CloudSDN: Enabling SDN Framework for Security and Threat Analytics in Cloud Networks

Prabhakar Krishnan(✉) and Krishnashree Achuthan

Amrita Center for Cybersecurity Systems and Networks,
Amrita Vishwa Vidyapeetham, Amrita University, Amritapuri, Kerala, India
kprabhakar@am.amrita.edu

Abstract. The "Software-Defined Networking (SDN), Network Function Virtualization (NFV)" are recent network paradigms and "OpenStack", a widely deployed Cloud management platform. The goal of this presented research work is to integrate the SDN, NFV into OpenStack based Cloud platform, draw practical insights in their inter-play, to solve the problems in the Cloud network orchestration and applications security. We review key prior works in this intersection of SDN, NFV and Cloud computing domain. The OpenStack based Cloud deployment integrates SDN through its Neutron module, which has major practical limitations with respect to scalability, security and resiliency. Aiming at some critical problems and overall Cloud security, we postulate certain SDN scheme that can distribute its own Network Function (NF) agents across the dataplane and deploy applications across the control plane that centralizes the network management and orchestration. A novel security scheme for Cloud Networks "CloudSDN", enabling SDN framework for Cloud security is proposed and implemented, addressing some well-known security issues in Cloud networks. We demonstrate the efficacy of the attack detection and mitigation system, under Distributed Denial of Service (DDoS) attacks on the Cloud infrastructure and on to downstream servers as well. We also present a comparative study with legacy security approaches and with classical SDN implementations. We also share our future perspectives on exploiting the myriad of features of SDN such as global view, distributed control, network abstractions, programmability and mitigating its security issues.

Keywords: SDN · NFV · DDoS · Intrusion Detection Systems (IDS) · Intrusion Prevention Systems (IPS) · Cloud · OpenStack · Network security

1 Introduction

Cloud Computing is a paradigm that aims at enabling ubiquitous, on-demand access to a shared pool of configurable computing and infrastructure resources. The modern Cloud data centers are designed for enterprise needs, distributed computations and data intensive applications, composing computational servers, data-storage systems internetworked with routers and Internet facing gateway devices. In large Data Center Networks (DCN), the prevalent security systems are usually connected in series mode

N. Kumar and R. Venkatesha Prasad (Eds.): UBICNET 2019, LNICST 276, pp. 151–172, 2019.
https://doi.org/10.1007/978-3-030-20615-4_12

causing network congestion and these mechanisms themselves become bottlenecks and offer limited protection in specific static network paths. The computing resources which include both hardware/software and networks, are usually geographically distributed across the globe, thus imposing challenges to the interconnecting network & operations. To solve this very issue of traffic orchestration and engineering, the emerging paradigms such as SDN/NFV are crucial to meet the user demands. The SDN enabled networking architectures, offer features such as programmability, flexible reconfigurations, dynamic policy enforcement and global views.

Security and privacy are of critical concern to cyber security and data center administrators and for Cloud service providers. For legacy network environment, SDN can be a value-add-on, whereas for today's Cloud data centers, in Clouds, virtual network implementation it is essential. In the networking domain, "Software-Defined Networking (SDN)" [1] is emerging as the most disruptive paradigm, redefining network architectures, topologies, orchestration and complex policies [2] of large applications, data centers and Cloud infrastructures. Network Function Virtualization (NFV) [3] is one of the rapidly adopted paradigms in the modern data centers, that offer virtualized networking services & functions as "Virtualized Network Functions (VNFs)".

Current day networking applications demand advanced services with varying policy-processes, so in data centers need to: "line up a sequence of NFs, various types of state changes by NFs: changing the packet contents (e.g., Network Address Translation-NAT changes addresses/ports), dropping packets (e.g., firewall), or absorbing packets and generating new ones (e.g., L7 load balancer terminates client's TCP session and establishes new session with the appropriate server)". In virtualized SDNFV data centers, the SDN controller can't track the packet-streams/flow, as it doesn't have full view of the NF processing functions that are either embedded in monolithic kernel or implemented as hardware chip in middlebox appliances. Therefore, limitations in SDN's global view of the network states of sessions, problems in optimal NF service chaining, have not been solved yet in these researches.

Further, to address open problems in Cloud security [4–6], "SDNFV enabled architectures help to bring in easy solutions and mechanisms for these cyber-threats. SDN defines the decoupling of the control plane and the data plane that share the traditional network equipment. On one hand, such decoupling is beneficial as it enables centralized decisions about data traffic in networks. This way, policies can be enforced quickly in response to emerging network requirements, as well as to network threats. On the other hand, SDN Security issues [7], such as fraudulent rule insertion, controller-switch communication flood, unauthorized controller access, and controller hijacking, could be exploited in Cloud environments to harm client applications and network performance. From the security point of view, it is relevant to investigate whether SDN constitutes a solution or a problem for Cloud Computing environments, since the answers to this question are important indicators of the trust that a Cloud customer can place in SDN and SDN based Cloud computing services. There are several proposals in the literature that address SDN security. Some position SDN as an additional defense measure to tackle security threats, IDS/IPS and DPI solutions and other proposals address SDN architectural vulnerabilities". Although many proposals are available in the literature, to solve legacy/traditional networking problems, there is a

dearth of concrete feasible design, that addresses the applicability of SDN in securing Cloud networks.

Our work begins by arguing "that the current SDN match-and-action model is rich enough to implement a collection of anti-spoofing methods. Secondly, we develop and utilize advance methods for dynamic resource sharing to distribute the required mitigation resources over a network of switches. None of the earlier works attempted to implement security/defense mechanisms in the SDN switch directly and exploited the match-action power of the switch data plane. They just implemented applications on top of the match-and-action controller model and these control programs monitored the flows to enforce security policies. Our method builds on the premise that the SDN data plane switches are reasonably fast and efficient to perform low level primitive operations at wire speed. As such solutions require a number of flow-table rules and switch-controller messages proportional to the legitimate traffic, in order to scale when protecting multiple large servers, the flow tables of multiple switches are harnessed in a distributed and dynamic network-based solution".

