Journal of Engineering Science and Technology
Vol. 16, No. 1 (2021) 152 - 164

USING MONKEY OPTIMIZATION ALGORITHM TO DETECT NERIS BOTNET

IBRAHIM A. SALEH*, MAISIRREEM A. KAMAL, LAHEEB M. IBRAHIM

College of Computer Sciences and Mathematics,
University of Mosul, Aljamia , Mosul, Nenava, Iraq
*Corresponding Author: i.hadedi@gmail.com

Abstract

Due to evolution on the Internet of Things (IoT), and because Internet applications occupy a lot of space in human life and with the increasing uses of computers every day, also threat posed by Botnets that infect a large number of and using them together to perform several malicious actions has become a growing issue to the Internet security. Neris Botnet has a great threat on computers, where Nersi BotMaster can know any password when the victim entered to his computer, also the Neris BotMaster can direct the victim's apparatus to send any data he want it from his computer, and can change any password on his device, and strikingly, the Neris BotMaster can see all operation the victim's doing on his computer, by enticing the victim to download a Neris Botnet by clicking on the link set by the master of Botnet, and spam messages, by enticing the victim to download a Neris Botnet by clicking on the link set by the master of bot, and spam messages. In this paper, a Spider Monkey Optimization (SMO) Algorithm is suggested to detect Neris Botnet using CTU-13 datasets (2' Scenarios (Botnet-42 and Botnet-43). The Botnet detection rate for Botnet-42 is 99.8% and detection rate for Botnet-43 is 97.7% due to using SMO

Keywords: CTU-13, Internet of Things (IoT), Internet Relay Chat (IRC), Neris Botnet, Peer to Peer(P2P), Spider monkey optimization Algorithm.

1. Introduction

Internet of Things (IoT) is a new topic in information technology with the rapid growth of technology and with the growing concern for internet usage, IOT has become a very important topic whereas the Internet is full of threats to Internet users and these threats are the production of technology that has been exploited by Internet hackers for malicious uses, Botnet is an example of using good techniques for bad intentions. The Botnet is defined as a malware that allows an attacker to take control of the infected machine, the controls of these bots are called Botmasters [1].

The Botnet is currently considered as a major threat to the Internet. When a Botnet hits a computer connected to the Internet, the computer works normally, but an attacker can control the Botnet and steal important information, send harmful information, or take personal data. Botnet programs is stealthy during its whole life cycle, it had generated a relatively network footprint and most of the time remains ideal for stealing information.

Botnet affects a variety of infected devices that work together to perform a wide range of harmful activities, such as DDoS attacks, stealing important information, and creating spam email [2]. Attackers can cause significant damage to the Botnet, especially as these attacks pose a serious threat to Internet users and difficult to prevent damage. This list demonstrates number of threats; these are Sniffing Traffic, spreading new malware, Google AdSense abuse, Attacking IRC Chat Networks, Manipulating online polls / games, … etc. [3]. With Botnets, users do not usually know that their computers can be a part of a fully configured Botnet network to infect other devices and launch cyber-attacks. the newly emerging Botnet, for example Star Wars' Twitter Botnet, Hajime Malware Botnet, WireX Android Botnet, The Reaper IoT Botnet, Satori IoT Botnet and Neris Botnet.

The Botnet lifecycle consists of creation, infection, rallying, waiting, execution phases as shown in Fig. 1. The Botnet scanning for unprotected computers, when Botnet finds computers without protection, Botnet starts to infect these computers and sending a report to BotMaster, until this moment the Botnet still hidden, it is declared by BotMaster to do its job (attack). Sometimes the attackers sending an email or use malicious Websites to infect computers connected to the Internet.

The detection of Botnet is an important research topic for researchers in the field of computer science where researchers used many techniques to detect Botnet. In this section, we review the research and studies in the field of Botanet detection in two directions, the first is the use of traditional methods in detection botnet, and the second direction is the use of artificial intelligence methods.

