Indian Journal of Science and Technology, *Vol 9(44), DOI: 10.17485/ijst/2016/v9i44/89812, November 2016*

ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Application of Artificial Intelligence to Software Defined Networking: A Survey

Majd Latah* and Levent Toker

Department of Computer Engineering, Ege University, Bornova, 35100, İzmir, Turkey; latahmajd@gmail.com, toker1960@gmail.com

Abstract

Background/Objectives: This paper surveys the application of Artificial Intelligence (AI) to the Software Defined Networking (SDN) paradigm which is a part of previous efforts to give the computer networks the ability of being programmed based on the separation between the control and forwarding planes. In SDN approach, the controller represents the central brain of the network which leads to an advanced level of flexibility and network intelligence. **Methods/Statistical Analysis:** Different artificial intelligence-based techniques have been applied to achieve an enhanced load balance, network security and intelligent network applications in the SDN approach. **Findings:** Ant colony algorithms were successful in increasing the maximal Quality of Experience (QoE) by 24.1% compared with the shortest path routing approach. Neural network based intrusion prevention system has shown a scalable performance with low false positive rate. Applying reinforcement learning based technique in adaptive video streaming system compared with the shortest path routing and greedy-based approaches has shown decreasing of the frame loss rate by 89% and 70% respectively. **Applications/Improvements:** This study highlights the first attempts for applying artificial intelligence in SDN paradigm. However, hybrid intelligent techniques could be the key for achieving more advanced behaviour in SDN-based networks.

Keywords: Artificial Intelligence (AI), OpenFlow, Software Defined Networking (SDN)

1. Introduction

In the last years, the conventional internet traffic has been changed to be more complex, especially in the era of big data, today's datacenters require more flexibility and scalability. Beside of the raising of advanced network applications and the existence of various types of devices even in the same area, thousands of end point devices could share and exchange different patterns of network traffic. That means that the current network infrastructure cannot meet all these needs and a new approach need to be proposed. Furthermore because there is no central control mechanism in the network, the configuration of network devices is not consistent and consumes a lot of time[1–3]. Also the traditional networks deal with distributed management and processing of network decision making. Therefore, we are dealing with a huge number of network nodes and that could be very expensive in some cases like Virtual Machine (VM) migration[4]. Furthermore, adding Quality of Service (QoS), Quality of Experience (QoE) and security policies to every end point in complex network architectures shows another disadvantage in the legacy network approach[4].

2. Software Defined Networking

The efforts to give the computer networks advanced level of programmability can be divided into three stages[5,6]:

- Active networks: (from the mid 1990s to the early 2000s) leading to add programmable functions in the network. Where the switching devices can perform operations in order to process the packets.
- Control and data plane separation (from 2001 to 2007) leading to a new skills like predicting or controlling routing behaviour.

**Author for correspondence*

- The OpenFlow API (from 2007 to 2010) which was the first widespread southbound interface between control and data plane that handles L2-L4 network flows. However, in order to handle L5-L7 flows for giving the ability to support Network Virtualization Function (NVF), OpenFlow protocol has to be extended[7]. Each flow table consists of: (i) header fields, (ii) counters and (iii) actions. The action will be applied when a packet matching occurs and the counters will be updated. If no packet matching occurs then a PACKET-IN message is sent to the controller over a secure channel which in turn encrypted using Transport Layer Security (TLS) to notify the controller about that packet[8,9] as shown in Figure 1.

OF 1.0[11] which is the first OpenFlow version was released with the supporting of one flow table in March 2008. The next versions of OpenFlow such as OF 1.1[12] support more advanced features like several flow tables which includes the using of "goto" instruction as a pointer referring to another flow table. OF 1.2[13] includes Ipv6, extensible matching using TLV structure and gives the switches the ability to communicate concurrently with multiple controllers. In OF 1.3[14] meter tables added for QoS support. Later in OF 1.4[15] TLV structures were used more to support optical ports on the switches and optimized operation to save the time spent to communicate between switch and controller in the case of the flow table is full[9].

