A Network-based Framework for RAT-Bots Detection

Ahmed A.Awad
Department of Electronics, Communications, and Computers Engineering
Helwan University,
Cairo, Egypt
Ahmed.shafie@h-eng.helwan.edu.eg

Samir G. Sayed
Department of Electronics, Communications, and Computers Engineering
Helwan University,
Cairo, Egypt
Egyptian Computer Emergency Readiness Team (EG-CERT)
National Telecom Regulatory Authority (NTRA), Egypt
s.sayed@ee.ucl.ac.uk

Sameh A. Salem
Department of Electronics, Communications, and Computers Engineering
Helwan University,
Cairo, Egypt
sameh_salem@h-eng.helwan.edu.eg

***Abstract*— Remote access Trojans (RATs) are used by attackers to compromise and control the victim machine. In this work, a novel Network-based framework is introduced for detecting RAT bots based on data mining techniques. Several machine learning (ML) techniques are used to differentiate between benign and RAT infected machines. Various performance measurements are used to evaluate the performance of the proposed framework. Experimental results demonstrate that the proposed system can achieve accuracy of 97.2% with 3.8% false positive rate.**

Keywords—Botnets, network-based detection, machine learning techniques, classification, information gain

I. INTRODUCTION

Most of today activities are done through the Internet, such as Internet banking, online shopping, online games, chatting, emails, and etc. The advances in the usage of internet technology are confronted with an increase in the risks that facing internet users due to the existence of malicious soft wares (malware) such as viruses, worms, computer trojans, backdoors, etc. botnet is amongst the unsafe threats confronting web clients which is viewed as the base of most digital violations. botnet is a group of infected machines, called bots, under the influence of an intruder called botmaster, who uses communication channels called Command and Control (C&C) channels to conduct coordinated attacks such as distributed Denial-of-service attacks (DDOS), phishing, spam emails, and identity of theft. several types of botnets exist which uses different famous protocols to perform their C&C communication such as internet relay chat (irc) or hyper-text transfer (http) protocols [1][2]. irc and http botnets are often used to conduct coordinated attacks such as DDOS, phishing, and spam emails [3]. remote access Trojans (RATs) is a malicious software designed as advanced persistent threat (APTs) for targeted attacks such as stealing sensitive business and personal information or damaging the victim machine. RATs are Trojans injected into legitimate programs to hide their malicious behaviors. The software downloaded on the victim machine is called the RAT server, while the one that runs on the attacker machine called the RAT client. The RAT server is downloaded on the victim host using one of the intrusion tactics (e.g. drive-by-download, spear phishing [4]). The RAT server maintains user privileges gained during installation, disabling windows functions that could threaten its functionality. Connection between RAT server and client is established either in direct or reverse way. In direct connection, the RAT client is the entity which establishes the connection to the server. While, in reverse connection the client is the entity that starts the connection. Commonly, reverse connections are the most common method used by RATs as most of enterprise firewalls prevent connections established from outside the enterprise network. The attacker can infect more than one machine in the same network or in different networks constructing RAT botnet. The severity of RAT bots is their capability to by-pass detection procedures taken by users causing serious damages for the infected machines [5]. Bots/Botnets detection techniques are classified into host and network detection techniques. In host-based detection techniques, the system behavior of each single machine is monitored to detect any botnet-related suspicious behaviors. The main disadvantages for hot-based techniques are their vulnerability to hot-resident and stealthy malwares, as well as the difficulties presented in installing the monitored systems in each host in the network. In network-based detection techniques, the network traffic is monitored and then analyzed for detection of bot infected machines. In one hand when the monitored traffic is encrypted, the botnet detection capability is reduced. On the other hand, monitoring the network traffic may raise concerns about the data privacy. Several researches have been conducted on detecting botnets using network analysis.

