity stops, gradually. Figure 5 shows the time evolution of average weight of clusters at resolution of 1 second. Based on this criterion, three phases of dictionary attack are completely identifiable. The first one is the scanning phase which can be identified by low average weight of clusters (between approximately 0 and 10). These values are based on the fact that three-phase handling has started but only a few of them are completed. The second phase i.e., brute-force phase when begins, average weight of clusters remarkably grows (up to 83). During this phase, various user/passwords are tested against a victim. This explains why the average weight of clusters increases. Finally, the third phase which is die-off phase indicates low average weight of clusters. In most cases, the average weight of clusters will be 10. Such traffic might be due to the fact that the attacker has not been able to finish the sessions. Moreover, variations in the average weight of clusters over the time seem to be a key feature of dictionary attack. At last, in Fig. 6, it is observed that a deeper analysis indicates that the time series of activity pattern is not constant over the time, i.e. every second of activity brings about one or some seconds of inactivity. The importance of this fact will be clear during the attack modeling.

F.2. Network time series

As we see in Figures 1 and 2, the traffic includes transmission of normal and malicious data which illustrate the time series, the number of clusters, and their weights at resolution of 1 second which include a dictionary attack. Observing of the peak points in the weight of clusters is typically created by the file transmission which does not create a remarkable variation in the time series of number of clusters. On the other hand, in the time series of normal state traffic, it cannot be concluded that there is a key criterion to describe the traffic evolution whereas the time series of average weight of clusters is suitable for the traffic of dictionary attack (Figure 4).

G. Traffic modeling

In this section, traffic models are described based on the previous section. As it was already mentioned, a dictionary attack includes three phases: scanning phase, brute-force phase and die-off phase. In this regard, we use active and inactive characteristics of the time series of the number of clusters and their weights to describe a discrete time Markov chain. Our model of dictionary attack has seven states. In three states of S_1, S_2, and S_3, the attacker is active and the traffic is generated and in three states of I_1, I_2, and I_3 the attacker is temporarily inactive and there is a final state called "End". States S_1 and I_1 model the scanning phase of the attack, S_2 and I_2 model the brute-force phase, S_3 and I_3 model the die-off phase and the End state finishes the traffic [1]. Since, in every transition from a state to another one, the number and weight of clusters are generated according to the empirical distribution related to the current state. This is equivalent to this fact that in states S_1, S_2 and S_3 of the proposed model, the following empirical states are assigned: a) an empirical distribution probability related to the number of clusters in every time interval (PN) and b) an empirical distribution probability related to the average weight of clusters (PAve).

To proceed, the modeling of normal traffic is easier than the modeling of attack traffic. In fact, a two-state Markov chain is used to describe the normal traffic. Similar to the dictionary attack traffic, the time series of normal traffic include active and inactive time intervals that are indicated with S (active) and I (inactive). The empirical distributions of this state will be similar to the attack state such that in each transition, random values of N (number of clusters) and Ave (average weight of clusters) will be generated. For more illustration, the dictionary attack models and normal traffic are presented in Figure 6.

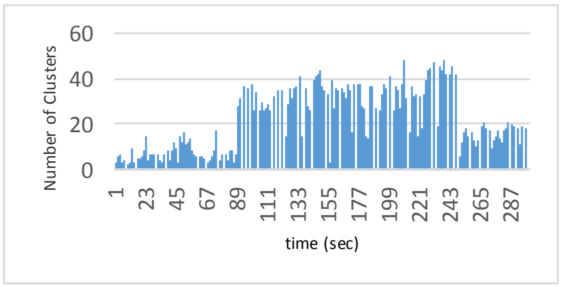

Fig. 3. Number of clusters during 5 minutes.

