Scientific Journal of Impact Factor (SJIF): 4.72

e-ISSN (O): 2348-4470
p-ISSN (P): 2348-6406

International Journal of Advance Engineering and Research Development

Volume 4, Issue 4, April -2017

Software Defined Network: A Survey

Nikita Bhalani[1], Nitul Dutta[2]

[1,2]*Computer department, Marwadi Education Foundation's Group of Institutions, Rajkot, India*

Abstract—*Software Defined Network is used to simplify network management by separating network control logic from its underlying hardware by enabling network administrators to exert more control over functionality of network and providing a unified global view of network. In this paper, our aim is to describe SDN related technologies, SDN architecture, openflow architecture, various simulators used to implement SDN concept and various SDN related researches that defined in which areas we can apply SDN concept.*

Keywords—*Software-Defined Networking, OpenFlow, Control and Data Plane, Distributed Mobility Management and SDN Simulators*

I. INTRODUCTION

The traditional network architectures which are *hardware centric network* are not suitable for dynamic computing and storage. Need to reexamine traditional network architectures due to the explosion of mobile devices and content, changing the traffic pattern, the "consumerization of IT", server virtualization, and the rise of cloud services. The complexity in existing network architecture leads to a few limitations, including inconsistent policies, inability to scale, and vendor dependence [1]. First of all, the traditional network based on packet accessibility. To fulfill the requirements of scalability, reliability, security and QoS from various applications, need to design different network protocols and developed independently to solve individual application problems.

Nick McKeown from Stanford University, proposed software defined network (SDN) [2].The Concept of Software Defined Network is emerging network paradigm proposed to overcome limitation of existing traditional network infrastructure. Complex design of traditional network is due to tight coupling of Data Plane and Control Plane in term of network switches and routers [3].

Software Defined Network is "*an emerging network architecture where the network control is decoupled and separated from the forwarding mechanism and is directly programmable*" [4]. The control plane and the data plane can be separated by means of a well-defined application programming interface (API) between the switches and the SDN controller. The mostly used API is OpenFlow [5], [6]. OpenFlow is the first standard communications interface that allows direct access to and manipulation of the forwarding plane of network devices such as switches and routers, both physical and virtual. The architectural principles of SDN are: (1) Separation of network control and forwarding functions. (2) Programmable interfaces for the network (at multiple levels). (3)Logically Centralized network control element. (Controller).Two Characteristics of SDN Implementation are: 1) Network controllers are independent software elements. 2) System elements use open, standardized interfaces.

This paper presents a review of Software defined Networks. Section II shows SDN overview with SDN architecture and compare SDN with Traditional Network scheme. In section III, we present the generalized architecture of Openflow protocol used to manage SDN southband interface. Section IV provides literature review of SDN. Various Simulators used for SDN is described in section V. Section VI briefly mentions Open research areas of SDN, followed by conclusion.

II. SDN OVERVIEW

A. *SDN architecture requirements and scope*

The SDN Architecture provides the following requirements [7](1) Independence from the characteristics of SDN controller distribution.(2) Policy and security boundaries related to information sharing and trust.(3) Based upon open SDN controller plane interfaces provide interoperability.(4) Support for management interfaces, across which resources and policy may be established, as well as other more traditional management functions.(5) Scalability and support for recursion to encompass all feasible SDN controller architectures.(6) Co-existence with existing business and operations support systems, and other administrative or control technology domains.(7) Scalability and support for recursion to encompass all feasible SDN controller architectures.

B. *SDN architecture Principles*

Figure 1 shows there are three SDN architecture principles (1) Decoupling of control and data plane (2) Logically centralized control (3) Exposure of abstract network resources and state to external applications.

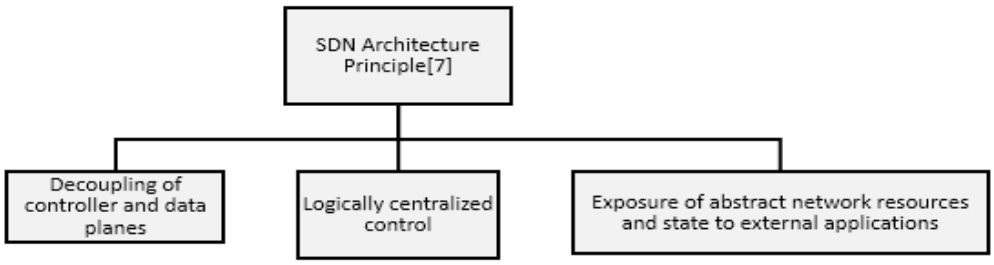

Fig. 1. Principles of SDN Architecture

1) Decoupling of controller and data planes

This principle calls for separable controller and data planes. However, it is understood that control must necessarily be exercised within data plane systems. The D-CPI between SDN controller and network element is defined in such a way that the SDN controller can delegate significant functionality to the NE, while remaining aware of NE state.

2) Logically centralized control

In comparison to local control, a centralized controller has a broader perspective of the resources under its control, and can potentially make better decisions about how to deploy them. Scalability is improved both by decoupling and centralizing control, allowing for increasingly global but less detailed views of network resources.

3) Exposure of abstract network resources and state to external applications

Applications may exist at any level of abstraction or granularity, attributes often described as differing latitudes, with the idea that further north suggests a greater degree of abstraction. Because an interface that exposes resources and state can be considered a controller interface, the distinction between application and control is not precise. The same functional interface may be viewed in different lights by different stakeholders. Just like controllers, applications may relate to other applications as peers, or as clients and servers.

C. Comparison of Traditional Network and SDN

As shown in Figure 2 in traditional network all control and data functionality are combined. Legacy infrastructure as Hardware centric network have Manageability, Flexibility, Extensibility issue due to static architecture not suitable for dynamic computing and storage for that need to developed SDN[1].So in SDN divide functionality of network into two separate plane as Data Plane and Control plane.