Through this research work, we propose a security framework *CloudSDN*, and implemented a security scheme with attack detection mechanism in data-plane and mitigation control in the SDN control plane. Our experiments prove that only a marginal change in processing costs for this co-operative security scheme in SDN (controller/switch). Further, this scheme give protection to the SDN infrastructure from getting into control-plane saturation, flow-table/miss attacks and surely defends downstream servers, middlebox appliances in the network. As data plane is where packets are processed, switches should be enabled with new packets processing functions for DDoS coarse-grained attack detection and mitigation action. We implemented a SDN Integrated Cloud Management system that consists of security monitoring data plane and threat analyzing control plane. We introduced new mechanisms in the SDN stack and run-time library for defense applications. Our evaluations have proved that the extensible stateful SDN data plane within the framework, with NF service chaining, provides superior security compared to traditional firewall/perimeter solutions. The framework also offers developers a set of API & library to implement their custom network functions (NFs) policy and deploy NFVs in CloudSDN framework. We have embraced the OpenStack Cloud [8] and "Open Virtual Network (OVN)" technologies [9] to build the SDN-NFV enabled Cloud computing environment. Our framework is deployed as an active defense mechanism against DDoS Amplification and flooding attacks in Cloud environments and the efficiency is evaluated under various scenarios and comparative analysis with legacy/other SDN approaches.

The rest of the paper is structured as follows: Sect. 2, presents the background for SDN/NFV enabled Cloud computing and articulate our understanding and outline related works in Cloud security. In Sect. 3 we present our proposed SDN-enabled framework for Cloud security called *CloudSDN*, in Sect. 4 we will have a design & implementation discussion and in Sect. 5 we share the results of our preliminary experimentation. The Sect. 6 concludes this paper with summary highlights and an outlook on using SDN for future virtualized modern data centers.

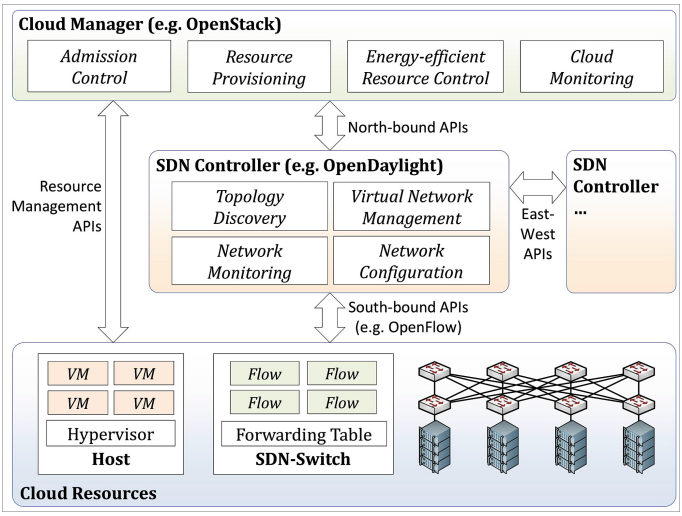

Fig. 1. SDN-centric cloud architecture [10]

2 Background and Motivation

We present here, the necessary background discussion of relevant technologies to build SDN-enabled Cloud networks and related works.

2.1 SDN-Enabled Cloud Computing

The Cloud computing applications dynamically demand new provisioning, traffic QoS and swift response to changing events/incidents. With SDN emerging as a reference architecture (Fig. 1) for Clouds, many modern data centers have embraced this paradigm shift for SDN-enabled Cloud Computing [10], such as Software-Defined Wide Area Network (SD-WAN), SD-Clouds. IBM proposed one of the first SDN enabled Cloud architecture called Meridian [11] with "OpenStack and IBM's Smart Cloud Provisioning technologies". They adopted programming model of SDN for provisioning and network management. PDSDN [12] project proposed a scheme/policy layer for "SDN controller-to-Cloud Manager" to improve the interactions with Cloud users/tenant. They implemented on "SDN Open Floodlight and OpenStack Cloud". Mayoral et al. [13] introduced SDN OpenDaylight controller into OpenStack Cloud platform and proved an improved network orchestration service in this integrated Cloud infrastructure.

2.2 SDNFV Converged Architecture

Deploying SDN in legacy IT data centers, require a series of changes in terms of redefining architecture, topologies, security policies, access-control mechanisms and so on. The networking appliances and routing equipment are substituted with virtual software switches (data plane).

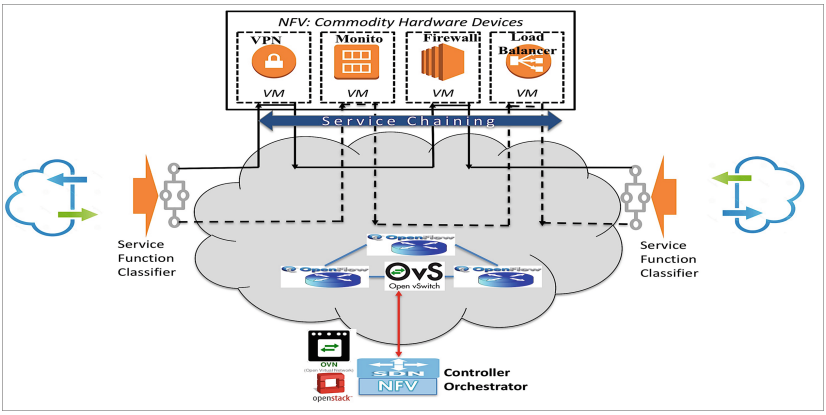

Fig. 2. The emerging SDNFV cloud computing paradigm

The Service Function Chaining (SFC) concept has emerged as a critical operation for IT networking and large-data center Clouds service providers, to establish a sequence of services and NFs (e.g. Firewalls, DPI, Load Balancing). This sequence of services is ordered in a "service list or chain", that is depicted in Fig. 2, namely SDNFV-enabled Cloud computing. The SFC traffic then is forwarded through this service chain by network components. The convergence of SDN and NFV "SDNFV-enabled Cloud computing" will unravel new paradigms, architectures and research problems/solutions for both academia and industry, as they complement and bring cost savings in terms of Capital Expenditure/Operational Expenditure (CAPEX/OPEX) and operational agility, energy saving, elastic provisioning, dynamic security for Cloud computing.