The traditional methods in detection botnet are reviewed in [3-7]. Thangapandiyan and Rubesh [3] studied and analysed different detection techniques based on user data and behaviour of the distributed computing environment. Where the researchers made a comparison on different types of Botnets, they found that the IRC Botnet has low Communicate latency and BotMasters be real control the bots. Also, the bots collapsed easily by closing IRC, and Peer to Peer displays more robust encryption and resistance to failure but they have a very high transmission time connection, where The HTTP Botnet hides their vulnerable activities through The HTTP protocol, but the Botnet is fully collapsed by closing web servers and BotMasters do not have full control over bots. Seshadri et al. [4] used decision trees,

Naïve Bayes, SVM and K Nearest Neighbour algorithms to detect botnet. The researchers found that decision tree is best one among other algorithms; it gives 99.9% positive detection. Obeidat et al. [5] proposed a system to detect peer-to-peer (P2P) Botnet. The results of the experiment showed that the system appear a high average true positive rate and very low average false positive rate during Botnet detection. Payam et al. [6] presented botnet detection system depended on two engines to detect network intrusion without any prior knowledge to prevent the occurrence of internal DDOS attacks in the future. He et al. [7] proposed Peer Digger, "a novel real-time system capable of detecting stealthy P2P bots" The Peer Digger identity P2P bots with a good result detection within 4 minutes.

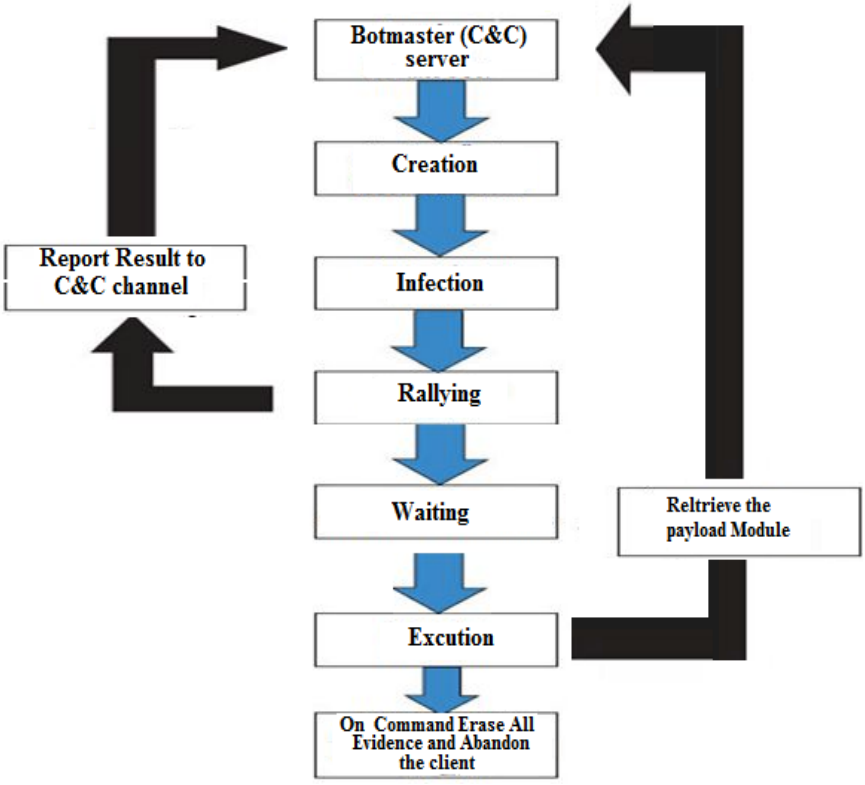

Fig. 1. Botnet lifecycle.

The Artificial intelligence methods in detection botnet are reviewed in [8-16], in the different detection techniques. Kuan et al. [8] produced a method for selecting a feature to detect Botnet using Support Vector Machines with Artificial Fish Swarm Algorithm (AFSA-SVM) method. Using AFSA only yields an accuracy rating slightly higher than the use has been suggested GA, but less time spent getting a fewer number of features subgroups. The experiment result shows that AFSA and Genetic algorithm are still available. It is applied to identify important features of robots, Filter unnecessary features, and use these algorithms in different applications easily. Mei and Gu-Hsin [9] proposed ant colony optimization algorithm (ACO) to detect a distributed scheme that discovers the paths from bots to C2 servers. The results show identify of botnet servers

efficiently. Huseynov et al. [10] proposed a bio-inspired computing technique called ant colony clustering for detect botnet attacks. Furthermore, it utilizes only a small sample of labelled data in the form of semi-supervised learning. Asadi et al. [11] introduced (BD-PSO-V) system to detect botnet. They based on particle swarm optimization (PSO) with a voting system. BD-PSO-V results had a promising performance against a variety of attacks. Habib et al. [12] and Zhang et al. [13] aimed to use particle swarm optimization (PSO) to detect botnet. Pektas et al. [14] and Wan-Chen and Hung-Min [15] used deep learning models to detect botnets. Homayoun et al. [16] uses Convolutional Neural Networks (CNNs) to detect botnet.

