Data Inspection in SDN network

Soliman Abd Elmonsef Sarhan
Computer and Systems Engineering
Ain Shams University
Cairo, Egypt
Soliman.sarhan90@gmail.com

Mohamed A. Sobh
Computers and Systems Eng. Dept.
Faculty of Engineering
Ain Shams University
mohamed.sobh@eng.asu.edu.eg

Ayman M. Bahaa-Eldin
Misr International University
On leave from Ain Shams University
ayman.bahaa@eng.asu.edu.eg

Abstract - ***Software Defined Networks can be considered the most important development in Computer Networking in the last decade. Deep packet inspection (DPI) technology significantly enhances the security and management of current networks but combined with software-defined networking (SDN), DPI becomes an even more powerful tool can centralize network strategy control and quicken automation. The conversion from the traditional networks to the SDN network has a significant challenge that need a careful examination. We focus on Improved DPI can give the exhaustive data to notify the SDN controller about the situation of the network and its activity streams of traffic flows. This enables SDN to regard the network as a comprehensive asset as opposed to a different collection of devices. (e.g. switches, security and other Layer 4-7 elements). Ultimately, connecting SDN and DPI will let network pros apply policy control and automation to the whole network as opposed to individual components or elements. Leveraging a central DPI capability will give knowledge and intelligence to every important function (security, controller, policy, and so on.) -- instead of the current system of each functional box performing its own DPI. So, it became a must to inspect the data in SDN architecture that fit the benefits of central control.***

Keywords—SDN, DPI, Operator, Open flow, QOS, Control plan , Data plan, application plan

I. Introduction

A. The necessity for another new network architecture

The limit of the present Internet is getting to be inadequate to meet the massive volumes of traffic types conveyed by the modern services (e.g., cell phones and content, big data systems, server virtualization functions, cloud and accounting services), which is produced because of an expansive number of clients, sensors and applications [1,2].

Legacy networks constructed with numerous layers of static Ethernet switches orchestrated in a tree model are unsuitable for the dynamic processing and capacity of storage needs currently. Rather, new networking establishment are required to give aloft effectiveness, reliability, and power efficiency. Also, they ought to enhance the network speedup, scalability and submission a lot of digital services that ensures quality of service. The implementation of these demands is impossible with the currently available network hardware because of their limited capabilities.

What's more, to actualize policies on a large-scale network and providing new services or support existing services, administrators today need to set the configuration to a huge number of network equipment and protocols, its hard to achieve a reliable arrangement of security, QoS, and different policies. networks turn out to be more perplexing with the expansion of thousands of hardware that must be overseen and configured. Also, integration to network appliances is very hard because of The internals contrast from vendor to vendor.

B. Software Defined Networking

The main principle is separation where the control layer is decoupled from the data layer. SDN [3,4] is modern methodology for network programmability, which giving the capability to manage and control network behavior through programming. The SDN structure empowers centralized control of data path autonomously of the technique used to link this network equipment which can be sourced from various vendors. The central control device establishes all the information and keeps up a wide network perspective of the data path components and connections that link them.

This features of centralization and real time view makes the controller able to control and manage all network functions while enabling to make simple modifications at the networking functions.

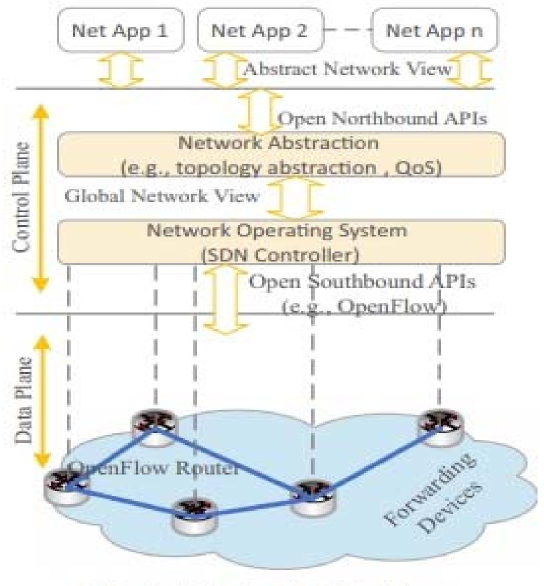


Fig. 1. Simplified view of an SDN architecture.