Decoupling the network system provides the ability to manage it through a high level of abstraction. SDN, which is the recent paradigm to programmable networks, facilitates the network operations such as routing or adding rules to the forwarding devices by one central controller. That means the forwarding entities will implement the decisions given by this controller. A comparison between SDN architecture and conventional network architecture shown in Figure 2. The main abstraction concepts defined by SDN are[6]: 1. Forwarding, 2. Distribution, and 3. Specification. The forwarding abstraction allows the achieving of any forwarding actions by the controller while it hides the low-level handling with switching devices. The distribution abstraction includes the replacing of the traditional distributed control plane by a logically centralized one. The specification abstraction allows the developers to write network applications by describing the desired flow actions and configurations without handling with low-level or physical configurations.

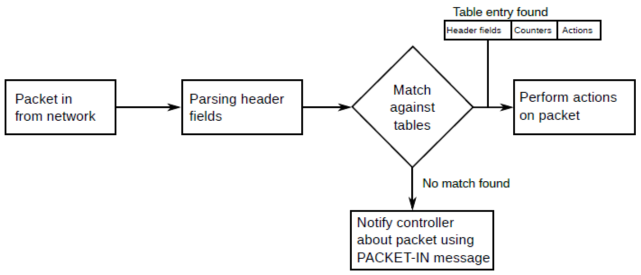


Figure 1. Basic packet forwarding in Open vSwitch[10].

The logical centralized control plane as shown in Figure 2 provides a global view of the network which opens the door for more optimized control of the forwarding elements. It could be achieved by single or distributed controller(s)[10]. Furthermore, FlowVisor[16] which is a proxy controller, provides a logical decentralization for network virtualization purpose. A brief comparison of controller platforms shown in Table 1.

The common SDN simulators and emulators such as Mininet, NS-3 and Estinet are described and compared in Table 2.

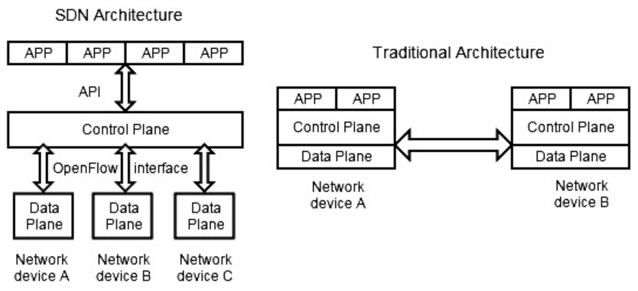


Figure 2. SDN vs traditional network architecture.

Table 1. A comparsion of controller platforms[9]

Controller	Language	Created by	OpenFlow version
NOX[17]	Python, C++	Nicira	1.0, 1.3
POX[18]	Python (2.7)	Nicira	1.0
Beacon[19]	Java	Stanford university	1.0.1
Maestro[20]	Java	Rice university	1.0
Floodlight[21]	Java	Big Switch Networks	1.0
Floodlight-plus[22]	Java	Big Switch Networks	1.3
Ryu[23]	Python	NTT Labs	1.0 to 1.4
(ODL)OpenDaylight[24]	Java	Linux Foundation	1.0, 1.3

Table 2. Comparision of SDN simulators and emulators[9]

Simulator/ emulator	Open source	Language	Platform	OpenFlow version
Mininet (Emulator)	Yes	Python	BSD open source	OF 1.3 of the reference user switch and NOX from CPqD and Ericsson
NS-3 (Simulator)	Yes	C++, Python	GNU GPLv2	Pre OF 1.0 and version of OF-SID that support MPLS
EstiNet (emulator/ simulator)	No	–	–	OF 1.3 and 1.0

The ability to program the SDN's forwarding plane by the SDN controller can save the cost required for adding specialized networking devices such as firewalls, load balancers and Intrusion Detection Systems (IDS)[25]. Furthermore, SDN approach provides more dynamic and less expensive solutions for technologies used in WANs such as path computation technology, extending these solutions as SDN application allows the network operators to program the Path Computation Element (PCE) directly[26]. Consequently, the controller achieves an adaptive behavior in different environments such as being a virtual machine manager in a private cloud[27].