After reading researches and studies about how to detect Botnet, and because Botnets are dangerous and harmful to internet users where an attacker can control a computer's infection and steal important information, we studied Neris Botnet, read papers about how to find out Neris Botnet, and after studied a swarm optimization algorithms that have the ability to solve optimization problems that are inspired by simulated natural systems to represent and solve the problem efficiently, we decide to apply Spider Monkey Optimization(SMO) Algorithm to detect Neris Botnet.

The rest of this paper, to obtain a summary of previous studies and research in the detection Botnet, Section 2, discussed Neris Botnet, while Section 3 reviews spider monkey optimization (SMO) algorithm and why it is used in detection Neris Botnet, Section 4 describes a proposed work, Section 5 presents experiment results. Finally, Section 6 is for conclusion and future work.

2. Neris Botnet

The Neris Botnet is a Botnet that enables to use HTTP protocol-based channel called C&C. The Nersi Botnet effectiveness is (Connections by C&C channels, Send SPAM, implement click- cheat using some advertisement services). The dataset that captured by Nersi Botnet packet that used in detection phase have the following Characteristics [4, 17, 18].

Binary used	**Neris .exe**
Md5:	Bf08e6b02e00d2bc6dd493e93e698
Name:	Neris
IP address:	147.32.84.165
Capture duration:	6.15 hours
Size of complete pcap	52GB
NetFlow Size:	369MB
Size of Botnet Pcap	56MB
Windows named	"SARUMAN"
Infected	Virtual Environment
Label of IP in Net Flows files	"Botnet"

3. Spider Monkey Optimization (SMO)

In this paper, a Spider Monkey Optimization algorithm is used to detect Neris Botnet. SMO algorithm is a population based stochastic meta-heuristic [19]. Where

SMO is a bio-inspired technique based on the intelligent behaviour of Monkey, SMO are problem-independent and are efficient in solving real world complex optimization problems to arrive at the optimal solutions. Monkey behaviour-based algorithms are among the Swarm intelligence algorithms (SI) first proposed in 2007 [19]. SMO is a term that describes the collective behaviour of autonomic decentralized systems. SMO simulates behaviour of spider monkeys search foraging in intelligent way. Spider monkeys are live in social by divided themselves into small groups. Spider monkey property is as follows [19-22]:

- Each group consists of 40-50 spider monkeys.
- In each group, the oldest female member is a GL which takes almost all the decisions within the group.
- The group is divided into small groups (3 to 8 members) to search for food in an efficient way.
- In each subgroup, there is a local female member who is the leader to supervise the foraging itinerary details.

Spider monkey optimization (SMO) algorithm [19-24]

The Major phases in Spider Monkey Optimization are described in detail below:

- **Population initialization:**

  The initial parameters are of population N spider monkey population size, each ones SM_i $(i= 1,2, \ldots, N)$ is a vector of D-dimensional vector where D denotes the number of problem variables to be optimized and SM_i represent the position of i^{th} spider monkey (SM) in the population. Each SM identifies the potential solution to the problem. Each SM_i is calculated as Eq. (1)

$$SM_{ij} = SM_{min_j} + \varphi \left(SM_{max_j} - SM_{min_j} \right) \quad (1)$$

  where: SM_{minj} and SM_{maxj} are lower and upper bounds of SM_i in j^{th} direction respectively and $\varphi \in (0,1)$.

- **Local leader phase (LLP):**

  In this phase, the SMO update their current position by the members local group and local leader (LL), and calculate the fitness values for new monkeys' position. Also, in this phase a Spider monkeys must reach a new higher fitness value by continues update old position with new position. the location equation for ith SM as follows, Eq. (2)

$$SM_{newij} = SM_{ij} + rand[0,1]x\left(LL_{ki} - SM_{ij}\right) + rand[-1,1]x\left(SM_{rj} - SM_{ij}\right) \quad (2)$$

  where SM_{ij} is the *jth* dimension of the *ith SM*, LL_{kj} represents the *jth* dimension of the *kth* local group leader position. SM_{rj} is the *jth* dimension of the *rth SM* which is chosen randomly within *kth* group such that r≠ i in the *jth* dimension.