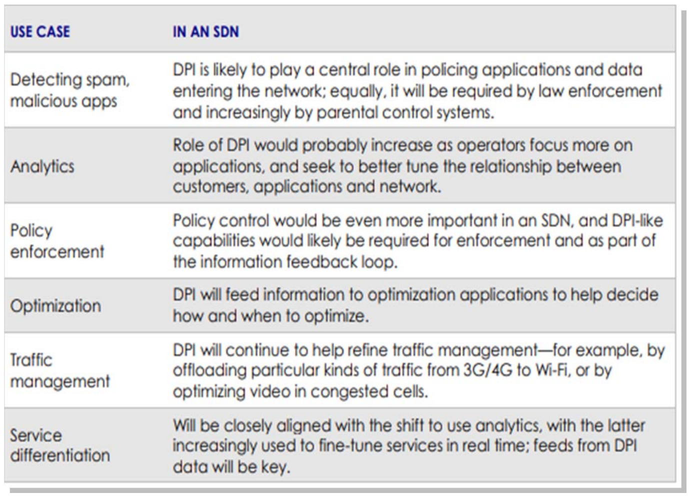
Fig: 1 describes the approach of SDN layers explaining the separation between diff layers applications layer, control layer and data layer. Applications layer perform their policies in the data layer by using the northbound API through the control layer without directly connecting to data layer. The control layer that connects to the data layer through the south-south APIs, where these APIs makes the SDN controller to communicate with network devices are in the data layer. These network devices should support standard APIs.

SDN gives possibility to control the whole network through a smart vision and saving resources that allows self-service provisioning, on-demand resource allocation. consequently, the legacy network can develop into an extensible service delivery system doesn't depend on vendor and fit for reacting quickly changing market needs, which incredibly streamlines design and operation at the network. Then, the network equipment never again needs to know and handle a huge number of protocols just simply receive and execute orders from the SDN control layer.

OpenFlow [5,6] consider the tangible implementation of the SDN structure. OpenFlow give the providers ability to redesign their traffic to trial a new protocol in currently networks without any negative effect at production environment. The main components of OpenFlow mechanism comprises of three sections: (i) a controller seted up on a remote host machine, (ii) flow tables created in OpenFlow switches, and (iii) an OpenFlow protocol make the controller able to create a secure connection to communicate with switches. The OpenFlow structure separating control logic from data forwarding attitude gives an adaptable ability to add and edit multiple forwarding roles in data layer.

C. *Role of DPI*

The SDN structure empowers centralized control of network, more flexible to implement management functions at network, simple adjustments to the networking operations but still need more info than layer 2-3 need progressively advanced view on users behavior to decide better strategy choices, and provide superior services, SDN needs to improve quality of service and DPI can do this role [7], [25], [26], [27], and [28] :