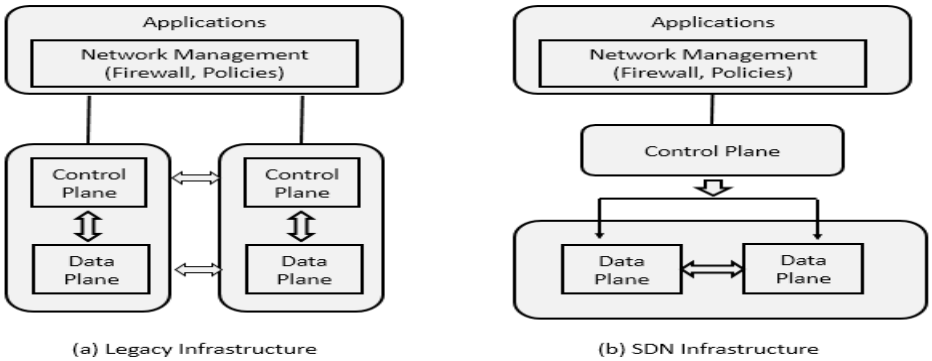

Fig. 2. Comparison between (a) the Lagacy Infrastructure and (b) SDN Infrastructure[8].

D. SDN Architecture

Open networking foundation (ONF) is a user-driven organization dedicated to the promotion and adoption of SDN through open standards development [9]. ONF white paper [14] in 2012 defined the concept of SDN and its standard protocol OpenFlow. In SDN, the control plane is decoupled from the data plane and is programmable [14]. Figure 3 shows the basic structure of SDN. SDN is divided into three layers (1) Application Plane (2) Control Plane (3) Data Plane [10].

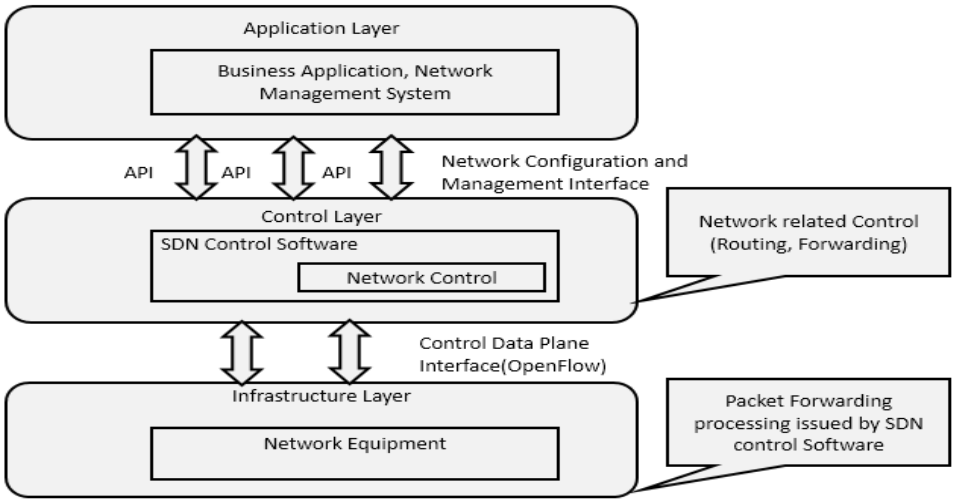

Fig. 3. The basic structure of SDN[11].

1) Application Plane:

The application label indicates a part that exploits the decoupled control and data plane to achieve specific goals, such as a security mechanism[12] or a network measurement solution[13].Applications communicate with a controller at the control plane via the northbound interface of the control plane.

2) Control Plan

The control plane is the part that manipulates forwarding devices through a controller to achieve the specific goal of the target application. The controller uses the southbound interface of the SDN-enabled switch to connect to the data plane.

3) Data Plane

The data plane is the part that supports a shared protocol (e.g., OpenFlow) with the controller and handles the actual packets based on the configurations that are manipulated by the controller.

III. OPENFLOW

OpenFlow is a protocol that allows a server to tell network switches where to send packets. OpenFlow was initially proposed by Stanford University, and it is now standardized by the ONF [15].OpenFlow is the protocol used for managing the southbound interface between control and infrastructure layer of the generalized SDN architecture [14].Openflow defines initial concept of SDN and SDN governs future development Openflow.

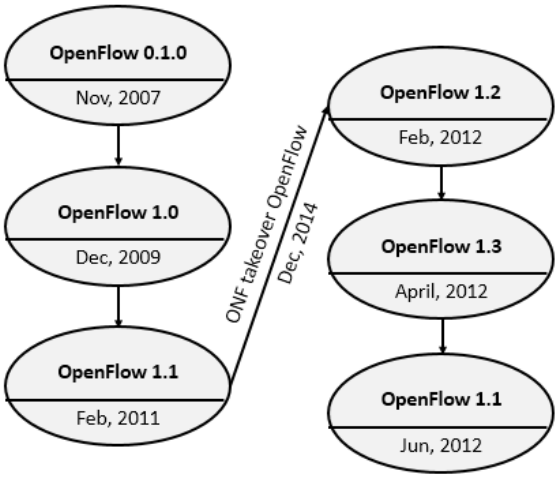

Fig. 4. The OpenFlow Version hirarchy

A. OpenFlow Architecture

The OpenFlow architecture consists of three basic concepts. (1) The network is built up by OpenFlow-compliant switches that compose the data plane; (2) The control plane consists of one or more OpenFlow controllers; (3) A secure control channel connects the switches with the control plane.