- **Global leader phase (GLP):**

  GL and LL group members introduce their experience to help all spider monkeys to modify their position. The location is calculated as Eq. (3).

$$SM_{newij} = SM_{ij} + rand[0,1]x\left(LL_{ki} - SM_{ij}\right) + rand[-1,1]x\left(SM_{rj} - SM_{ij}\right) \quad (3)$$

where GL_j is jth dimension of the *GL* location and $j \in \{1, 2, ...,D\}$ is the coincidentally chose index. In *GLP* phase, spider monkeys (SM_i) are updated their locations based on probabilities pi's those are considered utilize their fitness. In this way, there will be a chance for the best candidate to make himself ideally. The probability pi may be calculated using following expression:

$$p_i = \left(\frac{fitness_i x}{fitness_{max}}\right) + 0.1 \quad (4)$$

where *fitnessmax* is maximum fitness in the group of the *ith SM*. Further, A newly fitness function is calculated from generated position and compared with old one and choosing the best position.

- **Global leader learning (GLL) phase**:

  The GL in this phase is updated with the position of the best fit SM through greedy selection, if *GL* position is not updated, *Global Limit Count* is augmented by one.

- **Local leader learning (LLL) phase:**

  In this phase, the greedy selection helps LL to updated with the position of the best fit SM in that local group. After that compare, the updated position of LL and the old one. When the local leader is not updated, the *Local Limit Count* is incremented by one.

  Local leader decision (LLD) phase:

  If LLD update is not done with its own positions group members update by initial randomly or utilizing the information of *GL* and *LL*, this is done by perturbation rate, see Eq. (5).

$$SM_{newij} = SM_{ij} + \varphi_1 x\left(GL_j - SM_{ij}\right) + \varphi_2 x\left(SM_{ij} - LLsis_{ki}\right) \quad (5)$$

- **Global leader decision (GLD) phase:**

  The *GL* position in this phase is updated and watched for specific number of iterations called Global Leader Limit. Then the population is divided into number of groups starting at least two groups to a maximum extent possible number of groups by the *GL*. In the GLD phase, *LLL* operation is began in newly formed groups to select the *LL*. The *GL* cannot update their position When are formed the maximum number of groups, moreover, it pools all sub-groups into a single larger group based upon the inspiration from the fusion-fission structure of spider monkeys.

The SMO algorithm is shown below [23, 24]

SMO algorithm

Begin
Initialization value of control parameter
Initialize randomly the swarm using uniform distribution
The fitness value is calculated for each number of swarms
Global and Local leader are selected using greedy method depended on fitness value
Iteration = 0
While (stop criterion is not fulfilled) do

Calculate Local Leader Phase
Calculate Probability of each member of the swarm
Calculate Global Leader Phase
Calculate Global Leader Learning Phase
Calculate Local Leader Learning Phase
Calculate Local Leader Decision Phase
Calculate Global Leader Decision Phase
Iteration = Iteration + 1
End while
End of SMO algorithm

4. Proposed Method

The proposed work to detect Neris Botnet uses SMO algorithm consists of 3 phases, see Fig. 2.

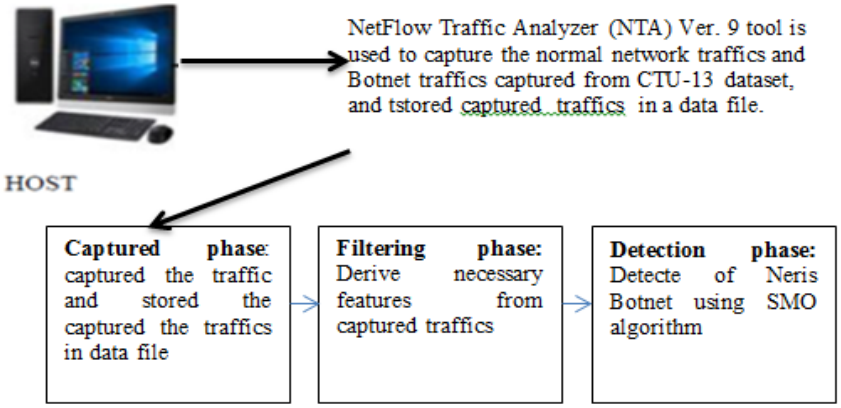

Fig. 2. Propose model.