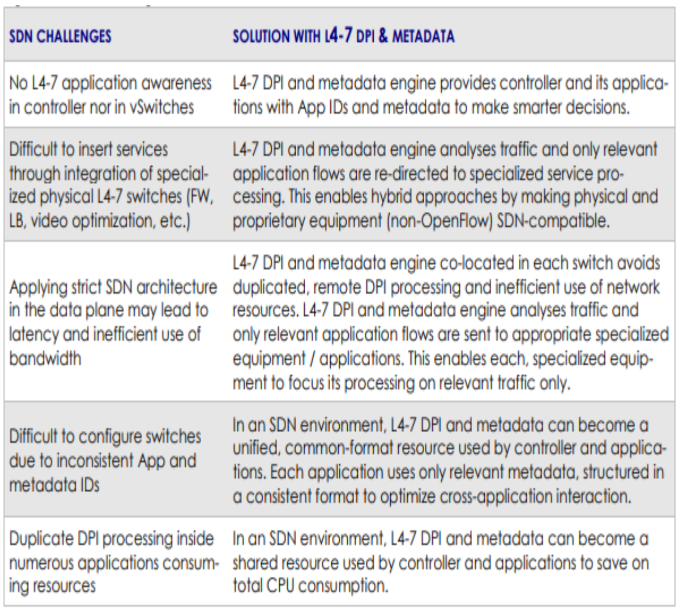


USE CASE	IN AN SDN
Detecting spam, malicious apps	DPI is likely to play a central role in policing applications and data entering the network; equally, it will be required by law enforcement and increasingly by parental control systems.
Analytics	Role of DPI would probably increase as operators focus more on applications, and seek to better tune the relationship between customers, applications and network.
Policy enforcement	Policy control would be even more important in an SDN, and DPI-like capabilities would likely be required for enforcement and as part of the information feedback loop.
Optimization	DPI will feed information to optimization applications to help decide how and when to optimize.
Traffic management	DPI will continue to help refine traffic management—for example, by offloading particular kinds of traffic from 3G/4G to Wi-Fi, or by optimizing video in congested cells.
Service differentiation	Will be closely aligned with the shift to use analytics, with the latter increasingly used to fine-tune services in real time; feeds from DPI data will be key.

Figure 2: DPI Use Cases in an SDN

- The needs to create feedback function where information on subscribers and network performance at real-time traffic, application utilization, subscribers behavior, appliance behavior, congestion status and others is forward to the SDN controller.

- The needs to know subscriber's utilization and behaviors (through the monitoring and investigation capacities are currently exist in most DPI programming) that to extend more fitting and customized benefit services to them.
- Networks needs to shield from infections, spam, DDoS attacks and harmful, illicit substance, DPI able to recognize dangers, Motives and used for organizations and parents content monitors systems.
- network administrators need to recognize and control high-affect applications for example streaming media traffic with a view to decrease or enhance their effects, mostly at congestion case, and enhance the Quality and rate of applications for example streaming media.
- Widespread deployment of the standard 3GPP Policy Control and Charging, additional to forwarding DPI data to policy servers to support decision-making; normally a similar equipment or programming additionally handles strategy implementation.



SDN CHALLENGES	SOLUTION WITH L4-7 DPI & METADATA
No L4-7 application awareness in controller nor in vSwitches	L4-7 DPI and metadata engine provides controller and its applications with App IDs and metadata to make smarter decisions.
Difficult to insert services through integration of specialized physical L4-7 switches (FW, LB, video optimization, etc.)	L4-7 DPI and metadata engine analyses traffic and only relevant application flows are re-directed to specialized service processing. This enables hybrid approaches by making physical and proprietary equipment (non-OpenFlow) SDN-compatible.
Applying strict SDN architecture in the data plane may lead to latency and inefficient use of bandwidth	L4-7 DPI and metadata engine co-located in each switch avoids duplicated, remote DPI processing and inefficient use of network resources. L4-7 DPI and metadata engine analyses traffic and only relevant application flows are sent to appropriate specialized equipment / applications. This enables each, specialized equipment to focus its processing on relevant traffic only.
Difficult to configure switches due to inconsistent App and metadata IDs	In an SDN environment, L4-7 DPI and metadata can become a unified, common-format resource used by controller and applications. Each application uses only relevant metadata, structured in a consistent format to optimize cross-application interaction.
Duplicate DPI processing inside numerous applications consuming resources	In an SDN environment, L4-7 DPI and metadata can become a shared resource used by controller and applications to save on total CPU consumption.