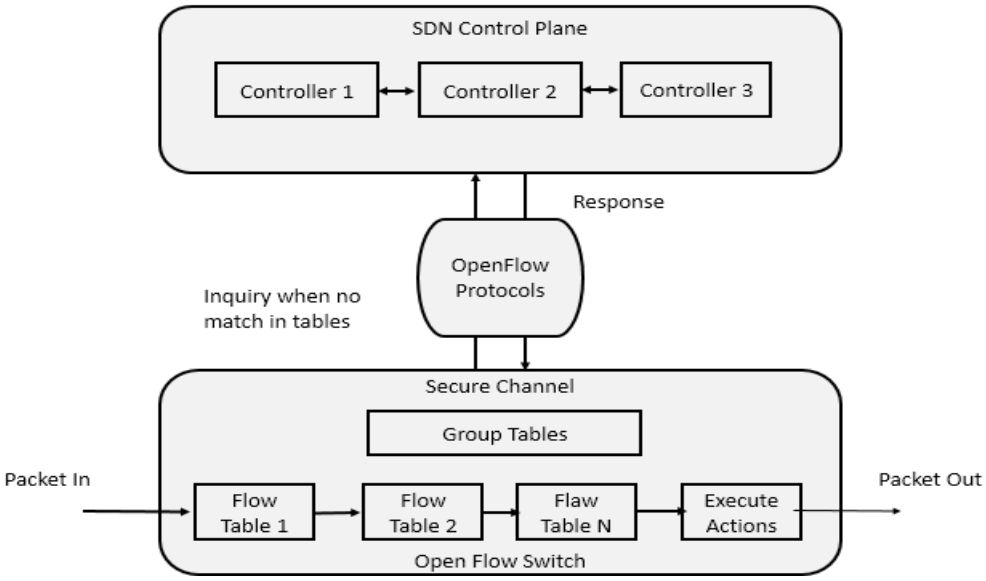

Fig. 5. The basic architecture of OpenFlow[24].

B. Openflow enable switches

An OpenFlow-compliant switch is a basic forwarding device that forwards packets according to its flow table. An OpenFlow switch consists of one or more flow tables and a group table. It performs packet look-ups and forwarding. OpenFlow-compliant switches come in two main types: OpenFlow-only and OpenFlow-hybrid. OpenFlow-only switches support only OpenFlow operations, i.e., all packets are processed by the OpenFlow pipeline. OpenFlow-hybrid switches support both OpenFlow operations and normal Ethernet switching operations, i.e., traditional L2 and L3 switching and routing[14].Flow table holds a set of flow table entries, each of which consists of match fields, counters and instructions, as illustrated in Figure 6.

Fig. 6. Flowtable entry for OpenFlow 1.0[15].

C. Openflow Controller

The OpenFlow controller is responsible for determining how to handle packets without valid flow entries. It manages the switch flow table by adding and removing flow entries over the secure channel using the OpenFlow protocol. SDN controllers can be implemented in the following three structures [17]: (1) Centralized structure (2) Distributed structure (3) Multi-layer structure.

TABLE I. OPENFLOW CONTROLLER

Name	Programming Language	License	Comment
NOX[18]	C++	GPL	Initially developed at Stanford University. NOX can be downloaded from [19].
POX[19]	Python	Apache	Forked from the NOX controller. POX is written in Python and runs under various platforms.
Beacon[20]	Java	BSD	Initially developed at Stanford.
Floodlight[21]	Java	Apache	Forked from the Beacon controller and sponsored by Big Switch Networks.
Maestro [22]	Java	LGPL	Multi-threaded OpenFlow controller developed at Rice University.

NodeFlow [23]	JavaScript	MIT	JavaScript OpenFlow controller based on Node.JS.
Trema[24]	C and Ruby	GPL	Plugins can be written in C and in Ruby. Trema is developed by NEC
OpenDaylight [25]	Java	EPL	OpenDaylight is hosted by the Linux Foundation, but has no restrictions on the operating system.

IV. SDN:STATE OF THE ART

This section described the literature survey for Software Defined Network related technology and various areas of research on which we can apply concept of SDN. Network administrators are facing difficulties in tracking the frequent users' access of network due to rapidly growing the demand and usage of network. SDN has emerged as promising solutions that change agility of internet protocol (IP) and the cost profile of network. Here we discuss the handover management, resource management as data offloading and load balancing and distributed mobility management approach using SDN concept.

A. Software Defined Networking to Improve Mobility Management Performance

Reference [26], explain mobility management mechanisms for different OSI protocol stack layers which are rely on centralized mobility management entity which is in charge of both data and control plane [27], [28], [29].Centralized mobility management has some restriction such as data processing overhead, network bottleneck, non-optimal routing and single point of failure [30] which is overcome using SDN approach that separate functionality of both control and data plane.

Mobility management performance is improved by using SDN that offering a logically centralized control model which detaches data and control plane that enable direct programing of control plane by using OpenFlow [31] communication protocol. Openflow enable switches and router comprise set of actions to improve performance of mobility management with dynamic configuration of Floatables via OpenFlow Controller (OC).

Here centralized mobility management concept for real time IP multimedia applications as Video Conferencing, Game net, and Voice over IP (VoIP) that support IP mobility replaced by distributed mobility management using OpenFlow protocol with SDN approach.

B. Software defined networking approach for handover management with real-time video in WLANs.

This paper presents SDN-based handover mechanism to reduce the number and duration of video freeze events .For handover management proposed a new architecture within the software defined networking framework that improves the quality of streaming video over WLANs [32].

The use of SDN/OpenFlow for improving streaming video, Evaluated the proposed system in a WLAN testbed deployed in an office building in downtown Berlin that consist of a streaming server act as video source that connected to WLAN APs via an OpenFlow switch .A controller dynamically configure the forwarding table of openflow switches to direct traffic from streaming server to station through right AP.

Proposed SDN based handover mechanism is not designed for fast moving users, high velocities are not possible [33] and only for limited communication factor. This mechanism consider a straight movement with 60 km/hand a communication range of 30m,handover performed at least every 3.5s.Thus,the proposed system is only suitable for pedestrian-type speed. Investigate how to couple the handover decision more tightly with the video player application for more intelligent scheduling of scanning or handovers.

C. Software defined networking-based resource management: data offloading with load balancing in 5G HetNet.

Reference [34] presents partial data offloading and load balancing algorithm to alleviate spectrum shortage concerns and to address the network congestion issues. SDN-based data offloading algorithm saves significant amount of cellular resources and decreases threshold miss probability simultaneously. As compared to the baseline algorithms, SDN-based load balancing achieves better resource allocation with higher network throughput and smaller number of handovers.

In SDN based partial data offloading algorithm, only part of the user data is offloaded onto the Wi-Fi network, remaining traffic is transferred across the cellular network that improve application performance by taking various service requirements into consideration and reduce cellular usage by leveraging Wi-Fi connectivity when available without affecting application performance [35].SDN based load balancing mechanism has ability to reduced network congestion over an area by distributing user traffic across neighboring Aps or BSs that decrease handover times and improve system performance.