Guofei et. al. in [8] developed a passive network monitoring tool called BotHunter, where BotHunter can detect bots based on infection sequence. The problem of this tool is that any bot having different cycle cannot be detected. Guofei et al. [9] introduce a detection framework called botminer. This framework is used for botnet detection independent of the communication protocol. botminer detect botnets by grouping hosts have the same communication and activity traffic patterns. For success detection more than one infected host must be exist in the monitored network to cluster their traffic successfully. Goebel et al. [19] provide a technique to detect IRC bots depending on unusual nicknames used by bots subscribed on IRC channels for communication with botmaster. Binkley et al. [12] introduced a method for anomaly detecting IRC botnet using TCP scan and IRC parsing. Strayer et al. [5] proposed a technique for IRC botnet detection by filtering and classifying the network flows according to specific flow features. This method is failed to detect bots with different protocols. Jiang and Omote in [10] develop an approach for detecting RAT network actions in their early stages using machine learning algorithms. Several machine learning techniques have been evaluated on a given set of network features to compare their detection results. Yamada M. and M. [11] develop an approach for detecting RAT infected machines based on their network activities. RAT alarm raised on hosts that have communication patterns established between RAT client and RAT server. Followed by reconnaissance network activities that is taken for compromising other hosts on the network using administrative network protocols.

In this paper a Network-based RAT bot detection framework is introduced. The proposed framework detects RAT server on the victim machine through a robust traffic analysis. The system behavior of a given machine is extracted and mapped into a group of feature vectors. After that, these feature vectors are classified through machine learning techniques to detect the existence of RATs behavior. The proposed framework is designed for detecting RAT bots without any consideration to the C&C protocol or architecture used by RAT botnets. Behavior analysis has been considered for avoiding encryption issues and privacy concerns. A set of network traffic features are selected according to a deep analysis for the network traffic produced from host infected machines. Moreover, the usage of machine learning (ML) techniques is considered a step towards detecting of zero day RAT bots and avoiding the problems raised by signature based techniques.

The paper is organized as follows. Section II describes the network-based RAT bot detection framework. The experimental results are shown in section III. Conclusions and future work is given in Section IV.

II. The Proposed Network-Based RAT-Bot Detection

In order to identify the RAT bots, a network-based detection framework is proposed. The proposed framework consists of flow-sniffer and flow-miner as illustrated in Fig.1. The flow-sniffer module captures the network traffic passing through the network router and produces reports. The flow-miner extracts a set of features from these patterns constructing feature vectors. Subsequently, ML techniques are applied on those vectors to identify the suspicious flows.

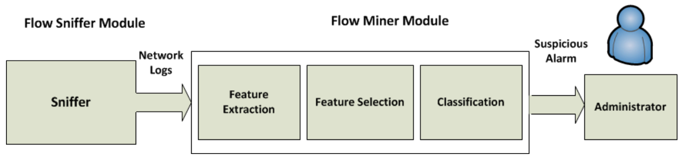

Fig. 1. Proposed Networsk-based framework

A. *Flow Sniffer Module:* This module is responsible for monitoring the traffic flows through the core switch of the monitored network. The monitoring is carried out through an open-source Java library (jNetPcap) for optimum packet decoding performance. This library enables us to capture network packets on either Microsoft Windows and Linux operating systems. In this context, the network packets that passed through the core switch are sent to the monitoring server machine by port mirroring where they are captured by the proposed flow sniffer module.

B. *Flow-Miner Module:* the network flow-miner module analyzes the reports that are generated by the flow-sniffer module to identify the RAT patterns. This module consists of three stages. The first stage is the feature extraction stage, while the second is the feature selection stage. Finally the classification stage is the third stage.

In the *feature extraction* stage, the network packets are captured to extract the required network features. The extracted network features are statistical features to eliminate privacy concerns and avoid dealing with the encrypted traffic. Commonly, RATs hide their communication traffic, where less communication patterns are carried between RAT servers and their clients. While benign applications are commonly having dense communication traffic. For RAT infected machines; the inbound traffic represented in the commands sent from client to sever is less than the outbound traffic which represents the results of client queries. This behavior differs from the corresponding network behavior of normal hosts that have large inbound traffic with respect to their outbound (e.g. video streaming, web browsing). Accordingly, these observations have a direct effect on the payloads, packets, inter arrival time (IAT) and first packet sizes in the forward and backward directions that could be used to differentiate between normal and RAT infected hosts. Consequently, the network features F0 to F25 are selected to achieve this goal as depicted in Table I.