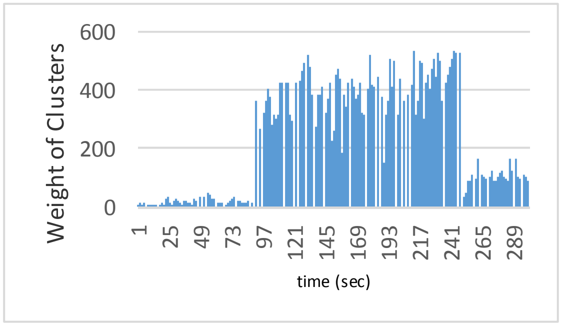

Fig. 4. Weight of clusters during 5 minutes

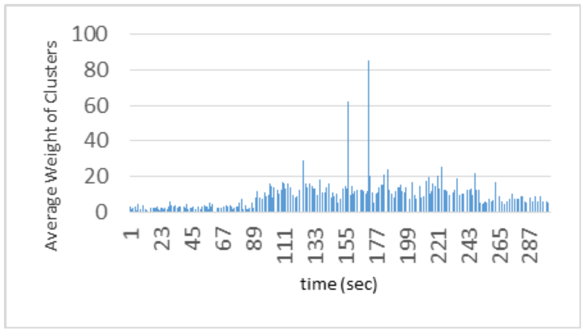

Fig. 5. Average weight of clusters during 5 minutes

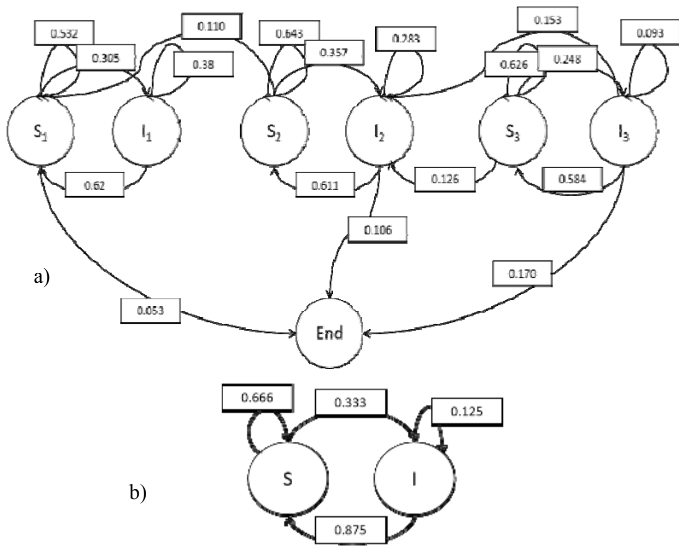

Fig. 6. Markov models: a) Markov chain for dictionary attack, b) Markov chain for normal state

As shown in Figure 6a, the model of dictionary attack has two states for each phase which consists of active and inactive states that are represented by S_i and I_i, respectively. Additionally, all of transitions are labeled by the probabilities of moving from a state to another one. The normal model of traffic is represented in Figure 6b that only two states are assumed for modeling the traffic in the normal case; S and I. In this figure the probabilities of labeled transitions represent the probabilities of moving between the states. In more details, there are several traces from the initial state of model to the end state with different probabilities. For example, in the dictionary attack model, one trace can be $S_1I_1S_1S_2I_2I_3S_3S_3S_3I_3$End. Also, SSIIISISII can be a trace in the normal model. As we introduced before, in each trace, the transition probability from one state to another, distribution probability of number of clusters and weight of clusters are calculated.

H. *Model validation*

To validate the proposed model, artificial time series are generated with statistical properties from the original data sets and they are compared to the original ones [3]. In this section, the testing methodology is described and related results are presented. We evaluate our model on the network traffic data set collected by CAIDA 2007 that has been previously investigated. Table III presents a review of the attacks in the data set in the form of number of clusters and weight of clusters per second. Likewise, Table IV indicates the main properties of traffic in the data set. As shown in Tables III and IV, the number of clusters per second in the attack case is more than one in the normal case of traffic. Also, the weight of clusters are more than one in the normal state. Consequently, we can say that these parameters are useful to differentiate between the attack and normal traffic, however, it is the combination of these parameters that can be essential to detect the attack (so, we suggested the average weight of clusters to detect the dictionary attack).