Figure 3: SDN Challenges & Solutions

II. DPI IN SDN ARCHITECTURE

DPI usually deployed in traditional network as an embedded it in different network devices, for example (firewalls, IDS, etc.). A main disadvantage of this methodology its very expensive because of publishing DPI software more than times on various equipment platforms. Furthermore, its hard to integrate between applications due to every vendor have a particular method for implementing DPI and arranging the outcomes. such as, system of one company classify a stream as twitter traffic while another system assigns it as social media traffic. DPI at SDN, can be work as a mutual function putted on main servers not as in legacy network as consider DPI apart of multiple network devices. This methodology brings down overall investment at DPI, where DPI can be installed at a small number of devices, comparison to DPI at legacy network, and in this manner power consumption is down. Furthermore, utilizing DPI make Integration between various functions and applications is less complex, for easy to perform a suitable format for Application Identifiers and its metadata.

A. Locating DPI in SDN-based Networks

As described before, SDN approach rely on three main layers applications, data and control layers. (Figure 4) illustrating these three layers where DPI could be placed for carrying on tasks like cyber security, user analytics, and traffic shaping. Sharing DPI information to the network can be applicable by deploying these scenarios [8]. Sharing of such information can save us huge energy and CPU costs as the classification of applications will be carried out once rather than multiple. Unified DPI facilitates control and management since all appliance would share "the same view" of the traffic.

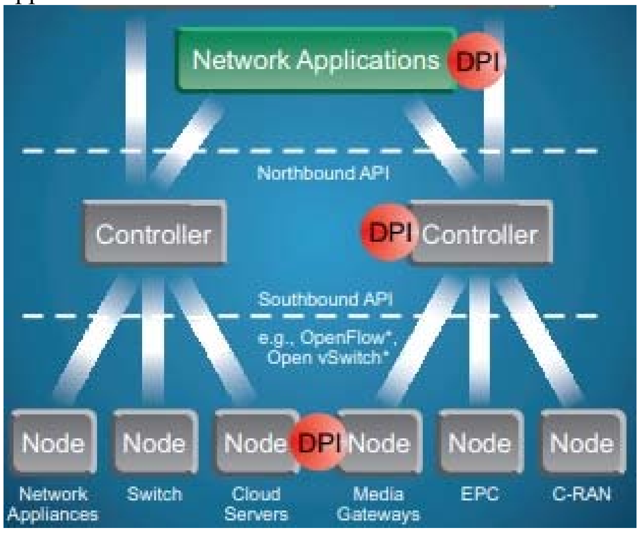


Figure 4: Dpi at different layers

The following describes the proposed scenarios for the DPI deployment:

1) Network applications layer

Application layer can be used to host the DPI as it can be embedded in the running applications. However, some cons can be happened in that scenario like the possible bottlenecks due to the long communication path as a portion of the traffic should be steered from the nodes to the running DPI engine placed in the application layer passing through the SDN controller. Then the rest of the traffic will be steered directly as the policy rules would be sent to the node by the application after the flow is being identified. So typically, not all of the traffic should be steered through the application layer. Because of time-delay This scenario should be deployed for non-time critical systems.

2) Controller layer

SDN controller also can deploy DPI software, which using the intelligence of network to control services or transmit it to applications layer through northbound API. The flow can be handled by the node (network appliances [switch, router, etc.) which send packets to the SDN controller for L4-L7 analysis, maybe using the Open Flow protocol. Some of pros it happened like keeps away from increasing in the cost of network nodes due to Finding DPI in the controller; nevertheless, parts of the traffic must be copied and transmitted from data layer to the controller, which could prompt some issues in performance and scalability. However, could minimize these concerns by using distributed controller structure.

3) Data layer

DPI software can be deployed at nodes in the data plan, Network nodes after identifying the Application Identifiers and its metadata, they can do one of two following:

• sending the results to the SDN controller or a network application, and after that get feedback rules or policies,

• applying a pre-configured policy directly. At the point when sending the data "results of DPI" to the SDN controller, it can configure the network node to apply a specific policy after having some instructions with a network application. A while later, all coming packets which have similar type not need DPI investigation. Deploying DPI in the data layer minimizes latency; Compared to the others scenario.