TABLE I. Selected Network Features

Feature	Name	Description	Flow Direction
F0	PacNum	Packet number	Forward
F1	NOB	Number of bytes	Forward
F2	ABS	Average Packet Size	Forward
F3	MABS	Maximum Packet Size	Forward

F4	MIBS	Minimum Packet Size	Forward
F5	IAT	Inter arrival time between packets	Forward
F6	APS	Average PayLoad size	Forward
F7	HL	Header Length	Forward
F8	FPacSize	First packet size	Forward
F9	NSP	Number of small/null packets	Forward
F10	NPS	Number of setting PSH flag	Forward
	NUS	Number of setting URG flag	Forward
	NFS	Number of times FIN flag set	Forward
F11	PacNumB	Packet number	Backward
F12	NOBB	Number of bytes	Backward
F13	ABSB	Average Packet Size	Backward
F14	MABSB	Maximum Packet Size	Backward
F15	MIBSB	Minimum Packet Size	Backward
F16	IATB	Inter arrival time between packets	Backward
F17	APSB	Average PayLoad size	Backward
F18	HLB	Header Length	Backward
F19	FPacSizeB	First packet size	Backward
F20	NSP	Number of small/null packets	Backward
F21	NPSB	Number of setting PSH flag	Backward
F22	NUSB	Number of setting URG flag	Backward
F23	NFSB	Number of times FIN flag set	Backward
F24	NONI	No. op packets/No. ip packets	
F25	SOSI	size op packets/size ip packets	

For the *feature Selection* stage, it is required to select the significant features to reduce training and testing time without affecting the performance of the detection results. In this context, the features that have the least redundant information and most effect in class selection are selected. In this paper, a ranker algorithm is used to evaluate each attribute's gain to the class according to Eq. (1). Therefore, the selected features are shown in Table II.

$$IG(Class\ , Attr\ ) = H(Class\ ) - H(Class\ \mid Attr\ ) \quad (1)$$

TABLE II. REDUCED NETWORK FEATURES

Feature	Name	Description	Flow Direction
F0	PacNum	Packet number	Forward
F1	NOB	Number of bytes	Forward
F2	ABS	Average Packet Size	Forward
F3	MABS	Maximum Packet Size	Forward
F4	FPacSize	First packet size	Forward
F5	APS	Average PayLoad size	Forward
F6	SOSI	size op packets/size ip packets	
F7	NPS	Number of times PSH flag set	Forward
F8	NFS	Number of times FIN flag set	Forward
F9	NOB	Number of bytes	Backward
F10	MABSB	Maximum Packet Size	Backward
F11	MIBSB	Minimum Packet Size	Backward
F12	FPacSizeB	First packet size	Backward
F13	HLB	Header Length	Backward
F14	NFSB	Number of times FIN flag set	Backward

It should be noted that a feature vector is constructed for each host in the monitored network containing fifteen features. These features are sent to a set of ML techniques to get a reliable decision about the monitored hosts.

In *Classification* stage, the feature vectors that are constructed by the feature extraction stage are examined, and tested. Then, the classification process decides whether these vectors representing benign applications or RAT activities. To achieve this goal, four machine learning techniques: Random Forest (RF), Bayesian Network (BN), Naïve Bayes Tree (NBTree), K-Nearest Neighbor (KNN) is used. The classifiers have been trained using data acquired from benign applications and RAT bots.

Random Forest (RF): To enforce the stability and accuracy of DT, a bagging method is used to generate a number of sub-decision trees. The final decision is taken by averaging output decisions generated from the trees[14].