3. Artificial Intelligence in SDN

Recently, the soft computing and artificial intelligence methods have started to play a significant role in most of modern systems such as intelligent transportation[28]. That gives us the chance to improve the performance of the current computer networks. The integration between the abstraction concept in SDN paradigm and AI techniques can lead to more adaptive behavior of network elements. Also it will introduce new mechanisms for dealing with both traditional network issues and SDN related new ones. In this section the recent efforts in this context will be discussed.

3.1 Load Balance and Flow Routing

Load balance function is a requirement for minimizing the latency and maximizing the throughput in computer networks that support multiple routing approaches. Load balancing also considered as a defense technique against some types of networks attacks such as DDoS attack[29,30].

The abstraction in SDN approach provides an important advantage which is the global view and discovering of topology of the network. In[31] a Back Propagation Neural Network (BPNN) used for achieving real time dynamic load balance and latency has been decreased by 19.3% compared to DLB and static Round Robin methods. The input vector for the neural network contains path information which are: 1. Bandwidth utilization ratio 2. Packet loss rate 3. Transmission latency and 4. Transmission hops[31]. Authors in[32] also proposed a BPNN based approach for load balancing in data centers. The BPNN applied internally inside the Open vSwitch in a way that reduce the time consumed for sending routing decision from the controller to the Open vSwitch. The input vector consists of: 1. Available bandwidth and 2. Packet loss. And from[33], which proposes a genetic algorithm in SDN based client server architecture. The fitness function defined by the Formula (1):

$$\mathrm{Min}\ \frac{\sqrt{\dfrac{\left(\sum_{j=1}^{k} X[j]^2\right) - \left(\dfrac{\sum_{j=1}^{K} X[j]}{K}\right)^2}{K}}}{\dfrac{\sum_{j=1}^{k} X[j]}{K}} \qquad (1)$$

Where K represents the servers and each one has X set of workload. The performance in[33] has been compared with random and round robin methods and shows better performance. In[34] also a genetic algorithm for flow routing optimization in SDN based audio over IP network has been introduced. The network described as a connected graph. The problem is to show that the graph meet the demand which is bandwidth and latency requirements of the source and destination. The fitness function given by the formula shown in Equation (2).

$$\mathrm{Max}\ \frac{\sum \textbf{embedded demands}}{\sum \textbf{demands}} \qquad (2)$$

Due to time consumption issues the authors did not implement the crossover operation. The population size and non allocation probability were the most important parameters to the algorithm. Also because the genetic algorithm has been implemented in python, the time efficiency was 10x time less than a mixed-integer linear programming algorithm implemented in C++. The advantage to use the genetic algorithm approach is to get a partial solution of the problem through the solving stage while this is not possible in linear program; this partial solution helps in evaluating other algorithms. In another context, an Ant Colony Optimization (ACO) approach for QoE-aware flow routing is presented in[35]. ACO is a swarm intelligence method that uses metaheuristic optimization[36]. In computer networks the Quality of Experience (QoE) indicates requirements for the customers to measure the value of provided service from customer's perspective. In[35] the SDN applications deliver user session parameters to the controller which in turn runs the ACO algorithm on a weighted graph, where the weights between vertices are delay and loss rate for each network device. The fitness function depends on the flow type and estimated value of corresponds QoE model (i.e.,: audio, video or data). ACO has achieved 24.1% increasing for the maximal QoE value obtained by the shortest path routing approach.

3.2 Network Security

The SDN approach introduces a set of new security challenges and it seems one of the biggest issues in SDNs. The potential threats include targeting the controller by programming vulnerabilities, error configurations and DDoS attacks on the secure channel[37] as shown in Figure 3.

In addition, SDN has advantages to traditional networks from the security perspective as shown in Table 3.