III. RELATED WORK

Algorithms of Aho-Corasick [10] and Wu-Manber [11] about exact multiple string matching used for DPI. Two basic solutions are using for regular expression matching, Nondeterministic Finite Automata (NFA) and Deterministic Finite Automata (DFA) [12, 13]. Proficient regular expression matching to date is an active search zone [14, 15, 13, 16, 17]. Accelerating DPI operations, on both two levels hardware [18, 19, 33] and software l [19, 13]. Most solutions to accelerate the DPI process at software level works on improving its underlying data structure.

As far as we know, no particular layout for accelerating DPI in a virtual or unified environment has been offered. QOSMOS [8] is consider the only exception is an industrial product that specializes in the classification of the protocol and does not deal with the general DPI issue. Furthermore, details of its implementation are not disclosed. And also Deep packet inspection as service [9] it's the nearest solution bot it focus on middle boxes that use dpi function, still costly because it implement a lot of dpi instance inside data plan, it split the work among instances running over different machines. In any case, the same notion was contemplated with regards to virtual IP-query, where data structures based on trie are considered [20, 21, 22]. and it's not viable to DPI because of the considerable variation in basic algorithms.

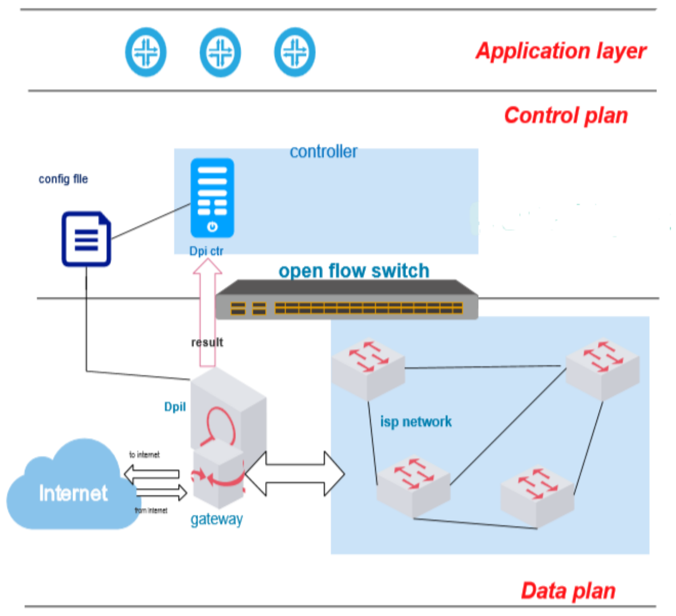


Figure 5: System Illustration.

IV. SYSTEM OVERVIEW

The underlying structure that supports DPI as a service will be detailed at this part. The basic design is to split dpi system to two main parts the first part DPI service works on scanning network traffic at data layer, classifying stream into known protocols type, filtering the data according to some policies and logs all detected patterns and send the scan result to the second main part Dpi controller to control and manage the policy chain. Its better if SDN environment with a Traffic Steering Application TSA (ex: SIMPLE [23]) to have more facilities that giving capability to connects policy chains to packets and forward the packets properly over the whole network. (see Figure 5).

The DPI Controller

DPI benefit versatility is essential since DPI is viewed as a bottleneck for all Operators to have whole vision. Thus, we will deploy DPI service instances at network gateway.

The DPI controller is a incorporated at control layer to be a central and able to do its role for manage the DPI operations over the network and By virtue of their location make it easy to speak with the SDN controller and providing with additional information to realize the suitable actions on nodes at data layer. Logically, the DPI controller established on top of the SDN controller (see Figure5). There are two main functions occur between the DPI instance and DPI Controller, first process dedicated to DPI controller is to write policies and rules for Dpi instance through configuration file. Correspondence between DPI Controller and dpi instance is done utilizing JSON messages sent through a direct connection to receive the result and logs from DPI instance.