Naive Bayes Tree (NBTree): This classification technique is hybrid between decision tree (DT) algorithm and naïve bayes technique (NB) to improve their accuracy instead of using each technique individually. For classifying a given sample a decision tree is built with the existence of the Naive Bayes classifiers instead of the tree leaves. NBTree classifier is used according to it's adaptively and efficiency [16].

KNN (lazy classifiers): This method is used for data classification by determining the class of the K-nearest points to the data sample. Nearest Neighbor algorithms are selected as a reason of their popularity and simplicity in machine learning and classification applications [17]. With KNN algorithms no need for training procedure and the overfitting problems is avoided with reasonable classification accuracy. The main disadvantage of KNN algorithms is the performance degradation when the scale of the training set is small. The distance can be calculated by several methods such as Euclidean distance. According to this method, The distance between two points X=(x_1, x_2), Y= (y_1, y_2) is calculated using formula in Eq. (4).

$$D(X,\ Y) = \sqrt{(y_1 - x_1)^2 - (y_2 - x_2)^2} \quad (4)$$

***Bayesian Network (BN)*:** The Bayesian network belongs to a family of probabilistic graphical models [7]. in this model random variables are represented by nodes, whereas links represent the probability amongst this variables.

The starting node and ending node of an arc are called, respectively, the predecessor, and descendant of each other.The classifier based on Bayes rule which states for calculating the posterior probability for a given data x and class C. Given a training set $D = \{X_1, \ldots, X_j, \ldots, X_n\}$, such that $X_j = \{x_{j,1}, \ldots, x_{j,i}, \ldots, x_{j,v}\}$, for a group of class labels $C = \{c_i\}$ given a data X , BN algorithms focus on identifying the maximum posterior probability $P(c|X)$, where

$$\max_{c_i \in C} P(c_i \mid X) = \frac{P(X \mid c_i)P(c_i)}{P(X)} \quad (5)$$

where $P(c \mid X)$ is the posterior probability of data X with respect to class c , $P(c\ )$ is the prior probability of class c , and $P(X \mid c)$ is the conditional probability of data x given class c . Where the joint probability distribution of event X_j is calculated as in Eq. (6)

$$P(x_1, \ldots, x_k) = \prod_{i=1}^{k} P(x_i \mid Parents\ (x_i)) \quad (6)$$

III. Experimental results

To evaluate the performance of the proposed framework, several controlled experiments have been conducted on virtual network environment. The experimental environment consists of four virtual machines to setup the network. The first virtual machine is Windows 7 acting as RAT client. Three Windows 7 virtual machines are used as victim machines which are infected by RAT servers. The virtual machines connected with each other via a virtual network. The training set for each classifier is divided into benign and malicious datasets. Benign dataset is collected by monitoring the normal behavior of different hosts running windows 7 as well as DARPA dataset that has been used for capturing the benign network behavior for different hosts in the monitored network resulting in capture a total of 1610 flow. The malicious traffic is collected from the behavior of eight botnets which are Cerberus, XtremeRAT, PandoraRAT, GremmRAT, Storm, Turkojan and NovaliteRAT, Waledac botnets. These RATs are obtained from the Egyptian Computer Emergency Readiness Team (EG-CERT) [26]. The benign and malicious datasets are partitioned into training and testing subsets. The training subset is used to train the classifiers with known benign applications and the seven RATs that are mentioned above. The testing set is used to evaluate the proposed framework over unknown RAT bots and benign applications. The procedure of selecting the training subset is based on 10-fold cross-validation (CV). The experiments are conducted using Matlab tool. The parameters for each machine learning algorithm are tunable to get an acceptable performance on the obtained dataset.