By improving efficiency of data offloading and load balancing algorithm for efficient resource management for future 5G network. Data offloading will play a more significant role when cellular network has higher loads. When load is more balanced within network, resources are utilized more effectively with higher throughput by using load balancing mechanism.

D. Software Defined Network for Distribute Mobility Management

This paper [36] present new distributed IP mobility management approach by using the SDN technique in which mobility functionality implemented with the help of distributed controllers by updating involved forwarding tables directly in case of handover. This approach avoid problems like the lacking of route optimization and the single point of failure.

Proposed SaDMM architecture consists different six entities: Mobile host (MH), access gateway(AGW), proxy server, extended DNS server, Controller(CTR) and border Controller(B-CTR).Mobility management include four mobile procedures

as Attachment, Paging and connection setup, Handover and location update and Interworking between upgraded hosts and legacy hosts. A prototype of SaDMM has been implemented, the test results show that SaDMM is an efficient mechanism for mobility management oriented to the future mobile network architecture.

E. SDN-based Distributed Mobility Management

Reference [37] present DMM scheme to solved problem of centralized mobility management such as dynamic routing and IP address allocation by applying SDN concept to DMM architecture, defines procedures of mobility support and analyzed signaling coast using numerical analysis.

DMM required tunneling that causes overhead and non-optimal routing. The proposed scheme can set up data path without tunneling, because data path is set up by flow table made by controller .To evaluate proposed management scheme defined network model and cost model and formulate signaling cost and packet delivery costs. SDN based DMM can be applied in multiple domains for better performance.

F. D-PMIPv6: A Distributed mobility management scheme supported by data and control plane separation.

This paper proposes D-PMIPv6, which is to achieve a distributed mobility management scheme supported by data and control plane separation based on PMIPv6 by splitting the traditional functional entity Local mobility anchor (LMA) into two parts: Control plane LMA and Data plane LMA. D-PMIPv6 improve performance in term of the packet delivery cost compare to other mobility management scheme [38].

In MPIPv6 or PMIPv6 have single mobility anchor for mobility management as routing functionality as signaling message and data traffic forwarded to this anchor that reduce system scalability and reliability. Proposed D-PMIPv6 can solved above problem by release the burden of mobility anchor, distributing data plane traffic volumes and allocating mobility anchor near MN.D-PMIPv6 architecture split the data and control plan of LMA by a new defined Control Plane Local Mobility Anchor(CLMA)and Data Plane Local Mobility Anchor(DLMA).Two kinds of handover scenario ,In scenario 1 MN still anchors at the same DLMA after handover occurs and in scenario 2 after handover occurs, the previous DLMA is not the best choice and a new DLMA is allocated to MN.

This paper introduces a localized routing mechanism to achieve distributing the data plane traffic and allocates a closer mobility anchor to each MN by a DLMA decision procedure. Defined model to analyze the capacity of D-PMIPv6, then compare the handover performance and the packet delivery cost between PMIPv6, D-PMIPv6 and other distributed mobility management schemes in the same network topology.

G. OpenFlow-based Proxy Mobile IPv6 over Software Defined Network (SDN)

Reference [39] state that proposed PMIPv6 that is used to handle the network based local mobility management using IP tunneling that have some weakness as IP tunneling overhead and sharing same path for the data and control planes. To overcome these Openflow-based Proxy Mobile IPv6 is proposed that separate mobility management functions from component of PMIPv6.

IN proposed Openflow-based PMIPv6 (OPMIPv6), the LMA functions located in a controller and the MAG functions can be located in either the controller or an access switch. Both functions use PMIPv6 protocol to support mobility management for MNs. Controller communicates with Switches by using Openflow protocol to control switches and set data forwarding path.

OPMIPv6 is more efficient than PMIPv6 because Signaling cost and packet delivery cost of OPMIPv6 is lower than PMIPv6. OpenFlow-based PMIPv6 can be applied to the distributed mobility management [40] as the proposed scheme separates the normal and mobile traffic uses the dynamic routing update.

H. Co-operative Radial-basis Neighborhood System for Distributed Mobility Management in Software Defined Mobile Networks

This paper [41] proposes a Co-Operative Radial-basis Neighborhood (CORN) system for distributed mobility management in Software Defined Mobile Networks that used CORN protocol which creates a virtual layer of Mobile Host nodes to establish end-to-end connectivity in the network. This approach reduces handoff latency, the control overhead and rate of packet loss in the network.

The proposed protocol uses the radial basis function (RBF) [42] to create the virtual co-operative neighborhood regions across the network. SDN provide the flexibility of virtualizing the network planes. In proposed CORN an incremental approach is applied to extend the neighborhood regions of MHs covering the inter and intra-domain regions of the network.

The proposed CORN protocol achieves an improved throughput rate with reduced latency as compared to the existing open-flow and software-defined virtual network techniques and also reduced the number of handoffs and packet delays. Proposed Scheme can be extended to include inter-cellular interference, carrier aggregation of MNs and security in SDMN that includes synchronization of CORN mobile host nodes with D2D, M2M and IoT technologies of the network.

I. Applying SDN/OpenFlow in Virtualized LTE to support Distributed Mobility Management (DMM)

This paper[66] Discussed Integration of SDN/Openflow in virtualized LTE System to support DMM and compare with other DMM enabling technologies such as IETF based DMM enabling technologies[67], Double NAT[68], Distributed Mobility Anchoring(DMA)[69], Inter-domain DMM[70] and Local IP Access(LIPS)/Selected IP Traffic Offload(SIPTO).

SDN/Openflow based DMM technology satisfy most of DMM requirements such as Distributed deployment, Transparency, IPv6 development, Flexible multicast distribution, Dynamicity, Separating Control plane and Data plane and network based .Co-existence requirement od DMM is not satisfied and all routers and switches must be Openflow capable. A DMM not introduced new security risks but amplify existing mechanism that cannot offer sufficient protection.