2.3 SDN - OpenStack Interaction Model

In our systematic research of SDN and OpenStack Cloud platform, we postulate that both these technologies can together deploy efficient solutions to enterprise data centric Cloud computing scenarios. Given in Fig. 3, Neutron subsystem is the OpenStack SDN component responsible for ensuring that virtual machines (VMs) have a functioning network. Neutron acts as an abstraction layer with its own plugin mechanism, which gives it the flexibility to incorporate SDN data plane components. OpenStack integrates SDN services through Neutron module. The Neutron server is a RESTful-based API in typical OpenStack style. The Neutron API is the point of contact for any request relating to the SDN configuration in an OpenStack Cloud. The Neutron Server, now extended to include an SDN plugin, acts as a central source of knowledge for all SDN-related information in OpenStack. Virtually all SDN approaches stipulate that flows of traffic from or to the Internet use a separate gateway, which is configured directly from the Cloud.

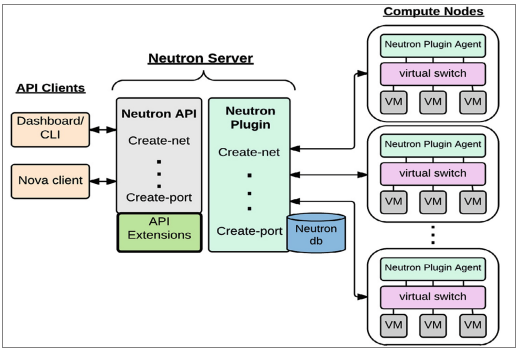

Fig. 3. OpenStack neutron architecture

But the OpenStack Neutron has major practical limitations with respect to scalability, security and resiliency. The reason is that "Neutron does not have its own Layer 3(L3) routing capability, but it uses the Linux kernel bridging and routing mechanisms instead. In a large Cloud environment with a lot of virtual networks, tenants, and applications, all traffic requiring routing and floating IP services need to be handled by the same Neutron L3 agent. Therefore, the agent becomes the choke point. SDN solutions can distribute their own L2/L3 agents among OpenStack nodes to help eliminate Neutron L3 agent bottleneck issue. And SDN controllers centralize the management of physical and virtual networks, so it helps simplify managing and monitoring tasks. OpenStack supports RESTful APIs for every component".

2.4 New Opportunities and Challenges in Cloud Security

SDN offers programmable networking infrastructure and NFV is capable of virtualizing network functions and both independently/combined offer new ways to monitor and secure Cloud networks. The two big questions for SDN in the networking field are: 1. Can SDN centric architecture secure Cloud networks? and 2. How Can SDN architectural vulnerabilities be protected? (Table 1).

2.5 Related Work

The authors of [14], propose an extension to controller for detecting DoS attack based on forwarding flow-tables on switches. In [15] Kumar et al. combine IDS and virtual switches, but the fine-grained latency & forwarding delay measurements were not presented. A "Moving-Target-Defense" scheme which modifies VM's identity is proposed by this paper [16], but the authors don't address insider threats in Cloud. In [17], the authors embed IDS into the control plane function and security is enforced based on dynamic changes to flow-table, but this adds overhead to controller process and choke point. The authors of [18], implemented IPS into controller process, but limited to POX controller. The authors proposed a Cloud-IPS with SDN in [19] and they leveraged the flow-table match-action/miss & sendto_controller as part of the

Table 1. SDN for cloud computing

Cloud networking issues	Advantages	Disadvantages
Proper installation of network firewalls	Central control logic and global topological view help to identify threats efficiently and accurately. Also, quick response to incidents and dynamically pushing the policies/rules	Unauthorized Access could compromise the firewall rules and policies of the network
Network security configurations	Legacy network architecture use appliances and vendor specific tools to orchestrate and manage configuration. SDN use standard interface/API to controller and OF protocol to switches for programming specific configurations	The operation of SDN paradigm revolves around the control protocol standard OpenFlow, which enables the data exchange between the controller and switches and applications. This opens up a critical attack vector for adversaries to saturate the control plane or MITM attacks disrupting the topology & policies. So the security and availability of SDN operations are critical
Internet protocol vulnerabilities	Legacy network protocols are designed based on packet-level decisions or policies on switches or routers. But this leads to congestion & bottlenecks. In SDN, flow-based traffic engineering and orchestrations are done by central control plane with global view, leading to enforcement of consistent policies	Legacy network architecture involves a series of network functions/protocols executed by separate entities. But in SDN, as the network is virtualized into programs and software applications are prone to design flaw or implementation vulnerability
QoS ("Quality of Service")	As SDN offers a programmable network architecture, it's easy to implement QoS policies and run time enforcement using dynamic mapping functions	The SDN based network and data centers incur higher communication overhead and provisioning of network bandwidth, compared to legacy network
Multi-tenant architecture	As SDN architecture is more software centric, its easily programmable for dynamic elastic provisioning models in a large complex multi-tenant data center	The operation of SDN paradigm revolves around the control protocol standard OpenFlow, which establishes flow-tables (equivalent to routing table in legacy network) in the series of switches in the data plane. It might be a challenge to enforce complex tenant SLAs/QoS, priorities and co-existing security/privacy boundaries just by using these flow-table pipeline. It may require sophisticated application software which again opens up bugs and vulnerabilities