Artificial intelligence and data mining techniques which have been used before for solving routing problems and optimizing the performance of the packet filters in conventional networks architectures[40–42] seem to play a significant role in SDN based networks after adding the

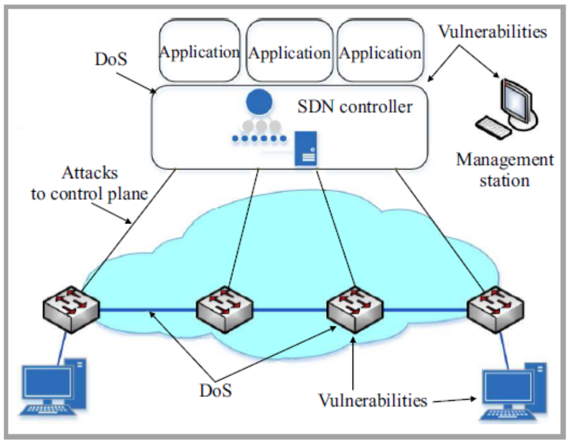


Figure 3. Potential vector of attacks in SDN[38].

Table 3. SDN security advantages[39]

SDN Characteristic	Security Use
Global Network View	– Network-Wide Intrusion Detection – Detection of Switch's Malicious Behavior. – Network Forensics.
Self-Healing Mechanisms	– Reactive Packet Dropping. – Reactive Packet Redirection.
Increased Control Capabilities	– Access Control.

programming ability as authors in[43] proposed an information security management system based on combination of fuzzy inference system and both of TRW-CB[44] and Rate Limiting[45] algorithms in SDN environment.

The TRW-CB algorithm which detects the SYN Flooding, caused by a host based on the idea that the benign host will obtain a higher successful connection probability than a malicious one[46]. The input for the fuzzy logic module obtained by the mentioned algorithms and the degree of attack obtained as output. The decision making system is implemented as application for the SDN controller with short-term learning module as shown in Figure 4. The proposed system has shown improved results compared with a non-fuzzy logic approach.

By taking the advantage of the global view in the SDN paradigm a BPNN based collaborative intrusion prevention system implemented in[47]. Each Open vSwitch is responsible for collecting data to perform inputs for several ANNs. The system is trained offline by MATLAB. Because it is a collaborative system, the Open vSwitches need to communicate with each other. Unfortunately in SDN paradigm, Open vSwitches cannot talk to each other. Therefore a neural forwarding table in each Open Vswitch has been realized and the controller also can help in building these tables. Figure 5 shows the template of a neural message and the experimental results show that as the network grows the detection rate of DDoS attack increases and the false positive rate decreases.

Whereas in[48] a Self-Organized Maps (SOM) approach for detection DDOS attack has been proposed. SOM is a variant of artificial neural networks based on unsupervised learning. SOM can be used as a classification mechanism[49] when handling with unlabeled input vector. The training in SOM is based on a set of desired features from flow entries of the Open vSwitches. The detection loop consists of three stages: 1. Flow collection which requests flow entries from

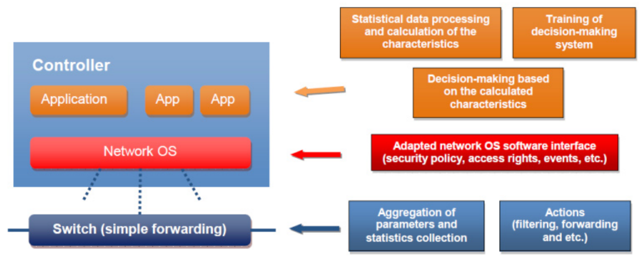
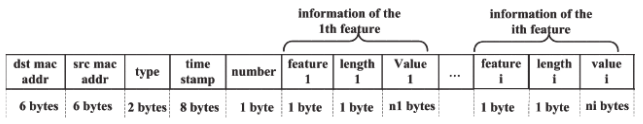

Controller
Application
App
App
Network OS
Switch (simple forwarding)
Statistical data processing and calculation of the characteristics
Training of decision-making system
Decision-making based on the calculated characteristics
Adapted network OS software interface (security policy, access rights, events, etc.)
Aggregation of parameters and statistics collection
Actions (filtering, forwarding and etc.)