J. Performance evaluation of a SDN/OpenFlow-based Distributed Mobility Management (DMM) approach in virtualized LTE systems

Proposed partial OpenFlow based DMM[71] and introduces a set of Functional Entities(FEs) such as FE_MCTX(Functional Entity Mobility Context Transfer),FE_I(Functional Entity Ingress),FE_E(Functional Entity Egress) and FE_IEC(Functional Entity Ingress Egress Control) which are required to support IP address continuity in a network with distributed mobility anchors[72].

Overcome limitation of Centralized Mobility Management such as single point of failure, processing overhead and non-optimal routing. The use and integration of a mobility prediction function, when the virtualized anchor point has to be migrated, in the DMM solution needs to be evaluated. Used NS-3 simulator and support user traffic redirections when the virtualized mobility anchor entity, running on a virtualization platform (e.g. Originating data center), is migrated to another virtualization platform (e.g., destination data center) and ongoing sessions supported by this entity need to be maintained.

K. A Three-Tier SDN based distributed mobility management architecture for DenseNets

Proposed three-tiered SDN architecture for DenseNet for DMM [72] by decouples control decisions and the data plane. Three tiered: Physical Layer, Controller Layer, and Management Layer for decrease delay and increase throughput over other cellular network. Provides simple and fast handover, easy management, support scalability and zero packet losses with respect to LTE based DMM.

The delay of the three-tiered SDN based DMM with MME connected to LC is less than three tiered based DMM with MME connected to GC. LC provides a periodic update to GC to maintain a global view of the whole network, and then GC updates the MME information so the delay is less; in addition, hangover is smoother and faster when MME connects directly to LC. Need Linux-based Mininet network simulators to create a virtual SDN controller (NOX) and Open vSwitch with OpenFlow protocol.

L. Comparative analysis of all existing solutions

In existing solution, we have observed some of the parameter that will evaluate performance of the SDN based DMM approaches and also address some issues. Here we have explained some of the parameters. In which we have explained some of the parameters in details like, Handover delay means the delay in the process of transferring an ongoing call or data session from one channel connected to the core network to another channel. Average Throughput, which calculated based on the number of bytes sent during the transmission between users. Packet delivery ratio is the ratio of received packet over sent packet in the network. Hop Count means number of node through which packet passed from source to destination. In Table II we have observed various parameters.

TABLE II. PARAMETER OBSERVE IN VARIOUS EXISTING PAPER

Paper	Handover Delay	Packet Delivery Cost	End to End Delay	Signaling Cost	Hop Count	Throughput
[10]	No	Yes	No	No	No	Yes
[14]	No	No	Yes	No	Yes	Yes
[38]	Yes	Yes	No	Yes	Yes	No
[36]	Yes	No	Yes	Yes	No	No
[26]	Yes	No	No	Yes	No	No
[74]	Yes	Yes	No	Yes	No	Yes
[66]	Yes	No	No	Yes	No	No
[71]	Yes	Yes	No	No	Yes	Yes
[73]	Yes	No	Yes	No	Yes	No
[37]	Yes	Yes	No	Yes	Yes	No

Here we have explained some of the issues. In which we have explained some of the issues in details like, Single point of failure means if one part of system is fails then entire network will be broken down. Controller placement means where we should place controller in given network. Optimal routing means packet should deliver to destination by optimal path. Data will be secured during transmission by using encapsulation and decapsulation technique. In Table III we have address various issues.

TABLE III ISSUES ADDRESS IN VARIOUS EXISTING PAPER

Paper	Single Point of Failure	Reliability	Non-optimal Routing	Scalability	Controller Placement	Flexibility	Security
[10]	No	No	No	Yes	Yes	Yes	Yes
[14]	No	Yes	No	Yes	Yes	No	Yes
[38]	Yes	Yes	No	Yes	No	No	Yes
[36]	Yes	No	Yes	No	No	No	No
[26]	Yes	No	No	Yes	No	No	No
[74]	No	Yes	No	No	Yes	No	No
[66]	Yes	Yes	Yes	No	No	No	Yes
[71]	Yes	No	Yes	Yes	No	No	No
[73]	Yes	No	Yes	Yes	No	No	No
[37]	Yes	No	Yes	No	No	No	No

V. SDN SIMULATORS

Network simulation is a method where a program models the behavior of a network either by calculating the interaction among the individual network entities using mathematical formula, or actually capturing and playing back observations from a production network. In the research area of computer and communications networks, simulation is a useful technique since the behavior of a network can be modeled by calculating the interaction between the different network components (they can be end-host or network entities such as routers, physical links or packets) using mathematical formulas. Network emulation, however, means that network under planning is simulated in order to assess its performance or to predict the impact of possible changes, or optimizations. The major difference lying between them is that a network emulator means that end-systems such as computers can be attached to the emulator and will act exactly as they are attached to a real network. The major point is that the network emulator's job is to emulate the network which connects end-hosts, but not the end-hosts themselves.

There are two type of Network simulators as Commercial and Open Source simulators. Some of the network simulators are commercial which means that they would not provide the source code of its software or the affiliated packages to the general users for free. All the user s have to pay to get the license to use their software or pay to order specific packages for their own specific usage requirements. Example: OPNET, QualNet. The open source network simulator has the advantage that everything is very open and everyone or organization can contribute to it and find bugs in it. The interface is also open for future improvement. Example: NS2, NS3, OMNeT++, SSFNet, J-Sim.

Several Simulation and emulation tools such as the OMNeT++ INET Framework, ns-3, Estinet and Mininet have been developed to evaluate the performance of Software-Defined Networking (SDN) [43]. Following table shows comparison between various simulators and emulators [44].