- **<u>Captured phase</u>:**

  NetFlow Traffic Analyzer (NTA) Ver. 9 tool is used to capture the normal network traffics and Botnet traffics are captured from CTU-13 dataset, these traffics stored in a data file. The CTU-13 is a dataset of Botnet traffic that was captured in the CTU University, Czech Republic, in 2011. The CTU-13 dataset consists of thirteen captures (called scenarios) of different Botnet samples, as shown in Table 1. Each capture performs different actions. The total datasets are 13 [1]. In this paper, two datasets of Neris bot traffic are analysed. These two datasets contain bot traffic which uses HTTP protocol to perform click fraud attack. The analysis of Botnet traffic captures is explained in Table 2. The CTU-13 Neris Botnet Dataset is captured along with the packets of the whole CTU-13 department [25].

  The CTU-13 Neris botnet Dataset is captured along with the packets of the whole CTU-13 department. The first hour of capture was composed of only Background traffic and latter, run the malware. The malware was stopped 5 minutes before ending the capture. The limited the bandwidth of the experiment to 20kbps in the output of the bot. [25].

Table 1. Botnet types per scenario. Adapted from [1].

Botnet Family	Scenarios	C&C Protocol	Main Malicious Action
Neris	1, 2 and 9	IRC	Spam, fraud, and network scanning
Rbot	3, 4, 10 and 11	IRC	DDoS
Virut	5 and 13	HTTP	DDoS, spam, and data theft
Menti	6	IRC	Data theft
Sogou	7	HTTP	Spam and data theft
Murlo	8	IRC	Network scanning
NSIS.ay	12	P2P	Data theft

Table 2. The analysis of Botnet traffic [1].

No.	Dataset Bot	#Packets	Remarks
CTU-MALWARE-CAPTURE-BOTNET-42	Neris	323154	"The botnet used an HTTP based C&C channel The bot sent spam and do some Click Fraud".
CTU-MALWARE-CAPTURE-BOTNET-43	Neris	176064	"The bot sent spam, connected to an HTTP CC, and use HTTP to do some Click Fraud".

- **Filtering phase:**

The filtering phase is used to extract the important features from the data file and using those features for analysis. Any Botnet have features like No, source IP, destination IP, Protocol, Timeline, Length, Source Port, Destination Port, TCP Segment Len, Frame Length, Header Length, Frame Number, Checksum, Header Checksum, Arrival Time, … etc. [25], In a proposed work a filtering phase begins to select attributes, the attributes are choices after many experiments. The best extracted attributes shown in Table 3.

Table 3. Attributes used in Neris botnet detection.

Feature name	Feature contents
Length	Size of packet include (header, trailer and data send).
Snapshot length	Data size for frame captured from network capturing and stored it in the capture file.
Last packet elapse Packet	Capture time and duration of the last packet protocol packet Numbers.
Time span/s	Is the duration period between the first and last packet?
Average App	Explain information for wire shark hardware.
Average size	Header packet average size.
Bytes	The protocol bytes No. take from total captured packets.
Average Byte/s	The average No. of protocol bytes from the total capture packets.
Average Bits/s	The average bandwidth of this protocol in relation to the capture time.

- **Detection phase**:

  In this phase the attributes captured from network traffic are used by SMO algorithm to detect Neris botnet. settings of control parameters are suggested as follows: SMO parameters setting are:

  - Size of Spider monkey (N) = 60
  - Number of Maximum group (MG) = 5
  - Global Leader Limit (GLL) = 60,
  - Local Leader Limit (LLL) = 1500
  - $pr \in [0.1, 0.4]$,

  $pr_{G+1} = pr_G + (0.4 - 0.1) / MIR$ is used to increase over iterations, where G is the iteration counter and is MIR is a maximum number of iterations. The stopping criteria is: A maximum number of evaluations function (2.0× 105) is reached, the number of executions =100. The fitness function is calculated as in Eq. (4).