TABLE III. 10-FOLD CV RESULTS OF PROPOSED FRAMEWORK

ML	Acc.(%)	TPR (%)	FPR (%)	TNR (%)	FNR (%)	AUC
RF	99.75	100	0.497	99.50	0.00	1.00
BayesianNet	99.44	99	1.0	100.00	1.00	1.00
NBTree	99.32	99	4.0	99.00	1.00	1.00
KNN	98.70	98.63	1.24	98.76	1.37	0.99

TABLE IV. 10-FOLD CV RESULTS FOR REDUCED FEATURES OF PROPOSED FRAMEWORK

ML	Acc.(%)	TPR (%)	FPR (%)	TNR (%)	FNR (%)	AUC
RF	99.63	99.63	0.37	99.63	0.37	0.99
BayesianNet	99.44	99.00	0.00	100.00	1.00	1.00
NBTree	99.44	99.00	0.00	100.00	1.00	0.99
KNN	98.70	98.63	1.24	98.76	1.37	0.99

For evaluation purposes, the used Key Performance indications (KPI) are: the overall accuracy (ACC), false positive rate (FPR), false negative rate (FNR), true positive rate (TPR) and true negative rate (TNR). These parameters are calculated as follows.

$$\begin{aligned} ACC &= \frac{T_P + T_N}{T_P + T_N + F_P + F_N} \\ TPR &= \frac{T_P}{T_P + F_N} \\ FPR &= \frac{F_P}{F_P + T_N} \\ TNR &= \frac{T_N}{T_N + F_P} \\ FNR &= \frac{F_N}{F_N + T_P} \end{aligned} \quad (7)$$

where T_P represents the number of detected malicious samples. T_N is the number of detected benign samples. F_P denotes the number of false detected samples as malicious. F_N represents the number of false detected samples as benign. Tables III and IV show the performance of proposed framework with different classifiers.

As shown in Table IV, Random Forest (RF) provides the best performance, more than 99% accuracy with 0.37% FPR when the number of trees is set to 20 as depicted in Table VI and Fig.3. Different tree numbers are evaluated using different KPIs and ROC curves as shown in Table V and Fig.2 for Full feature set, while their results for the reduced feature set are presented in Table VI and Fig.3. This performance resulted from the bagging methodology which avoids the DT over fitting problems. The main disadvantage of RF is the requiring of a higher training time. The KNN classifiers afford 98.70% accuracy with low FPR at K = 1. The main drawback is that the performance of KNN is dependable on the training set size. Different K values are evaluated using different KPIs and ROC curves as shown in Table VII and Fig.4 for Full feature set, while their results for the reduced feature set are presented in Table VIII and Fig.5.

TABLE V. 10-FOLD CV RESULTS FOR DIFFERENT TREE NUMBERS FOR RANDOM FOREST CLASSIFIER

Tnum	Acc.(%)	TPR (%)	FPR (%)	TNR (%)	FNR (%)	AUC
10	99.63	100	0.74	99.25	0.00	0.99
20	99.69	100	0.62	99.38	0.00	1.00

50	**99.75**	**100**	**0.5**	**99.50**	**0.00**	**1.00**
100	99.69	100	0.62	99.38	0.00	1.00
200	99.69	100	0.62	99.38	0.00	0.99

TABLE VI. 10-FOLD CV RESULTS FOR DIFFERENT TREE NUMBERS FOR RANDOM FOREST CLASSIFIER ON REDUCED FEATURE SET

Tnum	Acc.(%)	TPR (%)	FPR (%)	TNR (%)	FNR (%)	AUC
10	99.57	99.63	0.49	99.50	0.37	0.99
20	**99.63**	**99.63**	**0.37**	**99.63**	**0.37**	**0.99**
50	99.50	99.63	0.62	99.38	0.37	0.99
100	99.50	99.63	0.62	99.38	0.37	0.99
200	99.63	99.88	0.38	99.38	0.12	1.00