OpenFlow protocol. In [20], authors presented a framework "CloudWatcher", using a scripting interface – the specific suspicious flows in the network may be programmatically diverted to scrubbing nodes for further security screening. Yan et al. [21] did a systematic-review on DDoS mitigation with SDN capabilities, at the same time the architecture of SDN paradigm comes with critical vulnerabilities such as control-plane saturation, single-point-of-attack and other side-channel attacks. Chowdhary et al. recommended [22] a novel framework for DDoS mitigation based on game theoretical approach and they demonstrated their solution with OpenDaylight Controller in Mininet simulated SDN environment. Foresta et al. [23] presented the advancements in using SDN OpenvSwitch data plane mechanism as a firewall in OpenStack and compared various performance metrics with the native Linux bridge. The authors in paper [24] proposed an scalable SDN/NFV monitoring framework, by integrating in OpenStack Neutron subsystem and they evaluated in real time traffic monitoring use cases. The authors of [25] proposed a comprehensive analysis of enabling SDN for security in IoT networks and discussed design choices. They further demonstrated the efficacy and feasibility of a SDN Framework for fine grained security monitoring in the data plane, with exemplar applications for defending DDoS/Botnet Attacks.

3 Proposed Architecture

3.1 Architecture Overview

SDN Integrated Cloud Management Framework (Fig. 4) 'CloudSDN' consists of security monitoring in data plane and threat analyzing in control plane. We introduce the major Components of the *CloudSDN* Framework below:

- *Infrastructure*: This layer consists of virtual machines, physical hosts and Cloud infrastructure devices (IoT, mobiles, hubs, modems, services, applications). This resource layer is managed by the 'OpenStack Nova' directly.
- *Switches:* This layer consists of OpenFlow (OF) switches, Core switches, hybrid OF-enabled Edge switches. This layer is managed by the Controller, through OF protocols for data switching, security monitoring and policy enforcement. Probes/Sensors monitor flows/packet-stream, if any anomaly is detected, that flow is flagged (i.e., "DDoS attack"), that specific switch sends in-band message (enclosing the flow-metadata, alerts, extracted feature-digest, synopsis) to Trigger Core Switch or Controller. The corresponding defense-action & cleanup command (NF) will be executed by Mitigator, on that specific Edge switch by Actuators in the data plane.
- *Control Plane:* This layer consists of SDN Controller (OpenDaylight) modified with extensions for the new security monitoring, defense and attack mitigation functions. It leverages uploaded in-band message, feature-digest, synopsis to classify attack type and its global view of the network topology & attack sources. It then calls the defense-action library to implement specific defense-action in the switches that are in the path or closest to attack source.
- *Cloud Admin:* This layer implements Cloud management technologies such as OpenStack, with plug-ins and extensions to Neutron Layer for the SDN based

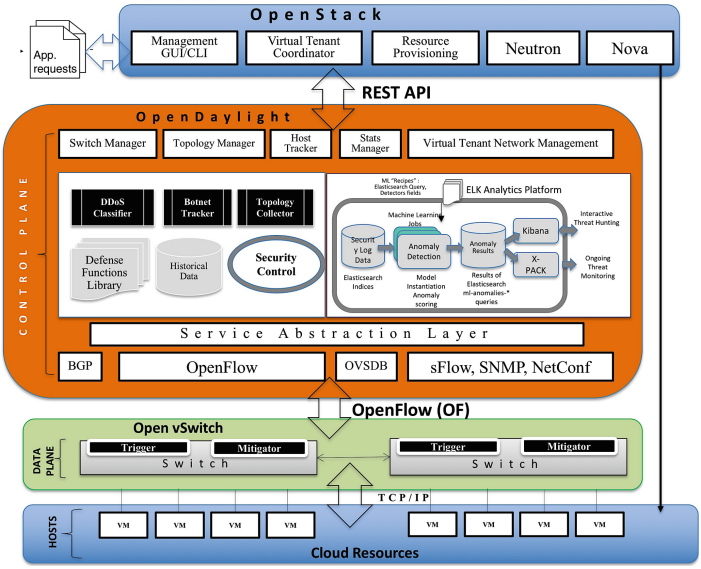

Fig. 4. CloudSDN - SDN integrated cloud management framework

security schemes. The Cloud users interact for services through REST API and Nova Layer establishes a "Virtual Network" for the new tenant.

– *Data Plane*: The following are the major steps in the network packet workflow on the data plane switches – (i) "new flow" in the switch (data plane) will be directed through the IDS/IPS embedded in the switch, chain of NFs following the "match-action" semantics of OpenFlow pipeline. (ii) As IDS/Trigger (NF) detects an Intrusion/Attack (through a 'challenge-response' method), controller will be notified through special OF message (in-band), defense actions will be distributed through flow-table actions (diversion or drop or filter or throttle), executed by IPS/Mitigator in the switch(es) in the path of the attack packets all the way back to the source.

3.2 Multi-plane Collaborative Defense

Our design goal is to architect a holistic monitoring and automated defense framework with fast attack detection. As the single-point of defense (centralized) cannot fully eliminate the threats in large network, hence the case of a multi-plane approach, that has distributed security monitor/enforcement with central control mechanisms deployed at key vantage points of the network. We further argue in SDN architecture, the control layer should do attack analysis on flows and control functions (e.g., attack classification and traffic trace-back). The controller should be responsible for conducting fine-grained attack detection and making high level defense strategies, leveraging its global view of the whole network, abundant computational and storage resources for historical data analysis. As data plane is where packets are processed, switches should be enabled with new packets processing functions for DDoS coarse-grained attack detection and

mitigation action. Therefore, on the data plane, a lightweight monitoring mechanism (sensor/probe) identifies attacks with the features extracted from the flows. After attacks are identified, the control plane makes a set of strategies to react. Enabling defense actuators on the data plane dynamically is a key step for executing these strategies.