5. Experimental Result

To test the efficiency of the proposed method to detect Neris Botnet, an experiment was performed on in the data file contain normal and Botnet traffics, as shown in Table 4. After the filtering was completed, the traffics are now used to find whether they are Normal or Neris Botnet, using computers having Microsoft Windows 10, and the proposed work was programmed using Matlab R2015a.To measure the efficiency and performance of proposed work to detect Nersi Botnet is calculated depending on parameters: Correct consistency Nersi Botnet (True positive (TP)), incorrect consistency Nersi Botnet(True Negative(TN)), correct rejected Nersi Botnet(False Positive(FP), incorrect rejected Nersi Botnet(False Negative(FN)). The efficiency can calculate as in (6)

$$Efficiency = \left(\frac{TP + TN}{TP + TN + FN}\right) \times 100$$

Table 4. Neris and normal packet (Example).

Normal Packet # 1		**Neris botnet Packet #1**	
Length	1073 MB	**Length**	58 MB
Snapshot length	4096	**Snapshot length**	65535
1-Last packet elapse	00:38:50	**1-Last packet elapse**	04:47:48
Packets	3730515	**Packets**	323154
Time span/s	2330.900	**Time span/s**	17268.814
Average app	1600.5	**Average app**	18.7
Average packet size, B	583	**Average of packet size, B**	164
Bytes	1045083217	**Bytes**	53096018
Average Byte/s	4578K	**Average Byte/s**	3074
Average Bits/s	36M	**Average Bits/s**	24K

Table 5 represents the performance of detection rate for Neris Botnet using SMO, the result of detection rate for Botnet-42 data set 99.8% is better than

detection rate 97.7% for Botnet-43 despite of the total Botnets in Botnet-42 is greater than the size of total Botnets in Botnet-43 data set, this is due to the ability of SMO to distinguish big data, and SMO algorithm is a multi-level method, which is more adaptable for big data, and The SMO algorithm is a population based stochastic meta-heuristic, also SMO algorithm is well balanced for good exploration and exploitation most of the times for detection in big data, see Fig. 3.

Table 5. Efficiency result.

No.	Data set	Total botnets	Efficiency (%)	Executed time (Sec)
1	Botnet-42	323154	99.8	12505
2	Botnet-43	176064	97.7	9255
Detection Rate			98.85	

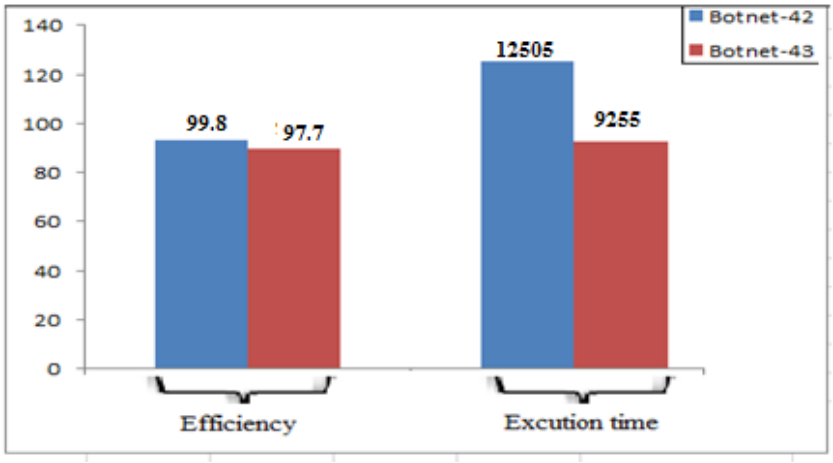

Fig. 3. Efficiency result.

The proposed work using SMO to be detected Neris Botnet is compared with other study [4], we find that the proposed method achieved a success detection rate, as shown in the Table 6 and Fig. 4.

Table 6. Efficiency detection rate of proposed work compared with [4].

No.	Data set	Detection method	detection rate%
1	CTU-13 (BOTNET-42, BOTNET-43)	Spider Monkey Optimization (SMO) Algorithm	98.85
2	CTU-13 (BOTNET-42, BOTNET 43)	Support Vector Machine (SVM) [4]	98.04
3	CTU-13 (BOTNET-42, BOTNET-43)	Naïve Bayes [4]	72.75
4	CTU-13 (BOTNET-42, BOTNET-43)	Decision Tree [4]	99.90
5	CTU-13 (BOTNET-42, BOTNET-43)	k-nearest Neighbour (KNN) [4]	98.19

Figure 4 illustrate the efficiency detection rate of compression among spider monkey algorithm and another algorithm.