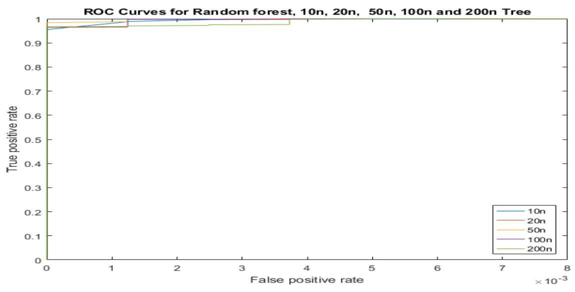

Fig. 2. ROC curve representing RF classifier using various number of trees

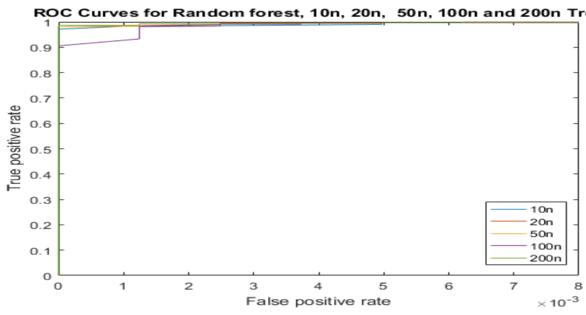

Fig. 3. ROC curve representing RF classifier using various number of trees on reduced feature set

TABLE VII. 10-FOLD CV RESULTS OF K- NEAREST NEIGHBOR (KNN) CLASSIFIER USING DIFFERENT K VALUES

K value	Acc.(%)	TPR (%)	FPR (%)	TNR (%)	FNR (%)	AUC
1	**98.70**	**98.6335**	**1.2422**	**98.7578**	**1.3665**	**0.9870**
2	98.32	98.1366	1.4907	98.5093	1.8634	0.9886
3	98.14	97.7640	1.4907	98.5093	2.2360	0.9895
4	97.95	97.6398	1.7391	98.2609	2.3602	0.9912
5	97.58	97.2671	2.1118	97.8882	2.7329	0.9922
6	97.39	97.2671	2.4845	97.5155	2.7329	0.9925
7	97.27	96.8944	2.3602	97.6398	3.1056	0.9913

TABLE VIII. 10-FOLD CV RESULTS OF K- NEAREST NEIGHBOR (KNN) CLASSIFIER USING DIFFERENT K VALUES ON REDUCED FEATURE SET

K value	Acc.(%)	TPR (%)	FPR (%)	TNR (%)	FNR (%)	AUC
1	**98.70**	**98.6335**	**1.2422**	**98.7578**	**1.3665**	**0.9870**
2	98.32	98.1366	1.4907	98.5093	1.8634	0.9886
3	98.14	97.7640	1.4907	98.5093	2.2360	0.9895
4	97.95	97.6398	1.7391	98.2609	2.3602	0.9912
5	97.58	97.2671	2.1118	97.8882	2.7329	0.9916
6	97.39	97.2671	2.4845	97.5155	2.7329	0.9925
7	97.27	96.8944	2.3602	97.6398	3.1056	0.9913

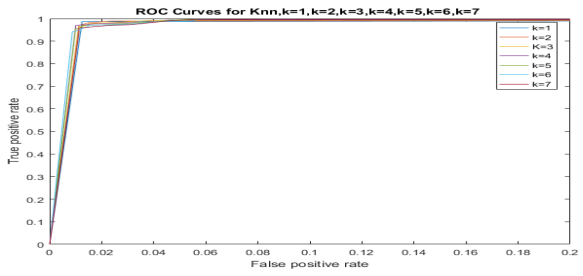

Fig. 4. ROC curve for KNN classifier using various K values

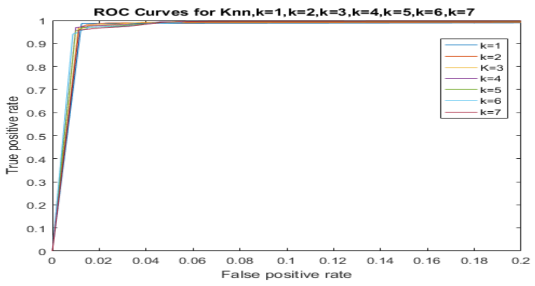

Fig. 5. ROC curve for KNN classifier using various K values for reduced feature set