TABLE III. AVERAGE NUMBER AND WEIGHT OF CLUSTERS IN DICTIONARY ATTACK

	Number of clusters	**Weight of clusters**
Per second	7.2	43.12

TABLE IV. AVERAGE NUMBER AND WEIGHT OF CLUSTERS IN NORMAL DATA

	Number of clusters	**Weight of clusters**
Per second	3.3	22.16

I. Traffic modeling

We describe the time series as a sequence of observations that are created as output by the models when a random path is selected. The paths begin from S1 in the attack state and S in the normal state. The generation process is as following (supposing the model is in S_i):

a. At time t, the model jumps from current state S_i to next state S_j with the probabilities extracted from the model (j=1,…, n).

b. If S_j is in End state, the trace will be finished.

c. When S_j is selected, the models randomly generate values of N (number of clusters) and W (total weight of clusters) that are obtained from the inverted empirical cumulative distribution function of output distributions of the state.

d. The model delivers the doublet of (N, Ave) as an output that is based on the random values generated in the previous step.

e. When the observations finish, the process repeats from the first step.

In each iteration, the model generates a doublet (N, Ave) which is independent of the previous outputs and is only controlled by empirical distribution probability of N and W related to the current state.

J. Testing methodology and experiment results

Our testing methodology measures a set of artificial traces and compares them with the original data set. Therefore, we consider the following conditions as statistical parameters:

a. The average number and standard deviation of number of clusters (μ_N, δ_N) and the average of the average weight of clusters and standard deviation of the average weight of clusters (μ_{Ave}, δ_{Ave}) over the time. Hence, these measurements separately describe the entire behavior of number of clusters and average weight of clusters in a trace.

b. Correlation coefficient: The correlation coefficient between the number of clusters and average weight of clusters ($\rho_{N,Ave}$) is calculated which could describe the correlation between them in an identical trace. As we know, if the value of correlation is high the selected parameters are then proportional.

Moreover, for every measurement, the related error 'm' is calculated in percentage through the following equation:

$$\text{Error} = (|m_{\text{original}} - m_{\text{artificial}}| / m_{\text{original}}) * 100$$

. In fact, this error is derived from the difference between the original and artificial traces. Here, we describes the results of our experiment summarized in Tables V and VI. The columns of Table V indicate the statistical values of original data set. Likewise, the columns of Table VI present the statistical values calculated for the artificial time series. As we see in Tables Va and VIa, the difference between the statistical parameters is very low and it represents the accuracy of attack traffic model. For more details, in Tables Va and VIa, we consider three phases for dictionary attack and N_i (i=1, 2, 3) represents the number of clusters in each phase. Also, Ave_i (i=1, 2, 3) represents the average weight of clusters in each phase of dictionary attack. As we know, the average weight of clusters in the second phase of attack is more than other phases because of more transmission of packets for testing passwords to attack to the victim which is clear in Tables Va and VIa. Thus, the difference between numerical results (min, max, average and standard deviation) of the original and artificial data is very low, so that we could conclude the accuracy of model is statistically high. Tables Vb and VIb represent low difference between the parameters and high accuracy of the normal traffic modeling. For Tables Vb and Vb, we assumed only two parameters (N and Ave) and only one phase for the normal traffic state. Hence, the parameters of all of the traffic are measured together. As it is clear, the values of mentioned parameters in the tables are approximately equal which confirm the statistical accuracy of model.