The DPI controller also make all information about network available at SDN controller that's help TSA to set the policy chain make the decision to the right path forwarding the traffic generally, TSA inserts some MPLS or VLAN tags in the packet to effectively direct it across the network devices ([41]) ,Dpi controller can receives from the TSA too its related policy chains and execute it if possible It identify every policy chain with a unique ID that is utilized later to manage the traffic directly between middle boxes like firewalls (FUTURE WORK).DPI controller is additionally in charge of initializing dpi instances, and some advanced features that require wide view of the network

A. DPI Instances Deployment

DPI controller abstracts the DPI process for the SDN controller, Dpi instance and TSA, Subsequently, registering and deployment DPI instances are considered the most important tasks required from DPI controller, it should manage resources of the DPI instance There are a lot of considerations for deployment, but we illustrate just a little. In the first place, we stress all DPI instances should be at the gateway of our network to inspect all packets that "from and to network or subscriber". Then the DPI controller will instruct the SDN controller to set the right policy chain. Moreover, the DPI controller ought to deal with the resources of DPI instance, to know the instances status so it is not flooded by traffic or any bad effect, and in this way, performs ineffectively. Therefore, the DPI controller have to collect measurement of all active instances' performance, according to this metrics can take a decision to migrate flows between instances or allocate more instances

B. DPI INSTANCE IMPLEMENTATION

The main target to Dpi instance Examine traffic in real-time to diagnose issues and confirm policy enforcement, Leverage analytics to identify trends and opportunities, enabling new services and identifying application quality trends, according to that we use an open source Dpi "nDPI [24]" at Dpi instance.

Application protocol is known in nDPI by two identifiers one of them is protocol Id (unique number), and the other protocol name (symbolic name e.g. Skype). Applications utilizing nDPI will most likely utilize the protocol Id, while people the symbolic name. Normally, a traffic dissector written in C detect the protocol, but also it can be recognized in terms of protocol characteristic, protocol/port and IP address. For example, the Yahoo traffic is detected by both the dissector for LAN-based communications, and by labeling as Yahoo the HTTP traffic on which packets header field is set to (*.yahoo.com) as host to this service, and also nDPI can Handling Encrypted Traffic. The nDPI library includes the signature of more than 170 protocols, nDPI contains inside a configuration for a lot of famous protocols that are detected by using the mechanism mentioned before. Also, have the capability to add a configuration file at runtime, to increase the set of detected protocols so the new protocols can be characterized without any change the protocol dissector. but also, it can be further extended at runtime by utilize file of configuration.
Adding a new protocol dissector not the only way for nDPI users to define protocols but adding the file of configuration at run time also can be efficient.
(example of configuration file format.)

```
# Format:
# <tcp|udp>:<port>,<tcp|udp>:<port>,.....@<proto>
tcp:80,tcp:8080@HTTP
udp:5061,udp:5062@SIP
udp:3260, tcp:3260, udp:860, tcp:860@iSCSI
tcp:3000@ntop
# Subprotocols
# Format:
# host:"<value>",host:"<value>",.....@<subproto>
host:"google.com"@Google
host:"venere.com"@Venere
```

according to previous configuration sample, whenever nDPI sees UDP traffic on port 5061or port 5062 nDPI label it as SIP. Library of nDPI is supported with some of OpenDPI design, for that library code is utilized to execute general functions, and protocol analyzation is executed in plugins. All the library initialization is performed only once at startup, without a runtime fine when a new packet comes in and needs analysis. nDPI anticipates that the coming packets separated in streams, set of packets with the same (VLAN, IP/port source/destination),as another meaning same session. and packet has been decoded until layer three. This means nDPI start the function of decoding the packet from the IP layer up. nDPI comes with a basic test application called (pcapReader.c) that demonstrates how to perform packets classification and provides utility functions for efficient flow processing.