TABLE IV. COMPARION OF SDN SIMULATORS

	NS-3	OMNet++	Mininet	Estinet
OpenFlow Version	0.8.9	1.2.0	1.0.0	1.0.0/1.1.0
Programming Language	C++	C++	Python	C++
OS Support	GNU/Linux, FreeBSD, OS X	OS X, Windows, Linux distribution0 073	OS X, Windows, Linux distributions (VM image)	Linux Fedora 17 32-bit
Simulation/Emulation	S	S	E	S/E

GUI Support	Yes -Monitoring Only -Configuration by C++	Yes - Configuration - Monitoring	Yes -Monitoring Only -Configuration by Python	Yes - Configuration - Monitoring
Controller	NOX/POX[1]	-	POX	NOX/POX/Foodlight[45]

VI. SDN OPEN RESEARCH AREAS

A. Open Issues of SDN

As SDN contain 3 layer required its own level of protection. We need mechanism to secure Network virtual device and Controller[46].Various research issues in SDN are: Virtualization Security Issue[47],Controller Security Issue[48][49][50],Controller Placement Problem[51],Infrastructure Layer Issues such as Storage[52][53],processing, hardware platform and performance issues, Control layer issues as Policy and rule validation, network status collection, network status synchronization, increase processing ability, decrease request frequency, Application layer issues as Adaptive routing, boundless mobility, network virtualization, Green computing, cloud computing etc.

Various implementation issues that need to be resolved to make large-scale deployment of SDN a reality such as: carrier grade networks, securing networks, interoperability, performance and scalability. SDN aims to benefit all types of networks, including wireless, cellular, home, enterprise, data centers, and wide-area networks. The Software-Defined Networking Research Group (SDNRG) investigates SDN from various perspectives with the goal of identifying the approaches that can be defined, deployed and used in the near term as well identifying future research challenges. In particular, key areas of interest include solution scalability, abstractions, and programming languages and paradigms particularly useful in the context of SDN. The SDNRG to provide a forum for researchers to investigate key and interesting problems in the Software-Defined Networking field.

B. SDN Related Research Areas

SDN Related Researches can be categorized according to the layer of SDN architecture as Application layer, Control Layer and Infrastructure Layer. Many challenges in SDN still need further research attention and many organizations have started research projects in various aspects of SDN as shown in Table III. There are still many open problems related to SDN. For example controller
redundancy, failure behavior and interoperability between devices from multiple vendors need to be addressed before widespread adoption in the Internet.

Recent SDN researches in the infrastructure layer, the control layer, and the application layer, as summarized in Table III. The success of SDN requires improvements and developments at all the three layers, including the infrastructure layer, the control layer, and the application layer. It needs collaboration of different organizations including vendors, academia, and communities, and interdisciplinary knowledge covering both hardware and software. An SDN switching device is relatively simple with a separate
control plane.

SDN is originally developed for IP based networks on campuses. To enhance advantages of decoupling the control plane from the data plane, a high level expressive and comprehensive interface to access and control switching devices should be provided to further ease network configuration and management. Many challenges in SDN still need further research attention and many organizations have started research projects in various aspects of SDN as Standardization of SDN, Implementation of SDN and Deployment of SDN.

TABLE V .SDN RELATED RESEARCH AREAS

SDN Layers	**Related Research Areas**		
Application Layer	• Routing[54][55], • Traffic Engineering, • NDN[56][57], • Wireless network, • Computer Security, • Function outsourcing[58], • Network Virtualization, • web Caching[59] • Green Networking • Cloud[60][61][62]		
Control Layer	**Southband Interface**	**Nortband Interface**	**Performance**

	• Programing Language • Formal Methods • Compilers • Distributed System	• Distributed System Database[63] • Network Measurement	• Algorithm analysis • Software Engineering
Infrastructure Layer	**Switching Device**	**Transmission Media**	
	• Integrated circuits • Embedded System • Hardware Testing	• SDR[64] • Mobile AD Hoc • ROADMs[65]	

VII .CONCLUSION

In this paper we presented an architecture of Software defined network and Openflow protocol. We also discuss about Openflow enable switches and Openflow controller. We gave the brief description SDN for improve Mobility management, handover management, load balancing, resource management. We compare various simulators and emulators for Software Defined Network. SDN related research area for application layer, Control layer and Infrastructure layer.

REFERENCES

[1] ONF Market Education Committee. Software-defined networking: the new norm for networks. ONF White Paper, 2012
[2] Yili Gong,Wei Huang, Wenjie Wang and Yingchun Lei, "A survey on software defined networking and its applications", *Frontiers of Computer Science*, vol. 9(6),pp. 827-845,2015.
[3] Zuo Q, Chen M, "Openflow-based SDN technologies", Journal of Software,vol. 24(5), pp. 1078–1097,2013.
[4] O.N. Foundation, Software-Defined Networking (SDN) Definition.<http://goo.gl/O2eTti>.
[5] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson,J. Rexford, S. Shenker, and J. Turner, "OpenFlow: enabling innovation in campus networks," *SIGCOMM Comput. Commun. Rev*., vol. 38(2), pp. 69–74, 2008.
[6] ONF, "Open networking foundation," 2014. [Online]. Available:https://www.opennetworking.org/
[7] ONF Market Education Committee. SDN Architecture Overview. ONF White Paper, 2014.
[8] WenjuanLi , WeizhiMeng and LamForKwok, "A surveyonOpenFlow-basedSoftwareDefined Networks:Security challengesandcountermeasures", *Journal of Network and Computer Applications*,vol. 68,pp. 126-139.
[9] Open Networking Foundation. Openflow switch specification. Version 1.3.1 (Wire Protocol OXO4), 2012.
[10] Hamid Farhady, HyunYong Lee and Akihiro Nakao, "Software-Defined Networking: A survey", *Computer Networks*, vol. 81, pp. 79–95, 2015.
[11] Yili Gong, Wei Huang ,Wenjie Wang and Yingchun Lei, "A survey on software defined networking and its applications", *Frontiers of Computer Science*,vol. 9(6),pp. 827-845.
[12] S. Shin, P. Porras, V. Yegneswaran, M. Fong, G. Gu, M. Tyson, Fresco: modular composable security services for software-defined networks, in: Proceedings of Network and Distributed Security Symposium, 2013.
[13] M. Yu, L. Jose, R. Miao, Software defined traffic measurement with opensketch, in: USENIX NSDI, vol. 13, 2013.
[14] Manar Jammal, Taranpreet Singh, Abdallah Shami, Rasool Asal and Yiming Li, "Software defined networking: State of the art and research challenges", *Computer Networks*, vol. 72, pp. 74-98, 2013.
[15] Open Networking Foundation.[Online]. Available: https://www.opennetworking.org/, accessed 28 August 2016.
[16] Wolfgang Braun and Michael Menth, "Software-Defined Networking Using OpenFlow: Protocols,Applications and Architectural Design Choices", *Future Internet*,vol. 6,pp. 302-336.
[17] G. Ferro, OpenFlow and Software-Defined Networking, November2012. <http://etherealmind.com/software-defined-networkingopenflow- so-far-and-so-future/>.
[18] Gude, N.; Koponen, T.; Pettit, J.; Pfaff, B.; Casado, M.; McKeown, N.; Shenker, S. NOX:Towards an Operating System for Networks. ACM SIGCOMM Comput. Commun. Rev. 2008,38, 105–110.
[19] NOXrepo.org. [Online]. Available: http://www.noxrepo.org, accessed 28 August 2016.
[20] Erickson, D. The Beacon OpenFlow Controller. In Proceedings of the ACM Workshop onHot Topics in Software Defined Networks (HotSDN), Hong Kong, China, 12–16 August 2013;pp. 13–18.
[21] *Project Floodlight: Open Source Software for Building Software-Defined Networks*. [Online]. Available: http://www.projectfloodlight.org/floodlight/, accessed 29 August 2016.
[22] Cai, Z.; Cox, A.L.; Eugene Ng, T.S. Maestro: Balancing Fairness, Latency and Throughput in the OpenFlow Control Plane; Technical Report; Rice University: Houston, TX, USA, 2011.