TABLE V. DETAILS OF OBSERVED DISTRIBUTION OF NUMBER AND AVERAGE WEIGHT OF CLUSTERS IN ORIGINAL DATA SET

a. DETAILS OF EACH PHASE OF DICTIONARY ATTACK

	Min	**Max**	**Average**	**Standard deviation**
N_1	0	11	3.44	3.15
N_2	0	6	1.93	0.97
N_3	0	7	2.14	0.86
Ave_1	0	11.46	3.13	1.78
Ave_2	0	95.23	16.48	4.44
Ave_3	0	36.23	8.17	5.76

b. BETAILS OF NORMAL STATE

	Min	**Max**	**Average**	**Standard deviation**
N	0	11	3.85	0.86
Ave	0	316.14	14.36	2.56

TABLE VI. DETAILS OF OBSERVED DISTRIBUTION OF NUMBER AND AVERAGE WEIGHT OF CLUSTERS IN ARTIFICIAL DATA SET

a. DETAILS OF EACH PHASE OF DICTIONARY ATTACK

	Min	Max	Average	Standard deviation
N_1	0	9	4.23	4.05
N_2	0	6	2.06	1.16
N_3	0	7	1.98	1.28
Ave_1	0	12.82	3.18	1.69
Ave_2	0	91.19	15.35	4.38
Ave_3	0	38.14	7.68	5.18

b. DETAILS OF NORMAL STATE

	Min	Max	Average	Standard deviation
N	0	10	4.04	0.98
Ave	0	329.54	14.22	2.11

In Tables VII and VIII, the column "Error" represents the relative error between the original and artificial values as percentage in the normal and attack cases. It is seen that the results represent high accuracy of proposed approach. As one can notice, the average, standard deviation and correlation coefficient are correctly estimated for the original data set of attack and normal states. In the average and standard deviation measurements, the relative errors are lower than nearly 5% for the artificial attack traffic and artificial normal traffic. Finally, in the correlation coefficient measurement, the error is nearly 6% in the attack state and nearly 5% in the normal state. Thus, all of these results show the accuracy of the proposed model. Table IX shows that the artificial data is created in high level of accuracy such that the anomalies could be detected with a high accuracy. These values represent that high accuracy in artificial data is created by Markov model and could prove that our model is somewhat exact.

TABLE VII. NUMERICAL RESULTS FOR DICTIONARY ATTACK MODEL

	Original data set	Artificial data set	Error (%)
μ_N	4.18	4.24	1.43
μ_{ave}	42.54	43.19	1.52
Δ_N	2.10	1.96	6.66
Δ_{Ave}	15.34	14.86	3.12
$\rho_{N,Ave}$	0.78	0.82	5.12

TABLE VIII. NUMERICAL RESULTS FOR NORMAL MODEL

	Original data set	Artificial data set	Error (%)
μ_N	3.12	3.25	4.16
μ_{ave}	23.87	23.06	3.39
Δ_N	1.46	1.28	12.32
Δ_{Ave}	38.55	36.84	4.43
$\rho_{N,Ave}$	0.66	0.69	4.54

TABLE IX. NUMERICAL RESULTS FOR DETECTION RATE IN ORIGINAL AND ARTIFICIAL DATA SET

	TP	TN	FP	FN	DR (%)
Original data	0.91	0.89	0.11	0.09	90
Artificial data	0.95	0.93	0.07	0.05	94

III. CONCLUSION AND FUTURE WORK

In this paper, we proposed a new approach to detect attacks using genetic-based graph clustering algorithm. The number and weight of clusters is calculated for the normal and malicious network traffic graph. Then, the attack is detected using the differences of these parameters. The results represent that the approach detects the attack in the network flows in a high accuracy. Also, the normal and dictionary attack traffic in network flows are modeled using Markov chain. Consequently, three phases are assigned for the behavior of dictionary attack: scanning phase, brute-force phase and die-off phase. The values of these parameters are precisely calculated using the CAIDA data set. Approximately, the artificial traces could be used for estimating the statistical parameters such as average, min, max and correlation coefficient. The model is appropriately capable for simulating the behavior of network traffic. In the end, for future works, we recommend changing the size of time intervals and creating also Markov models to achieve high accuracy in detection rate. Also, using graph clustering algorithm, other parameters such as maximum degree of nodes in the clusters, or number of edges in greatest cluster, and some other parameters can be calculated.