Figure 4. Fuzzy logic based protected SDN architecture[43].



					information of the 1th feature				information of the ith feature		
dst mac addr	src mac addr	type	time stamp	number	feature 1	length 1	Value 1	...	feature i	length i	value i
6 bytes	6 bytes	2 bytes	8 bytes	1 byte	1 byte	1 byte	n1 bytes		1 byte	1 byte	ni bytes

Figure 5. Template of neural message transmitted between Open vSwitches[47].

all Open vSwitches. 2. Feature extraction which takes the output of the flow collection module and extracts the most important features that forms a potential DDoS attack. These features include: Average of Packets per flow (APf), Average of Bytes per flow (ABf), Average of Duration per flow (ADf), Percentage of Pair-flows (PPf), Growth of Single-flows (GSf), and Growth of Different Ports (GDP) and (iii) SOM classification which is used as classification method. These stages implemented as application level modules in SDN controller. The proposed approach compared by different methods conducted on the well-known KDD-99 data set has shown a lower overhead[48,50].

3.3 Intelligent Network Applications

The integration between SDN and AI field opens the door for building more intelligent network applications. As authors in[51] proposed a reinforcement learning approach for adaptive video streaming in SDN paradigm. The controller represents a periodically decision maker that determines the time of selecting a new path and when the server needs to change the quality of the video. Markov decision process used for modelling the actions of the decision making. The Q-learning technique used in the case of unknown rewards for moving between the current and next state. The percentage of packet losses and the number of quality changes represent the most significant parameters to define the reward. The Q-values are updated by the function shown in Equation (3) and stored in Q-table where γ and η represent the discount factor and learning factor respectively.

$$\hat{Q}(s_t,a_t) = \hat{Q}(s_t,a_t) + \eta(r_{t+1} + \gamma \max_{a_{t+1}} \hat{Q}\,(s_{t+1},a_{t+1}) - Q(s_t,a_t)) \tag{3}$$

And the softmax function shown in Equation (4) represents the probability of selecting an action in state *s* time *t*.

$$P(a|s) = \frac{\exp\left[\frac{Q(s,a)}{\mathbf{T}}\right]}{\Sigma_{b\epsilon A} \exp\left[\frac{Q(s,b)}{\mathbf{T}}\right]} \tag{4}$$

Where T represents a random move already used in simulated annealing method to escape from the local optima problem. The controller can change the current path and/or adaptively extract/add the selected layers based on the available bandwidth to increase the QoE of the video streaming service. The mentioned approach compared with the shortest path routing and greedy-based approaches has shown a decreasing of the frame loss rate by 89% and 70% respectively.

4. Conclusions

In this paper, we provide an overview of the integration between SDN paradigm and AI techniques. We described the basic SDN architecture and the significant role of OpenFlow protocol in it. Then we summarized the recent research contributions to provide more intelligent network behavior in the SDN approach. Neural networks have been applied in different scopes such as load balancing and network security. Employing AI techniques in SDN security aware systems showed a decreasing of the false positive detection rate. Also the results for the adaptive video streaming system have showed decreasing of the frame loss rate. As a result we see that the research in this area is growing rapidly and hybrid intelligent approaches can also bring more improvements to the field of SDN-based networks.

5. References

1. Open Networking Foundation. Software-defined networking: The new norm for networks. Available from: https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf
2. Astuto BN, Mendonca M, Nguyen XN, Obraczka K, Turletti T. A survey of software-defined networking: past, present, and