[23] *NodeFLow OpenFlow Controller*. [Online]. Available:https://github.com/gaberger/NodeFLow, accessed on 29 August 2016.
[24] *Trema: Full-Stack OpenFlow Framework in Ruby and C*.[Online]. Available:http://trema.github.io/trema/, accessed 29 August 2016.
[25] *OpenDaylight*.[Online]. Available:http://www.opendaylight.org/, accessed on 29 August 2016.
[26] Morteza Karimzadeh, Anna Sperotto, and Aiko Pras, "Sotware Defined Networking to Improve Mobility Management Performance", *8th IFIP WG 6.6 International Conference on Autonomous Infrastructure, Management, and Security*, Brno, Czech Republic, springer LNCS vol. 8508,pp.118-122.
[27] Johnson, D., Perkins, C., Arkko, J., et al.: Mobility support in IPv6. IETF RFC 3775 (June 2004).
[28] Gundavelli, S., Chowdhury, K., Devarapalli, V., Patil, B., Leung, K., et al.: Proxy Mobile IPv6. IETF RFC 5213 (June 2008).
[29] 3GPP Technical Specification 29.060, General Packet Radio Service (GPRS);GPRS Tunnelling Protocol (GTP) across the Gn and Gp interface (Release 8), http://www.3gpp.org.
[30] Bertin, P., Bonjour, S., Bonnin, J.-M., "Distributed or centralized mobility?" *In: IEEE Global Telecommunications Conference, GLOBECOM 2009*, pp. 16. ,2009.
[31] *The OpenFlow Switch Specification. Version 1.3.0*, [Online]. Available: http://archive.openflow.org, accessed 01 september 2016.
[32] Peter Dely et al., "A software-defined networking approach for handover management with real-time video in WLANs", *Journal of Modern Transportation*, vol. 21(1), pp. 58–65, 2013.
[33] Sibecas S, Corral C, Emami S et al (2002) On the suitability of 802.11a/RA for high-mobility DSRC. In: Proceedings of IEEE vehicular technology conference, Birmingham
[34] Xiaoyu Duan, Auon Muhammad Akhtar, Xianbin Wang, "Software-defined networking-based resource management: data offloading with load balancing in 5G HetNet", EURASIP Journal on Wireless Communications and Networking , vol. 2015, pp. 181-194, 2015.
[35] A Balasubramanian, R Mahajan, A Venkataramani, Augmenting mobile 3G usingWiFi. ACM MobiSys, USA, 2010).
[36] Y. Li, H. Wang, M. Liu, B. Zhang and H. Mao, "Software defined networking for distributed mobility management," 2013 IEEE Globecom Workshops (GC Wkshps), Atlanta, GA, 2013, pp. 885-889.
[37] H. Yang and Y. Kim, "SDN-based distributed mobility management," *2016 International Conference on Information Networking (ICOIN)*, Kota Kinabalu, 2016, pp. 337-342.
[38] Li Yi, Huachun Zhou, Daochao Huang and Hongke Zhang, "D-PMIPv6: A distributed mobility management scheme supported by data and control plane separation", *Mathematical and Computer Modelling*, vol. 58,pp. 1415-1426.
[39] S. M. Kim, H. Y. Choi, P. W. Park, S. G. Min and Y. H. Han, "OpenFlow-based Proxy mobile IPv6 over software defined network (SDN)," 2014 IEEE 11th Consumer Communications and Networking Conference (CCNC), Las Vegas, NV, 2014, pp. 119-125.
[40] R. Kuntz, D. Sudhakar, R. Wakikawa and L. Zhang, "A summary of Distributed Mobility Management," IETF draft, August 2011.
[41] M. B. Krishna, "Co-Operative Radial-Basis Neighborhood System for Distributed Mobility Management in Software Defined Mobile Networks," *2015 IEEE Globecom Workshops (GC Wkshps)*, San Diego, CA, 2015, pp. 1-6.
[42] Wong, W. E., Debroy, V., Golden, R., Xu, X. and Thuraisingham,B., "Effective Software Fault Localization Using an RBF Neural Network", *IEEE Transactions on Reliability*, March 2012, Vol. 61(1), pp. 149-169.
[43] Hamid Farhady, HyunYong Lee and Akihiro Nakao, "Software-Defined Networking: A survey", Computer Networks, vol. 81, pp. 79–95, 2015.
[44] Pakawat Pupatwibul, Ameen Banjar, Abdallah AL Sabbagh, and Robin Braun," A Comparative Review: Accurate OpenFlow Simulation Tools for Prototyping", *JOURNAL OF NETWORKS*,vol. 10,pp. 322-327,2015.
[45] S. Y. Wang, C. L. Chou and C. M. Yang, "Estinet openflow network simulator and emulator", IEEE Communications Magazine, vol. 51, no. 9, pp. 110-117,2013.
[46] Kapil Dhamecha, Bhushan Trivedi, "SDN Issues- A Survey", *International Journal of Computer Applications* (0975-8887), 2013, 73-18.
[47] Wang, Anjing et al., "Network Virtualization Technologies, Perspectives and Frontiers", *Journal of Lightwave Technology 31.4 (2013)*, pp.523-537.
[48] "Security Implication in Data Center" [Online].Available: https://www.opennetworking.org/solution -brief-sdnsecurity-considerations-in-the-data-center, accessed 5 September 2016.
[49] Shin, Seugwon, et al., FRESCO: Modular Composable Security Services for Software-Defined Networks. To Appear in the ISOC Network and Distributed System Security Symposium 2013.
[50] Porras, Philip, et al., "A security enforcement kernel for OpenFlow networks", Proceedings of the first workshop on Hot Topics in software defined networks. ACM,2012.