Once the CloudSDN framework starts running, Sensors on the Edge switches keep monitoring every flow on the data plane constantly. If any abnormal flow is captured (i.e., DDoS attack), the specific switch notifies an appropriate Core switch. In order to respond more quickly against DDoS attack and reduce the workload of controller, a coarse-grained attack detection algorithm & trigger mechanism are implemented in the data plane hierarchical switches, which acts a security middlebox/proxy to the controller. The Core switches then invoke pre-defined action-set which includes a proxy-challenge/response technique to detect DoS attack type or handle the packet in the data plane itself with relevant rules. If there is no matching action-set, the Core switch asks the Edge-switches to do sampling of suspicious flows and triggers new fine-grained attack detection/classification process in the control plane, with information of extracted attack features. Over the control plane, the threat analytics system leverages uploaded attack features to classify DDoS-attack type and its global topology perspective to locate the attack sources.

3.3 Security Service Function Chaining

We implemented dynamic Virtualized Network Function (VNF) service chaining through loadable modules/NFs/applications on the data plane switches. The policies for optimal chaining and placement of these NFs are determined by the Controller. To this end, we extend the classic Open vSwitch (OvS) stack to add stateful-functionalities and security-awareness. So, this translates to eliminating the middleboxes or standalone NFV machines (VMs) out of the data center. Some example NFs that can be deployed as dynamically loadable modules are: “Firewall, load-balancer, Proxy, NAT”. We may argue that additional CPU resource is required to execute the VNFs either in VMs or in Switches. Our experience revealed that there is only a marginal difference in network overhead between two approaches. So, for IDS either at Wire-speed solution, NF’ service chaining in OvS switch OpenFlow pipeline seems to be the trade-off.

3.4 Operation of CloudSDN

The operation of CloudSDN at run time is given in (Fig. 5), The security scheme in the SDN stack is divided into two phases: detection phase and reaction phase.

In the detection phase, which is spread across both data plane and control plane: a lightweight anomaly detection Statistical/Feature-based flow monitoring algorithm is proposed to serve the data plane as DDoS-attack sensor. The DDoS-attack traffic manifests higher volume and asymmetry in the network and we will monitor these features for our detection. On the control plane, a machine learning DDoS-attack classifier and applications are utilized to locate a DDoS attack in finer granularity (for e.g. attack type, malware, botnet origin location). Specifically, features extracted from attack traffic and holistic information of the network are fed into DDoS-attack classifier and botnet tracker.

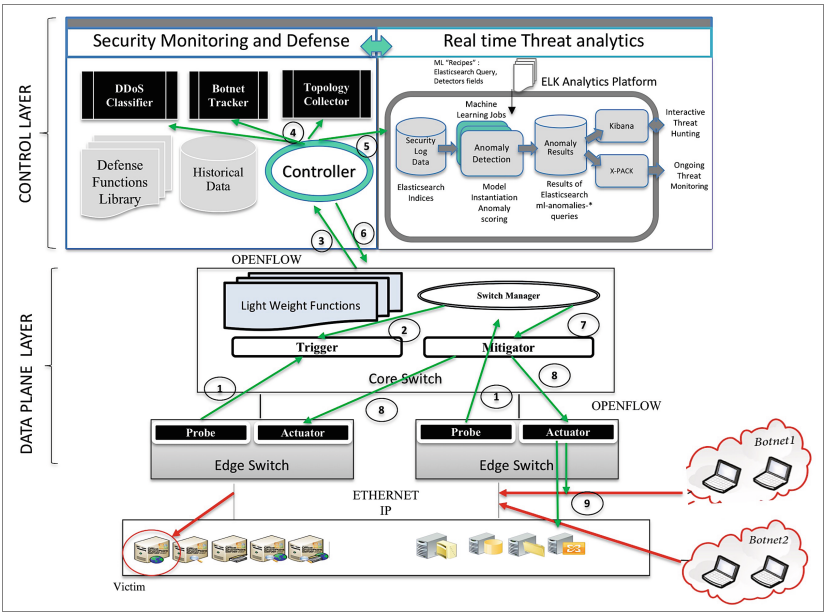

Fig. 5. Operation of CloudSDN framework

In the reaction phase, first level reactive functions executed in data-plane and second level reaction occurs later in control plane, based on the results obtained from the detection phase. A novel defense strategy offloading mechanism is proposed to enable DDoS attack defense actuators to be executed on the SDN Core/Edge switches

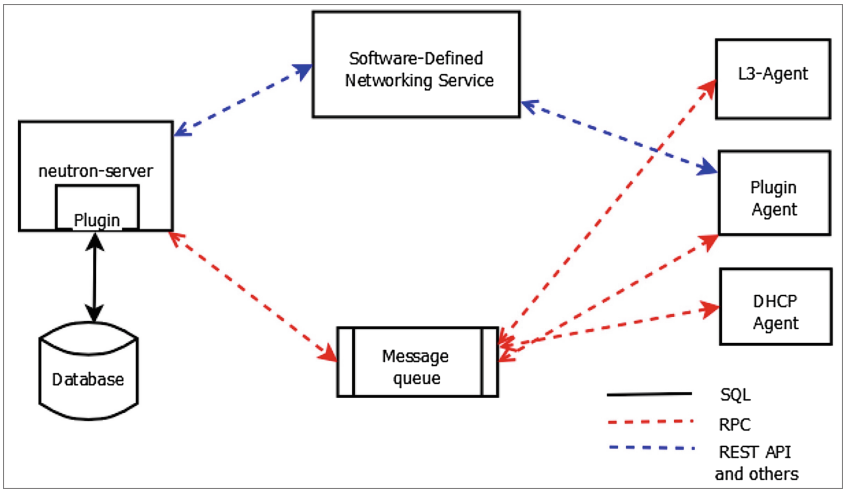

Fig. 6. OpenStack neutron and SDN services

automatically. Thus, SDN controller can be free from conducting specific defensive actions, resulting in attack reaction efficiency and overall traffic load optimization. More specifically, we concentrate on exploiting the computational resources of switch CPUs and the flexibility of southbound interface, in order to deploy defense actuator NFs on the switches which are closest to the botnet.