To examine the reliability of proposed framework, the performance is evaluated against seven unknown RAT bots using the classifier which provide the highest detection rate with low positive rate. In the proposed framework, the main concern is on the false positive rates rather than the false negative rates. Accordingly, Random Forest (RF) is chosen for the testing phase. In this context, the seven RAT botnets used in the testing phase are SpyNet, DarKComet, NJRAT, SolitudeRAT, SchwarzesonneRAT, ProRAT, and Zeus. In addition, for evaluating false positive rate, several benign applications have been tested especially the ones that have similar behavior to the RAT-bots. The considered benign applications are the browsing applications such as Spark, antivirus signature updates as Avira update, desktop applications as Word, Excel, Netbeans IDE, Explorer, Email client as outlook, yahoo mail, and chatting application such as IRC client (mIRC). Results are shown in Table IX, as well as the ROC for RF classifier is shown in Fig.6.

Although, it is essential to compare the proposed methodology with other competitive methodologies, the comparison process is difficult to achieve due to several factors such as the unavailability of proper documentation which helps in the accurate realization for each methodology. Furthermore, the difficulty of obtaining the same botnets used by the other methods to be a common ground in the evaluation process.~~,~~ Also, the absence of common comparison methods and the lack of unified evaluation metrics among the introduced frameworks in the literature as reported in [13]. This gives another challenge for the comparison process.

TABLE IX. EXPERIMENTAL RESULTS OF PROPOSED FRAMEWORK

ML	Acc.(%)	FPR (%)	FNR (%)	TNR (%)	TPR (%)	AUC	Testing time (s)
RF	97.20	3.80	0	96.21	100	0.9691	0.08

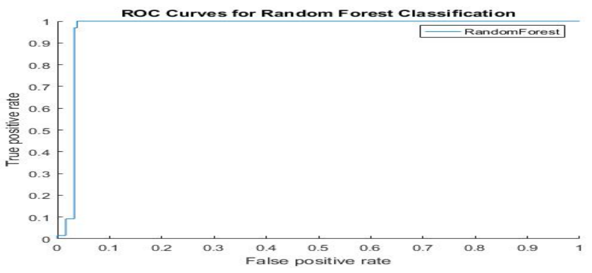

Fig. 6. ROC curve for RF classifier

IV. CONCLUSION

In this paper a Network-based framework for RAT detection has been introduced. The proposed framework relies on traffic features that describe the network behavior of the monitored hosts. The framework consists of two modules which are Flow-Monitor which extracts flow features and Flow-Miner which takes the decision about each monitored host in the system. The experimental results using several machine learning (ML) techniques are carried out then, a test phase is conducted for evaluating the system performance against unknown threats and applications. Experimental results show that the proposed system can achieve accuracy 97.20% with 3.8% false positive rate. For the near future, a hybrid network and host based technique is to be introduced to achieve further improvements in the performance of the proposed framework, and considering other types of bots and malwares.

ACKNOWLEDGMENT

This work has been supported by the Egyptian Computer Emergency Readiness Team (EG-CERT), National Telecom Regulatory Authority (NTRA), Egypt. The authors would like to thank Professor Sherif Hashem and Eng. Waleed Zakria for providing useful suggestions.

REFERENCES

[1] J. B. Grizzard, V. Sharma, C. Nunnery, B. B. Kang, and D. Dagon, "Peer-to-peer botnets: Overview and case study," in Proceedings of the First Conference on First Workshop on Hot Topics in Understanding Botnets, ser. HotBots'07. Berkeley, CA, USA: USENIX Association, 2007, pp. 1–1.

[2] M. Stevanovic and J. Pedersen, "An efficient flow-based botnet detection using supervised machine learning," in International Conference on Computing, Networking and Communications (ICNC). IEEE, Feb. 2014, pp. 797 – 801.