[51] Heller, Brandon, Rob Sherwood, and Nick McKeown. "The controller placement problem." Proceedings of the first workshop on Hot topics in software defined networks. ACM, 2012.
[52] E. Karpilovsky, M. Caesar, J. Rexford, A. Shaikh, and J. vander Merwe, "Practical network-wide compression of IP routing tables", *IEEE Trans. Netw. Serv. Manage.*, vol. 9(4), pp. 446–458, 2012.
[53] T. Pan, X. Guo, C. Zhang, W. Meng, and B. Liu, "ALFE: A replacementpolicy to cache elephant flows in the presence of mice flooding," in *Proc.IEEE ICC*, pp. 2961–2965.2012.
[54] M. R. Nascimento, C. E. Rothenberg, M. R. Salvador, and M. F. Magalhães, "QuagFlow: Partnering Quagga with OpenFlow," *SIGCOMM Comput. Commun. Rev.*, vol. 40(4), pp. 441–442,2010.
[55] M. R. Nascimento et al., "Virtual routers as a service: The RouteFlow approach leveraging software-defined networks," in Proc. 6th Int. CFI Technol., 2011, pp. 34–37.
[56] N. Blefari-Melazzi, A. Detti *et al.*, "An OpenFlow-based testbed for information centric networking," in *Proc. Future Netw. Mobile Summit*,2012, pp. 4–6.
[57] R. Ravindran, X. Liu, A. Chakraborti, X. Zhang, and G.Wang, "Towards software defined icn based edge-cloud services," in *Proc. IEEE 2nd Int. Conf. CloudNet*, pp. 227–235,2013.
[58] G. Gibb, H. Zeng, and N. McKeown, "Outsourcing network functionality,"in Proc. 1st Workshop HotSDN, 2012, pp. 73–78.
[59] T. Li, N. Van Vorst, R. Rong, and J. Liu, "Simulation studies of OpenFlow-based in-network caching strategies," in Proc. 15th CNS Symp., 2012, pp. 12:1–12:7.
[60] T. Benson, A. Akella, A. Shaikh, and S. Sahu, "CloudNaaS: A cloud networking platform for enterprise applications," in *Proc. 2nd ACM SOCC*, 2011, pp. 8:1–8:13.
[61] G. Stabler, A. Rosen, S. Goasguen, and K.-C. Wang, "Elastic IP and security groups implementation using OpenFlow," in *Proc. 6th Int. Workshop VTDC*, 2012, pp. 53–60.
[62] J. Rubio-Loyola *et al.*, "Scalable service deployment on softwaredefined networks," *IEEE Commun. Mag.*, vol. 49, no. 12, pp. 84–93, 2011.
[63] P. Lin, J. Bi, and H. Hu, "ASIC: An architecture for scalable intra-domain control in OpenFlow," in Proc. 7th Int. Conf. Future Internet Technol., pp. 21–26,2012.
[64] T. Ulversoy, "Software defined radio: Challenges and opportunities,"*IEEE Commun. Surveys Tuts.*, vol. 12, no. 4, pp. 531–550,2010.
[65] J. Elbers and A. Autenrieth, "From static to software-defined optical networks," in *Proc. 16th Int. Conf. ONDM*, pp. 1–4,2012.
[66] Morteza Karimzadeh, Luca Valtulina and Georgios Karagiannis, "Applying SDN/OpenFlow in Virtualized LTE to support Distributed Mobility Management (DMM)", *4th International Conference on Cloud Computing and Services Science*, Barcelona, Spain,2014, pp. 86-91.
[67] Chan, H. (editor), December 2013. Requirements for Distributed Mobility Management. IETF Internet draft (work in progress). IETF.
[68] Liebsch, M., 2012. Per-Host Locators for Distributed Mobility Management . IETF Internet draft (work in progress). IETF.
[69] Seite, P., Bertin, P., Lee, J.H., 2013. Distributed Mobility Anchoring. IETF Internet draft (work in progress). IETF.
[70] Bernardos, C.J., Zuniga, J.C., 2013. PMIPv6-based distributed anchoring. IETF Internet draft (work in progress). IETF.
[71] L. Valtulina, M. Karimzadeh, G. Karagiannis, G. Heijenk and A. Pras, "Performance evaluation of a SDN/OpenFlow-based Distributed Mobility Management (DMM) approach in virtualized LTE systems," *2014 IEEE Globecom Workshops (GC Wkshps)*, Austin, TX, 2014, pp. 18-23.
[72] M., Liebsch, P., Seite, G., Karagiannis, S., Gundavelli, "Distributed Mobility Management-Framework & Analysis", IETF Internet draft (work in progress), February 2014.
[73] I. Elgendi, K. S. Munasinghe and A. Jamalipour, "A Three-Tier SDN based distributed mobility management architecture for DenseNets," *2016 IEEE International Conference on Communications (ICC)*, Kuala Lumpur, 2016, pp. 1-6.
[74] M. B. Krishna, "Co-Operative Radial-Basis Neighborhood System for Distributed Mobility Management in Software Defined Mobile Networks," 2015 IEEE Globecom Workshops (GC Wkshps), San Diego, CA, 2015, pp. 1-6.