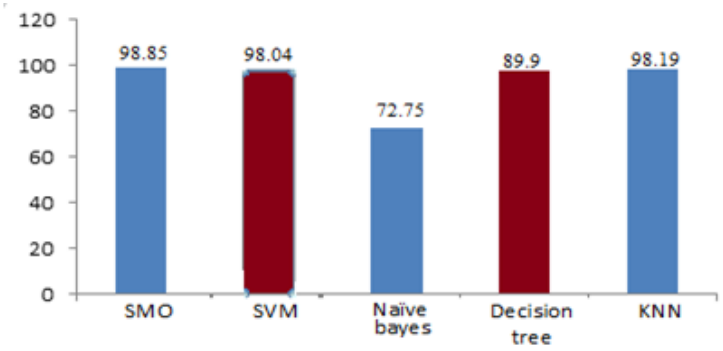

Fig. 4. Efficiency detection rate of proposed work compared with [4].

6. Conclusion

This paper proposes work to detect Neris Botnet using a Spider Monkey Optimization (SMO) algorithm with high accuracy detection rate. To know the accuracy of the proposed work a CTU-13 (Botnet-42 with total Botnets 323154 and Botnet-43 with total Botnets 176064) is used to obtain the accuracy of data set evaluation. From experimental result, the proposed work realizes high accuracy using SMO, the average efficiency detection result for Neris Botnet is 98.85%, where the detection rate for Botnet-42 is 99.8% and detection rate for Botnet-43 is 97.7% due to using SMO, it gives a higher detection with the consideration of the two parameters: detection speed and detection accuracy to measure the validity of (SMO) for Botnet detection.

In this paper, the proposed work to detect Neris Botnet showed high performance in speed and accuracy detection rate and it proved that SMO can be used to detect Botnet like Neris Botnet. For future work, we propose using other intelligent Swarm algorithms to detect anther types of Botnet viruses.

Nomenclatures

D	The number of problem variables to be optimized.
$fitness_{max}$	Maximum fitness in the group of the i^{th} *SM*..
GL_j	*GL* location and $j \in \{1, 2, ..., D\}$.
i^{th}	Represent the position of spider monkey (*SM*) in the population.
LL_{kj}	Represents the j^{th} dimension of the k^{th} local group leader position.
SM_i	A vector of D-dimensional vector.
SM_{minj} *and* SM_{maxj}	Are lower and upper bounds of SM_i in j^{th} direction respectively and $\varphi \in (0,1)$.

Greek Symbols

φ	$\varphi \in (0,1)$

Abbreviations

ACO	Ant Colony Optimization

AFS	Artificial Fish Swarm Algorithm
C&C	Command-and-Control
CNN	Convolutional Neural Networks
DDoS	Distributed Denial-of-Service Attacks
EHO	Elephant Herding Optimization
GL	Global Leader
GLD	Global Leader Decision
GLL	Global Leader Learning
IoT	Internet of Things
IRC	Internet Relay Chat
LL	Local Leader
LLD	Local Leader Decision
LLL	Local Leader Learning
P2P	Peer to Peer
PSO	Particles Swarm Optimizations
SI	Swarm Intelligence
SMO	Spider monkey optimization Algorithm

References

1. Bhale, K.M. (2016). *Botnet detection tools and techniques: A review*. A report submitted to Centre for Cyber Security Institute for Development and Research in Banking Technology,1-29.
2. Silva, S.S.C.; Silva, R.M.P.; Pinto, R.C.G., and Salles, R.M. (2013). Botnets: a survey. *Computer Networks*, 57(2), 378-403.
3. Thangapandiyan, M.; and Rubesh, A.P.M. (2018). Botnet detection techniques in cloud computing environment: a survey. *International Journal of Pure and Applied Mathematics*, 118(22), 929-939.
4. Seshadri, R.Ch.; Vinod, B.P.; and Srinivas P.N. (2018). Detecting bots inside a host using network behavior analysis. *International Journal of Computer Applications*, 180(47), 1-4.
5. Obeidat, A.A.; Al-Kofahi, M.M.; Bawaneh, M.H.; and Hanandeh, E.S. (2017). A novel botnet detection system for p2p networks. *Journal of Computer Science*, 13(8), 329-336.
6. Payam, V.A.; Timo, H.; Gil, D.; Mikhail, Z.; and Mahsa, M. (2016). Unsupervised network intrusion detection systems for zero-day fast- spreading attacks and botnets. *International Journal of Digital Content Technology and its Applications (JDCTA)*, 10(2), 1-13.
7. He, J.Y.Y.; Wang, X.; Tang, C.; and Zeng, Y. (2014). Peer digger: digging stealthy p2p hosts through traffic analysis in real-time. *17th International Conference on Computational Science and Engineering, China*.
8. Kuan, Ch.L; SihYang, Ch.; and Hung J.C. (2014). Botnet detection using support vector machines with artificial fish swarm algorithm. *Hindawi Publishing Corporation, Journal of Applied Mathematics*, 2014.
9. Mei, Ch.; and Gu-Hsin, L. (2018). Detection of C&C servers based on swarm intelligence approach. *Journal of Computers*, 29(5), 190-201.
10. Huseynov, K.; Kim, K.; and Yoo, P.D. (2014). Semi-supervised Botnet detection using ant colony clustering. *31st Symposium on Cryptography and Information Security Kagoshima, Japan, January 2014*, 21-24.