4 Design and Implementation

The design strategy is to propose extensions to the existing classical SDN architecture and OpenStack Neutron layers, to realize a multi-plane cooperative DDoS security framework.

4.1 OpenStack Neutron SDN Layers

The Fig. 6 depicts the interconnection between the SDN services and OpenStack Neutron layer. The interaction model within Neutron is shown with different colored lines. SDN services are defined as Python class and Neutron invokes them through API calls.
Neutron Plugin: The "Modular Layer 2 (ML2)" plugin, implements generic API, as a "plug-and-play" driver. We used the mechanism drivers for Open vSwitch (OVS), OpenDaylight (ODL). The plugin does all networking services ("creation, updation and deletion of networks, subnets and port resources, port binding") and provides connection between VM's and Cloud controller, also to outside networks.

Neutron Agent: Every plugin comes with an agent module on compute nodes and connect to the virtual switch (OVS) on the node. We designed new plugin using the ML2 core backend and opensource OF Agent to run on virtual switch itself (by extending the OpenvSwitch). The workflow inside Neutron is - "(i) An operation request is sent through the API to the Neutron server. (ii) The Neutron server makes an entry in the database and invokes the corresponding plugin via a REST API call. (iii) Upon receiving this request, the plugin calls the southbound protocols to perform necessary changes to the network elements".

4.2 OpenvSwitch SDN Data Plane

Our baseline implementation is derived from OpenvSwitch of OVN project. The data plane in CloudSDN Architecture (Fig. 7) is not just a group of forwarding entities but consist of a group of software sensors and actuators to detect and react to DDoS attacks. We have added another authoritative-layer/aggregation/core switch in the data plane acting as a main security firewall in the dataplane level, with attack '*Trigger'* and '*Mitigator'*. These can be a subset of existing switches in the network or dedicated switches or routers that have larger memory and processing capability. These Core switches run the complex NFs (service-chain or proxy or security challenge-response) for attack detection, coarse grained flow/packet inspection. These switches will execute

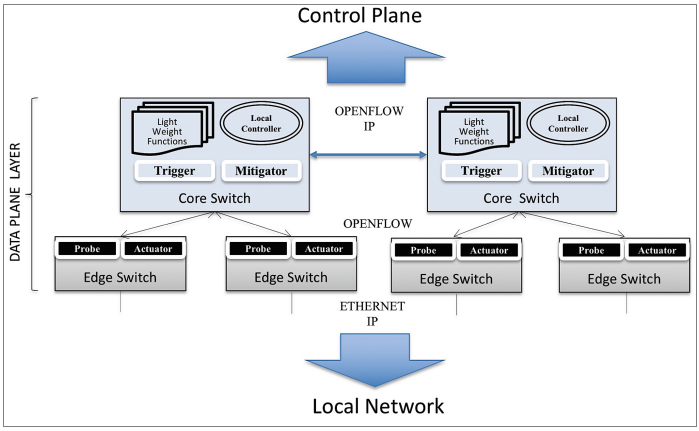

Fig. 7. Data plane operation in CloudSDN

some of the offloaded stateful functions (cached flow-rules/action set) instructed by the controller, on the Edge-switches which have '*probes*' and '*actuators*'.

This leads to our three key functionalities to enhance the existing switches in the data plane. (i) Capture key signatures/features of attacks. (ii) Load mitigation functional modules as instructed by controller and (iii) Execute the mitigation functions to handle the attack traffic. Unlike many other monitoring methods, the monitoring NFs that run as threads on switch software, extract key features of DoS-attacks by polling stat counters of OpenFlow (OF) switches. For non-OF switches the monitoring is done by querying through standard protocols such as SNMP, IPFIX, sFlow.

The functional components of the data plane layer include:

- *Attack probe/sensor:* smart components embedded into Edge switch, that runs a lightweight monitoring logic to detect attacks from hardware/port and changes in flow patterns/characteristics, by exploiting advanced match/actions in OpenFlow 1.5.1 Flow table. For non-OpenFlow switches, we enable the switch management protocols such as SNMP, sFlow, IPFIX.
- *Connection-state:* consists of flow-table analyzer, well-known attack signature database, learning-engine that correlates traffic patterns. dissects packet's headers, creates connection endpoint-profiles from metadata, geo-location, time-stamp and creates profile-synopsis for threat analysis.
- *Feature Extraction:* traffic matching specific rules are captured for attack signature/feature extraction. The *features-digest* includes volume and asymmetry.
- *Attack Detection Trigger:* Active smart probes/sensors deployed in key defensive points in the network (SDN, Legacy, IoT) which may indicate DDoS attacks (suspicious or abnormal traffic pattern, statistical thresholds, matching-flows, stateful filters, rate/velocity of key protocol messages) sends out alerts or trigger commands to controller.

– *Mitigation:* This hosts a suite of Flow/packet-handling network functions virtualized as micro-services/modules. To tackle a botnet campaign (multiple attacks may pass through switches simultaneously) multiple mitigation (chain) has to work independently on single/multiple switches.

4.3 OpenDaylight SDN Control Plane

The controller based on OpenDaylight SDN implementation runs the entire network, as the brain of the framework. It inspects the current DDoS attack (e.g., attack types and its traces) and makes proper strategy to defend it. It should be able to: (i) classify DDoS attacks and (ii) track the botnets, locate sources of attack. The components are:

– *Topology Collection*: runs the LLDP protocol for SDN networks and IPFIX,SNMP, sFlow, NetConf protocols for discovering the topology from legacy-IP networks.
– *Flow Analyser*– This sweeps & correlates the historical flow-tables and active flow-tables deployed by controller-to-switches. This augments the coarse-grained analysis done at data plane, by working on historical larger-data set and anomaly detection, fine-grained correlations between flow-tables from all over the network.
– *Attack Classifier/Detection:* Main Classifier/Detector of any attack type, leveraging the extracted attack-synapses (a small table of extracted features from attack packets) received from switches and trained with dataset samples from known sources.
– *Analytics Engine:* with the help of synopsis-object and features-digest, runs through a suite of classifiers workflow based on machine learning, rate-limiting, entropy based, behavioral correlation and anomaly-based algorithms.
– *Botnet Tracking and Attack Traceback*: It retraces the attack path, identifying the switches in the path from the victim to source network. Starting from point of attack detection (victim's network) moving backwards in the opposite trajectory of attack traffic, the traceback engine, queries the devices (switches/middleboxes) for attack-synapses (small auxiliary table consisting of device profile, metadata and specific features from flows, flow-table entries for the adjacent, neighborhood switches). Using SDN controller's global network topology view, A co-operative traceback analytics algorithm is run across all the participating switches (using our customized OvS switch NF) to trace plausible paths to the source of attack. On analyzing the attack synapses data set, the malicious flow path, the attack source and the affected switches are identified.
– *Attack Mitigation/Response:* When attack-type/botnet-pivots and affected switches in the attack path, source network, are determined, this module sends instructions to those switches with "mitigation cleanup" action through OpenFlow/other standard dataplane protocols. This will block the attack traffic upstream and by purging malicious flows off the switch tables it also prevents ternary content-addressable memory(TCAM)bloating and packet drops issues in the network path.
– *OVSDB:* OpenvSwitch Database records all the flow information; flow rules are installed on to OpenFlow enabled Core Aggregating and Edge switches.

4.4 Implementation

OpenStack has already adopted some of the networking functions of SDN implementations. To optimize and also to secure the OpenStack Cloud deployments we have developed our native SDN components and interfacing modules for general legacy switches. As OpenvSwitch(OvS) based Virtual switches are used in more than 60% of the SDN/NFV enabled data centers, we used it as baseline and implemented extensions. The Fig. 8 shows the implementation of our SDN stack with stateful layers and the packet flow. It consists of two stages processing of application-logic/stateful tables within the switch, spread across User/Kernel spaces and the stateful/application control within the controller.

We illustrate the workflow of the typical Cloud computing use case in Fig. 9. A tenant requests for resources to OpenStack, then Nova module provisions required VM instances in the cloud. Then the SDN OpenDaylight (OD) controller schedules a virtual network (VN) through a RESTful call. The OD calls OpenFlow(OF), OpenvSwitch database (OVSDB) to configure VNs and send the Flow rules to OF switches. The Topology manager stores the VN configurations and topologies. Depending on the operational conditions, this pool of resources in the Cloud may be provisioned or reconfigured. We integrated the SDN and OpenStack in an experimental network, with a suite of anti-DDoS applications built into the package. The OVS switch on top of each cluster of compute nodes will monitor, detect attacks and execute mitigatory Network Functions (NFs) to restrict the attacks to the data plane itself and constantly communicating the statistics to SDN controller through the Neutron Plugins.

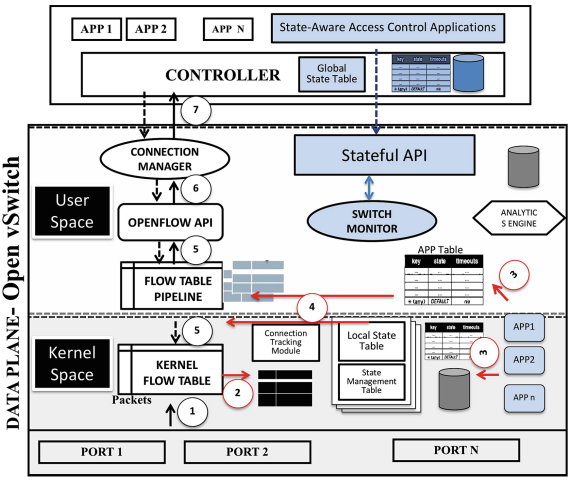

Fig. 8. Modified SDN stack in CloudSDN

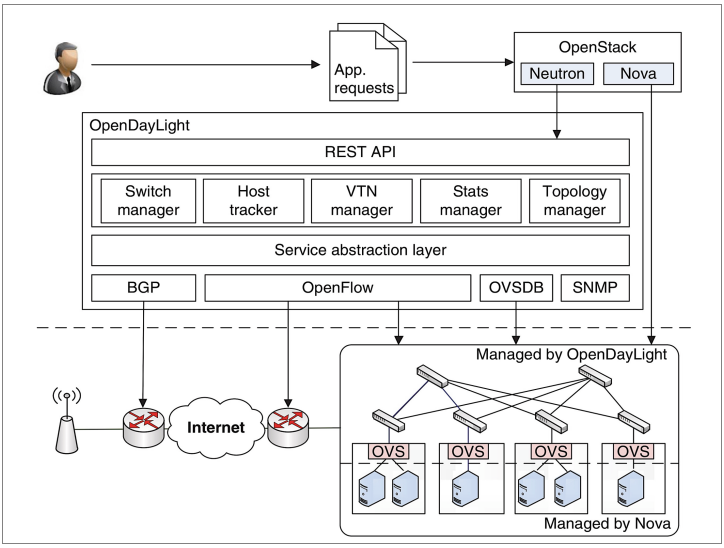

Fig. 9. Typical cloud computing workflow

We improved over the state-of-the-art works, in these aspects: (i) we have implemented a stateful/security-aware SDN dataplane and hence some light-weight detection/computation functions are offloaded to the switches for in-line processing (ii) We implemented the OvS data plane stack using the "Data Plane Development Kit (DPDK)" that consist of Network Interface Card (NIC) drivers/libraries/APIs for high-speed packet processing. Due to fastpath kernel processing and acceleration with DPDK, the flow-analysis pipeline processing throughput is significantly higher in the switch (iii) As a consequence of above two improvements, the processing power & throughput of network-ports of controller is freed up for other functions.