V. EXPERIMENTAL RESULTS

According to our Solution we solve the issue we mentioned it before at locating Dpi at SDN , locating Dpi instance at Data layer and separating Dpi controller avoid us of network latency and solve performance and scalability issues because of Duplications traffic if you locate Dpi at control or application layer additionally locating it at network gateway is costly less than locating more Dpi appliance across network at data layer.

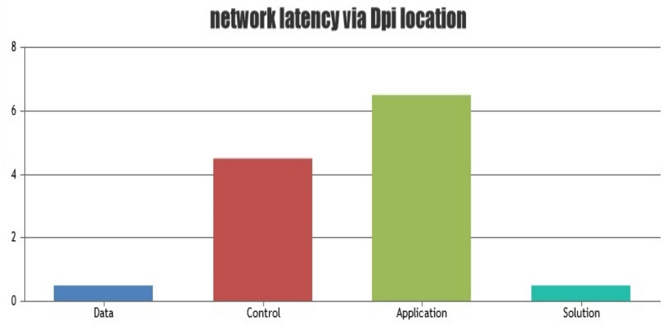


Figure 6: System Illustration

Network latency at case of Application layer is higher than controller layer and data layer is the less (figure 6) we neglect the processing latency because its a constant we assume we use the same Dpi at each layer.

VI. FUTURE WORK

1) Planned to integrate with radius traffic to make full mapping to subscriber at ISP operators.
2) making flow table to manage flow traffic between middle boxes (Firewall, IDS, IPS,…).

VII. CONCLUTION

Growth of traffic is creating a big challenge for network operators makes it very hard to manage and monitor, while SDN can facilitate bandwidth management by providing the capability for the operators to control network assets in a centralized way with a comprehensive view rather than a decentralized or local view. However only centralization is not enough, SDN operators still need more information and also the capabilities to identify and analyze the traffic and applications flowing on their networks in real time, which represents another big challenge facing network operators.
DPI have the ability to identify types of traffic in the network at real-time, and also able to associate that information at an incrementally granular level with another related data like (subscriber, location, etc). This paper focuses on Deep Packet Inspection and locating DPI at SDN layers. The main idea is to split dpi service into two main parts. The first part is the DPI service located at network gateway which is responsible for scanning the traffic and forward results to the second main part:

the DPI controller with the task of managing the policy chain, helping the operators to have more control over networks that provide a large number of applications and services and providing improvements, speed and flexibility in the network.
Finally, the proposed SDN implementation for the framework and experiments is meant to demonstrate that these ideas are transformed into performance enhancements and substantially more flexible and scalable layout at real time.

VIII. REFERENCES

[1] A. Metzger, C.C. Marquezan, Future internet apps: the next wave of adaptive service-oriented systems, in: ServiceWave, 2011, pp. 230–241.

[2] G. Ortiz, J. Cubo, Adaptive Web Services for Modular and Reusable Software Development: Tactics and Solutions. IGI Global, 2013. 1–415. Web. 6 Nov. 2014. http://dx.doi.org/10.4018/978-1-4666- 2089-6.

[3] N. McKeown, Software-defined networking, INFOCOM keynote talk, April 2009, Rio de Janeiro, Brazil.

[4] H. Kim, N. Feamster, Improving network management with software defined networking, Communications Magazine, vol. 51 (2), IEEE, 2013, pp. 114–119.

[5] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, Openflow: enabling innovation in campus networks, ACM SIGCOMM Comput. Commun. Rev. (2008) 69–74.

[6] ONF, The openflow 1.3.1 specification, Tech. rep. September 6, 2012

[7] Qosmos Co-publishes White Paper with Heavy Reading on "The Role of DPI In An SDN World"... Paris, France – December 13, 2012. Network Intelligence.

[8] Intel-Qosmos White Paper: Service-Aware Network Architecture Based on SDN, NFV and Network Intelligence

By qos-redacteur | Published 02/09/2013

[9] Deep packet inspection as a service

Anat Bremler-Barr, Yotam Harchol, David Hay, Yaron Koral

, Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies, 2014.