REFERENCES

[1] A. Sperotto, G. Schaffrath, R. Sadre, C. Morariu, A. Pras, B. Stiller, "An Overview of IP Flow-Based Intrusion Detection", Communications Surveys & Tutorials, Vol. 12, No. 3, pp. 343-356, 2010

[2] R. Hofstede, V. Bartos, A. Sperotto, A. Pras, "Towards real-time intrusion detection for NetFlow and IPFIX", 9th International Conference on Network and Service Management (CNSM)", pp. 227-234, 2013

[3] N. Hoque, D. K. Bhattacharyya, J. K. Kalita, "FFSc: a novel measure for low-rate and high-rate DDoS attack detection using multivariate data analysis", Security and Communication Networks, Vol. 9, No. 13, pp. 2032-2041, 2016

[4] P. Hick, E. Aben, K. Claffy, J. Polterock, The CAIDA DDoS attack 2007 dataset, 2007

[5] Y. Gao, Z. Li, Y. Chen, "A DoS Resilient Flow-level Intrusion Detection Approach for High-speed Networks", 26th IEEE International Conference on Distributed Computing Systems (ICDCS 06), pp. 39-46, 2006

[6] T. Dubendorfer, B. Plattner, "Host behavior based early detection of worm outbreaks in internet backbones", 14th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprise (WETICE 05), pp. 166–171, 2005

[7] G. Androulidakis, S. Papavassiliou, "Intelligent Flow-Based Sampling for Effective Network Anomaly Detection", IEEE Global Telecommunications Conference (GLOBECOM 07), pp. 1948–1953, 2007

[8] M. J. Chapple, T. E. Wright, R. M. Winding, "Flow Anomaly Detection in Firewalled Networks", Securecomm and Workshops, pp. 1–6, 2006

[9] P. Barford, D. Plonka, "Characteristics of network traffic flow anomalies", IMW 01: Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement, pp. 69-73, 2001

[10] L. Akoglu, H. Tong, D. Koutra, "Graph based anomaly detection and description: a survey", Data Mining and Knowledge Discovery, Vol. 29, No. 3, pp. 626-688, 2015

[11] K. Henderson, T. Eliassi-Rad, C. Faloutsos, L. Akoglu, L. Li, K. Maruhashi, B.A. Prakash, H. Tong, "Metric forensics: A multi-level approach for mining volatile graphs", 16th ACM International Conference on Knowledge Discovery and Data Mining, pp. 163-172, 2010

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

[13] Q. Ding, N. Katenka, P. Barford, E. D. Kolaczyk, M. Crovella, "Intrusion as (anti) social communication: characterization and detection", 18th ACM International Conference on Knowledge Discovery and Data Mining, pp. 886-894, 2012

[14] L. Akoglu, M. McGlohon, C. Faloutsos, "OddBall: Spotting anomalies in weighted graphs", 14th Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 410-421, 2001

[15] P. Bonacich, P. Lloyd, "Eigenvector-like measures of centrality for asymmetric relations", Social Networks, Vol. 23, No. 3, pp. 191-201, 2001

[16] B. Perozzi, L. Akoglu, P.L. Sanchez, E. Muller, "Focused clustering and outlier detection in large attributed graphs", 20th ACM Special Interest Group on Knowledge Discovery and Data Mining (SIG-KDD), pp. 1346-1355, 2014

[17] C. Liu, X. Yan, H. Yu, J. Han, P.S. Yu, "Mining behavior graphs for backtrace of noncrashing bugs", 5th SIAM International Conference on Data Mining, pp. 286-297, 2005