future of programmable networks. IEEE Communications Surveys and Tutorials. 2014; 16(3):1617–34.
3. Bakshi K. Considerations for Software Defined Networking (SDN): Approaches and use cases. IEEE Aerospace Conference, Big Sky; MT. 2013 Mar. p. 1–9.
4. Shinde MB, Tamhankar SG. Review: software defined networking and OpenFlow. International Journal of Scientific Research in Network Security and Communication. 2013Jun; 1(2):18–20.
5. Feamster N, Zegura E, Rexford J. The road to SDN: An intellectual history of programmable networks. ACM SIGCOMM Computer Communication Review archive. 2014; 44(2):87–98.
6. Kreutz D, Ramos FMV, Verissimo P, Rothenberg CE, Azodolmolky S, Uhlig S. Software-defined networking: A comprehensive survey. Proceedings of the IEEE. 2015; 103(1):14–76.
7. Basta A, Kellerer W, Hoffmann M, Hoffmann K, Schmidt E-D. A virtual SDN-enabled LTE EPC architecture: A case study for S-/P-gateways functions. Future Networks and Services (SDN4FNS); Trento. 2013 Nov. p. 1–7.
8. Jammal M, Singh T, Shami A, Asal R, Li Y. Software defined networking: State of the art and research challenges. Computer Networks. 2014; 72:74–98.
9. Rowshanrad S, Namvarasl S, Abdi V, Hajizadeh M, Keshtgary M. A survey on SDN, the future of networking. Journal of Advanced Computer Science and Technology. 2014; 3(2): 232–48.
10. Braun W, Menth M. Software-defined networking using OpenFlow: Protocols, applications and architectural design choices. Future Internet. 2014; 6(2):302–36.
11. OpenFlow Switch Specification Version 1.0.0. Available from: https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.0.0.pdf
12. OpenFlow Switch Specification Version 1.1.0. Available from: http://archive.openflow.org/documents/openflow-spec-v1.1.0.pdf
13. OpenFlow Switch Specification Version 1.2.0. Available from: https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.2.pdf
14. OpenFlow Switch Specification Version 1.3.0. Available from: https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.3.0.pdf
15. OpenFlow Switch Specification Version 1.4.0. Available from: https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.4.0.pdf
16. FlowVisor. Available from: https://github.com/OPENNETWORKINGLAB/flowvisor/wiki
17. NOX. Available from: http://archive.openflow.org/downloads/Workshop2009/OpenFlowWorkshop-MartinCasado.pdf
18. POX. Available from: https://en.wikipedia.org/wiki/Pox
19. Beacon. Available from: https://openflow.stanford.edu/display/Beacon/Home
20. Ng TSE, Cai Z, Cox AL. Maestro: A system for scalable OpenFlow control. Available from: http://www.cs.rice.edu/~eugeneng/papers/TR10-11.pdf
21. Floodlight OpenFlow Controller - Project Floodlight. Available from: http://www.projectfloodlight.org/floodlight/
22. Announcing release of Floodlight with OF 1.3 support. Available from: http://sdnhub.org/releases/floodlight-plus-openflow13-support/
23. Ryu 3.9 documentation. Available from: http://ryu.readthedocs.org/en/latest/getting_started.html#what-s-ryu.
24. Open day light. Available from: http://www.opendaylight.org/
25. Gorsansson P, Black C. Software defined networks - A comprehensive approach.1st ed. Morgan Kaufmann, an imprint of Elsevier; 2014.
26. Eadala SY, Nagarajan V. A review on deployment architectures of path computation element using software defined networking paradigm. Indian Journal of Science and Technology. 2016 Feb; 9(10). DOI: 10.17485/ijst/2016/v9i10/84944.
27. Mandekar AV, Chandramouli K. Centralization of network using openflow protocol. Indian Journal of Science and Technology. 2015 Jan; 8(S2). DOI: 10.17485/ijst/2015/v8iS2/61217.
28. Mittal P, Singh Y. Development of intelligent transportation system for improving average moving and waiting time with artificial intelligence. Indian Journal of Science and Technology. 2016 Jan; 9(3). DOI: 10.17485/ijst/2016/v9i3/84156.
29. Davis B. Leveraging the load balancer to fight DDoS. Available from: http://www.sans.org/reading-room/whitepapers/firewalls/leveraging-load-balancer-fight-ddos-33408
30. Califano A, Dincelli E, Goel S. Using features of cloud computing to defend smart grid against DDoS attacks. 10th Annual symposium on information assurance (Asia 15), ALBANY; 2015Jun. p. 44–50.
31. Chen-Xiao C, Ya-Bin X. Research on load balance method in SDN. International Journal of Grid and Distributed Computing. 2016; 9(1):25–36.
32. Ruelas AMR, Rothenberg CE. Implementation of neural switch using OpenFlow as load balancing method in data center. Campinas, Brasil: University of Campinas; 2015.
33. Chou L-D, Yang Y-T, Hong Y-M, Hu J-K, Jean B. A genetic-based load balancing algorithm in openflow network.