[3] U. Y. T. S. Yamada M., Morinaga M. and M. Takenaka, "Rat-based malicious activities detection on enterprise internal networks," in The 10th International Conference for Internet Technology and Secured Transactions (ICITST-2015), 2015, pp. 321–325.

[4] Y. Liang, G. Peng, H. Zhang, and Y. Wang, "An unknown Trojan detection method based on software network behavior," Wuhan University Journal of Natural Sciences, vol. 18, no. 5, pp. 369–376, 2013.

[5] W. Strayer, D. Lapsely, R. Walsh, and C. Livadas, "Botnet detection based on network behavior," in Botnet Detection, ser. Advances in Information Security, W. Lee, C. Wang, and D. Dagon, Eds. Springer US, 2008, vol. 36, pp. 1–24.

[6] F. Y. W. Law, K. P. Chow, P. K. Y. Lai, and H. K. S. Tse, A Host-Based Approach to BotNet Investigation Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 161–170.

[7] J. Han, M. Kamber, and J. Pei, "6 - mining frequent patterns, associations, and correlations: Basic concepts and methods," in Data Mining (Third Edition), third edition ed., ser. The Morgan Kaufmann Series in Data Management Systems, J. Han, M. Kamber, , and J. Pei, Eds.

[8] Guofei Gu, Phillip Porras, Vinod Yegneswaran, Martin Fong, and Wenke Lee. Bothunter: Detecting malware infection through ids-driven dialog correlation, 2007.

[9] Guofei Gu, Roberto Perdisci, Junjie Zhang, and Wenke Lee. Botminer: Clustering analysis of network traffic for protocol- and structure independent botnet detection. In Proceedings of the 17th Conference on Security Symposium, SS'08, pages 139-154, Berkeley, CA, USA, 2008. USENIX Association.

[10] D. Jiang and K. Omote. An approach to detect remote access trojan in the early stage of communication. In 2015 IEEE 29th International Conference on Advanced Information Networking and Applications, pages 706-713, March 2015.

[11] Unno Y. Torii S. Yamada M., Morinaga M. and Takenaka M. Rat-based malicious activities detection on enterprise internal networks. In The 10th International Conference for Internet Technology and Secured Transactions (ICITST-2015), pages 321-325, 2015.

[12] J. R. Binkley and S. Singh, "An algorithm for anomaly-based botnet detection," Proceedings of the 2nd Conference on Steps to Reducing Unwanted Traffic on the Internet, 2006. [Online]. Available: http://dl.acm.org/citation.cfm?id=1251296.1251303

[13] S. García, M. Grill, J. Stiborek, and A. Zunino. An empirical comparison of botnet detection methods. Comput. Secur., 45:100_123, September 2014.

[14] Kamaldeep Singh, Sharath Chandra Guntuku, Abhishek Thakur, and Chittaranjan Hota. Big data analytics framework for peer-to-peer botnet detection using random forests. Information Sciences, 278(Complete):488-497,2014.

[15] P. Barthakur, M. Dahal, and M. K. Ghose. A framework for p2p botnet detection using svm. In 2012 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery, pages 195-200, Oct 2012.

[16] Yu Liang, Guojun Peng, Huanguo Zhang, and Ying Wang. An unknown trojan detection method based on software network behavior. Wuhan University Journal of Natural Sciences, 18(5):369-376, 2013.

[17] S. Garg, A. K. Singh, A. K. Sarje, and S. K. Peddoju. Behaviour analysis of machine learning algorithms for detecting p2p botnets. In 2013 15th International Conference on Advanced Computing Technologies (ICACT), pages 1-4, Sept 2013.

[18] EG-CERT, http://www.egcert.eg/.

[19] J. Goebel and T. Holz, "Rishi: Identify bot contaminated hosts by irc nickname evaluation," in Proceedings of the First Conference on First Workshop on Hot Topics in Understanding Botnets, ser. HotBots'07. Berkeley, CA, USA: USENIX Association, 2007, pp. 8–8. [Online]. Available:http://dl.acm.org/citation.cfm?id=1323128.1323136