[18] S. Gunnemann, I. Farber, B. Boden, T. Seidl, "Subspace clustering meets dense subgraph mining: A synthesis of two paradigms", 10th IEEE International Conference on Data Mining (ICDM), pp. 845-850, 2010

[19] X. Xu, N. Yuruk, Z. Feng, T. A. J. Schweiger, "Scan: a structural clustering algorithm for networks", 13th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), pp. 824-833, 2007

[20] S. Chakrabarti, "Dynamic personalized page rank in entity-relation graphs", 16th International Conference on World Wide Web (WWW), pp. 571-580, 2007

[21] J. Neville, D. Jensen, "Iterative classification in relational data", AAAI Workshop on Learning Statistical Models from Relational Data, pp. 13-20, 2000

[22] K. M. Kapsabelis, P. J. Dickinson, K. Dogancay, "Investigation of graph edit distance cost functions for detection of network anomalies", 13th Biennial Computational Techniques and Applications Conference (CTAC 06), pp. 436-449, 2006

[23] T. Ide, H. Kashima, "Eigenspace-based anomaly detection in computer systems", 10th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), pp. 440-449, 2004

[24] M. Kuramochi, G. Karypis, "Frequent subgraph discovery", 2001 IEEE International Conference on Data Mining (ICDM), pp. 313-320, 2001

[25] D. Chakrabarti, "Autopart: parameter-free graph partitioning and outlier detection", 8th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD), pp. 112-124, 2004

[26] D. Chakrabarti, R. Kumar, A. Tomkins, "Evolutionary clustering", 12th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), pp. 554-560, 2006

[27] C. Tantipathananandh, T. Berger-Wolf, "Constant-factor approximation algorithms for identifying dynamic communities", 15th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), pp. 827-836, 2009

[28] M. Mongiovi, P. Bogdanov, R. Ranca, A. K. Singh, E. E. Papalexakis, C. Faloutsos, "Netspot: Spotting significant anomalous regions on dynamic networks", 13th SIAM International Conference on Data Mining (SDM), pp. 1-9, 2013

[29] L. Peel, A. Clauset, Detecting change points in the large-scale structure of evolving networks, CoRR, abs/1403.0989, pp. 38-53, 2014

[30] D. Doval, S. Mancoridis, B. S. Mitchell, "Automatic Clustering of Software Systems using a Genetic Algorithm", 1999 International Conference on Software Tools and Engineering Practice (STEP 99), pp. 73-81, 1999

[31] J. Karimpour, S. Lotfi, A. Tajari Siahmarzkooh, "Intrusion detection in network flows based on an optimized clustering criterion", Turkish Journal of Electrical Engineering & Computer Sciences, accepted for publication: 10.3906/elk-1601-105.

AUTHORS PROFILE

Aliakbar Tajari Siahmarzkooh received the B.Sc. in Software Engineering from the Ferdowsi University of Mashhad, Iran, the M.Sc. degree in Computer Science from the University of Tabriz, Iran. He is a Ph.D student at the University of Tabriz. His research interests include network security, cryptography and intrusion detection systems.

Jaber Karimpour received the B.Sc. Degree in Computer Science and Applied Mathematics from the University of Tabriz (Iran) in 1998, the M.Sc. Degree specializing in the computer systems are of Applied Mathematics from the University of Tabriz in 2000, and the Ph.D. degree in Computer Systems form the University of Tabriz. He is currently an Assistant Professor in the Department of Computer Sciences at the University of Tabriz and has been the manager of Information Technology of the university since 2011. His current research interests include cryptography, network security, formal specification, and verification.

Shahriar Lotfi received the B.Sc. in Software Engineering from the University of Isfahan, Iran, the M.Sc. degree in Software Engineering from the University of Isfahan, Iran, and the Ph.D. degree in Software Engineering from the Iran University of Science and Technology. He is Assistant Professor of Computer Science at the University of Tabriz. His research interests include compilers, super- compilers, parallel processing, evolutionary computing and algorithms.