Advanced Technologies, Embedded and Multimedia for Human-centric Computing. 2013; 260:411–7.

34. Balaguer R. Flow embedding algorithms for software defined audio networks [Master thesis]. Zurich, Switzerland: Swiss Federal Institute of Technology. Available from: http://ftp.tik.ee.ethz.ch/pub/.../MA-2014-14.pdf
35. Dobrijevic O, Santl M, Matijasevic M. Ant colony optimization for QoE-centric flow routing in software-defined networks. 2015 11th International Conference on Network and Service Management (CNSM); Barcelona. 2015 Nov. p. 274–8.
36. Latah M. Solving multiple TSP problem by K-means and crossover based modified ACO algorithm. IJERT. 2016 Feb; 5(2):430–4.
37. Akhunzada A, Ahmed E, Gani A, Khan MK, Imran M, Guizani S. Securing the software defined networks: taxonomy, requirements, and open issues. IEEE Communications Magazine. 2015Apr; 53(4):36 –44.
38. Jankowski D, Amanowicz M. Intrusion detection in software defined networks with self-organized maps. Journal of Telecommunications and Information Technology. 2015; 4:3–9.
39. Dabbagh M, Hamdaoui B, Guizaniy M, Rayes A. Software-defined networking security: Pros and cons. IEEE Communications Magazine. 2015; 53(6):73–9.
40. Bai H. A survey on artificial intelligence for network routing problems. NM,USA: University of New Mexico; 2007.
41. Mustafa U, Masud MM, Trabelsi Z, Wood T, Al Harthi Z. Firewall performance optimization using data mining techniques. 2013 9th International Wireless Communications and Mobile Computing Conference (IWCMC); Sardinia. 2013 Jul. p. 934–40.
42. Mukherjee D, Acharyya S. Ant colony optimization technique applied in network routing problem. International Journal of Computer Applications. 2010; 1(15):66–73.
43. Dotcenko S, Vladyko A, Letenko I. A fuzzy logic-based information security management for software-defined networks. 16th International Conference on Advanced Communication Technology (ICACT); Pyeongchang. 2014 Feb. p. 167–71.
44. Mikians J, Barlet-Ros P, Sanjuas-Cuxart J, Sol´e-Pareta J. A practical approach to portscan detection in very high-speed links. PAM'11 Proceedings of the 12th International Conference on Passive and Active Measurement; Atlanta. 2011Mar. p. 112–21.
45. Williamson MM. Throttling viruses: Restricting propagation to defeat malicious mobile code. Proceedings 18th Annual Computer Security Applications Conference; Las Vegas. 2002 Dec. p. 61–8.
46. Mehdi SA, Khalid J, Khayam SA. Revisiting traffic anomaly detection using software defined networking. RAID'11 Proceedings of the 14th International Conference on Recent Advances in Intrusion Detection; California. 2011Sep. p. 161–80.
47. Chen X-F, Yu S-Z. CIPA: A collaborative intrusion prevention architecture for programmable network and SDN. Computers and Security. 2016; 58:1–19.
48. Braga R, Mota E, Passito A. Lightweight DDoS flooding attack detection using NOX/OpenFlow. 2010 IEEE 35th Conference on Local Computer Networks (LCN); Denver, CO. 2010 Oct. p. 408–15.
49. Deepa SN, Devi BA. A survey on artificial intelligence approaches for medical image classification. Indian Journal of Science and Technology. 2011 Nov; 4(11). DOI: 10.17485/ijst/2011/v4i11/30291.
50. Yan Q, Yu FR, Gong Q, Li J. Software-Defined Networking (SDN) and Distributed Denial of Service (DDoS) attacks in cloud computing environments: a survey, some research issues, and challenges. IEEE Communications Surveys and Tutorials. 2016; 18(1):602–22.
51. Uzakgider T, Cetinkaya C, Sayit M. Learning-based approach for layered adaptive video streaming over SDN. Computer Networks. 2015; 92(P2):357–68.