[10] Alfred V. Aho and Margaret J. Corasick. Efficient string matching: An aid to bibliographic search. Commun. of the ACM, 18(6):333–340, 1975.

[11] Sun Wu and Udi Manber. A fast algorithm for multi-pattern searching. Technical report, Chung-Cheng University, University of Arizona, 1994.

[12] Michela Becchi and Patrick Crowley. A hybrid finite automaton for practical deep packet inspection. In CoNEXT, page 1, 2007

[13] Sailesh Kumar, Sarang Dharmapurikar, Fang Yu, Patrick Crowley, and Jonathan Turner. Algorithms to accelerate multiple regular expressions matching for deep packet inspection. In SIGCOMM, pages 339–350, 2006.

[14] Michela Becchi and Patrick Crowley. An improved algorithm to accelerate regular expression evaluation. In ANCS, pages 145–154, 2007.

[15] Domenico Ficara, Stefano Giordano, Gregorio Procissi, Fabio Vitucci, Gianni Antichi, and Andrea Di Pietro. An improved DFA for fast regular expression matching. Computer Communication Review, 38(5):29–40, 2008.

[16] Sailesh Kumar, Jonathan Turner, and John Williams. Advanced algorithms for fast and scalable deep packet inspection. In ANCS, pages 81–92, 2006.

[17] Fang Yu, Zhifeng Chen, Yanlei Diao, T. V. Lakshman, and Randy H. Katz. Fast and memory-efficient regular expression matching for deep packet inspection. In ANCS, pages 93–102, 2006

[18] Zachary K. Baker and Viktor K. Prasanna. Time and area efficient pattern matching on FPGAs. In FPGA, pages 223–232, 2004.

[19] Sarang Dharmapurikar, John Lockwood, and Member Ieee. Fast and scalable pattern matching for network intrusion detection systems. IEEE Journal on Selected Areas in Communications, 24:2006, 2006.

[20] Jing Fu and Jennifer Rexford. Efficient IP-address lookup with a shared forwarding table for multiple virtual routers. In CoNEXT, page 21, 2008.

[21] Hoang Le, Thilan Ganegedara, and Viktor K Prasanna. Memory-efficient and scalable virtual routers using FPGA. In FPGA , pages 257–266, 2011

[22] Haoyu Song, Murali Kodialam, Fang Hao, and TV Lakshman. Building scalable virtual routers with trie braiding. In INFOCOM, pages 1–9, 2010.

[23] Zafar Ayyub Qazi, Cheng-Chun Tu, Luis Chiang, Rui Miao, Vyas Sekar, and Minlan Yu. SIMPLE-fying middlebox policy enforcement using SDN. In SIGCOMM, pages 27–38, 2013.

[24] nDPI: Open-source high-speed deep packet inspection, 2014 IEEE,International Wireless Communications and Mobile Computing Conference (IWCMC).

[25] Ebada Essam-Eldin ElDessouky, HasanDAĞ, Ayman M. Bahaa-Eldin, "Protecting Openflow Switches against Denial of Service Attacks" ,2017 12th International Conference on Computer Engineering and Systems (ICCES), 479-484, 2018, IEEE

[26] Mohammad Mousa, Mohmed Sobh, Ayman M. Bahaa-Eldin, "Software Defined Networking concepts and challenges", 2016 11th International Conference on Computer Engineering & Systems (ICCES), 79-90, 2016, IEEE

[27] Mohammad Mousa, Mohmed Sobh, Ayman M. Bahaa-Eldin, "Autonomic Management of MPLS Backbone Networks using SDNs", 2017 12th International Conference on Computer Engineering and Systems (ICCES), 169-174, 2018, IEEE

[28] Reham ElMaghraby, Nada Abd Elazim, Ayman M. Bahaa-Eldin, "A Survey on Deep Packet Inspection", 2017 12th International Conference on Computer Engineering and Systems (ICCES), 188-197, 2018, IEEE