11. Asadi, M.; Jamali, M.A.J.; Parsa, S.; and Majidnezhad, V. (2020). Detecting botnet by using particle swarm optimization algorithm based on voting system. *Future Generation Computer Systems*, 107 , 95-111.
12. Habib, M.; Aljarah, I.; Faris, H.; and Seyedali, M. (2019). *Multi-objective particle swarm optimization for botnet detection in internet of things*. In book: Evolutionary Machine Learning Techniques, Springer, 203-229.
13. Zhang, Y.; Wang, Y.; and Qi, L. (2010), Botnet detection using support vector machines with artificial fish swarm algorithm. *Third International Workshop on Advanced Computational Intelligence*, Suzhou, Jiangsu, China, Aug. 2010, 25-27.
14. Pektaş, A.; Acarman, T.; Kuan-Cheng, L.; Sih-Yang, Ch.; and Jason, C.H. (2019). Deep learning to detect botnet via network flow summaries. *Neural Computing and Applications*, 31, 8021-8033.
15. Shi, W.-C., and Sun, H.-M. (2020). DeepBot: a time-based botnet detection with deep learning. Methodologies and Application. *Soft Computing*, 24, 16605-16616.
16. Homayoun, S.; Ahmadzadeh, M.; Hashemi, S.; Dehghantanha, A., and Khayami, R. (2018). *BoTShark: a deep learning approach for botnet traffic detection, project: botnet detection using advanced machine learning*. In book:Cyber Threat Intelligence. Springer International Publishing AG, part of Springer Nature 2018
17. Saiyan, S.; Youksamay, C.; Nunnapus, B.; Nattawat, K.; and Jirayus, C. (2016). Improving intrusion detection on snort, rules for botnet detection. *Convergence Security*, 1, 19-40.
18. Aswal. P.S.; and Bijalwan, A. (2016). Bitcoin mining-based botnet analysis. *International Journal of Computer Applications*, 145(6), 23-27.
19. Bansal, J.C.; Sharma, H.; Jadon, S.S.; and Clerc, M. (2014). Spider monkey optimization algorithm for numerical optimization. *Memetic Computing*, 6(1), 31-47.
20. Lenin, K.B.; Ravindhranath, R.; and Suryakalavathi, M. (2015), Modified monkey optimization algorithm for solving optimal reactive power dispatch problem. *Indonesian Journal of Electrical Engineering and Informatics (IJEEI)*, 3(2), 55- 62.
21. Sandeep, K.R.K.; and Vivek, K.SH. (2013). Modified position update in spider monkey optimization algorithm. *International Journal of Emerging Technology in Computational Applied Sciences (IJETCAS)*, 198-204, 2279-0055.
22. Kumar, S.; Kumari, R.; and Sharma, V. K. (2015). Fitness based position update in spider monkey optimization algorithm. *Procedia Computer Science*, 62, 442-449.
23. Kavita, G.; and Kusum, D. (2016). Tournament selection-based probability scheme in spider monkey optimization algorithm, *Advances in Intelligent Systems and Compute*r, 382, 239-250.
24. Kavita, G.; Kusum, D.; and Bansal, J.C. (2017). Improving the local search ability of spider monkey optimization algorithm using quadratic approximation for unconstrained optimization. *Computational Intelligence*,33(2), 210-240.
25. Samson, F; and Vaidehi, V. (2017). Hybrid botnet detection using ensemble approach. *Journal of Theoretical and Applied Information Technology*, 95 (8), 1646-1654.