5 Preliminary Experimentation

In this domain of SDN-OpenStack Integration for Cloud platforms, formal specification or benchmarks aren't published in the open and vendors haven't published performance/validation. Hence, deriving from various literature study, we designed an evaluation strategy with security perspective and enumerated a set of network characteristics and Key Performance Indicators. The key objectives of our evaluation are:

- Can detect and defend large distributed attacks ("100 s of Gbps")
- Responsiveness, with acceptable performance hit for the legit users or applications
- To cope with rapidly changing dynamic attack patterns and scale with the network

A Cloud computing cluster (Fig. 10) with 4 machines - controller, network & 3 compute nodes. For evaluation and comparative study each compute node is loaded with different network hypervisor switch – 1. legacy Linux Bridge Firewall (LB FW), 2. native OvS Firewall module and 3. CloudSDN OvS Security modules. We used TCP background traffic, as the majority of traffic (99.1%) in data centers is TCP and about "64% DDoS attacks includes TCP-SYN, DNS and NTP amplification traffic".

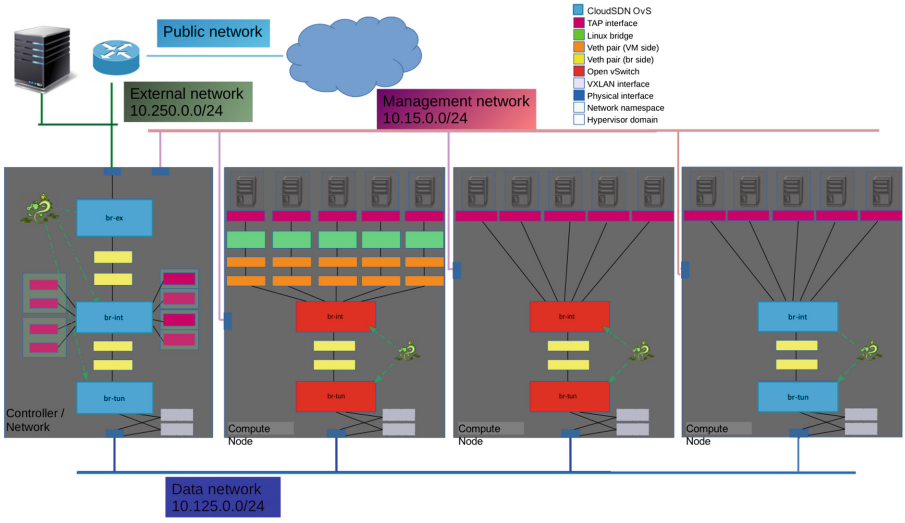

Fig. 10. SDN enabled OpenStack testbed

5.1 Comparison with Traditional Snort-Iptables IDS/IPS

- *IPS Efficiency:* Fig. 11(a) "total-packets processed/sec" varying attack rates. Traditional IPS drops packets and efficiency decreases as rate increases (12K pps to 0 at 37K pps). CloudSDN IPS sustains the packet processing under the same attack.
- *IDS efficiency*: Fig. 11(b) As the attack rate increases the Traditional IDS efficiency drops as the DoS attack floods the network, fills up the "Iptables Queue", saturates and eventually all packets are dropped. In CloudSDN IDS withstands large attack.
- To observe the impact on throughput of normal traffic, in the same network, we setup DoS attack generators towards a Server/VM inside the cluster and target to saturate the gateway switch. CloudSDN maintains throughput of benign traffic while dropping the attack traffic at the switch. With the traditional IPS, the normal traffic throughput is significantly impacted to a point of complete shutdown (Fig. 12a).

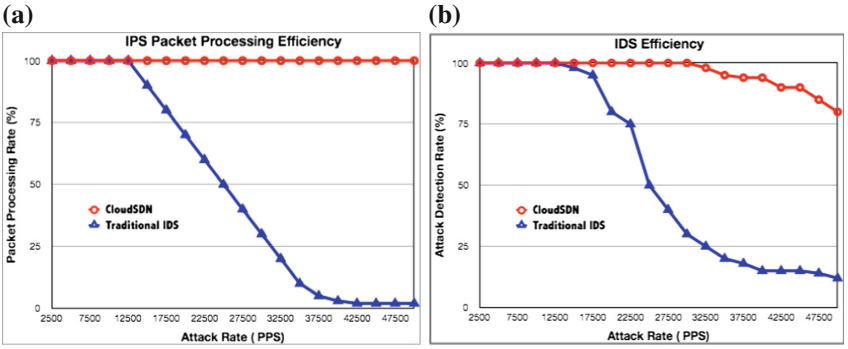

Fig. 11. (a) IPS processing (b) IDS detection efficiency

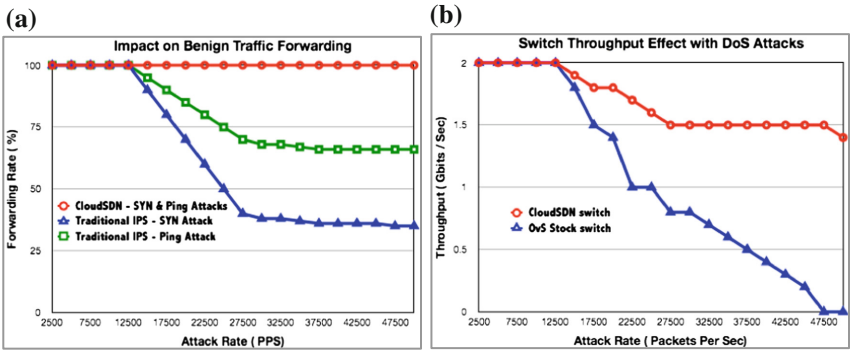

Fig. 12. (a) Impact on benign traffic (b) Switch throughput

5.2 Comparison with Classic OpenFlow SDN

- *Throughput Effect:* Fig. 12(b) Using various 'burst-intervals', UDP attacks (NTP amplification) is fired towards VMs in the cluster passing through switch. We ran normal FTP on another pair of VM's in the cluster. The results show that CloudSDN throughput is sustained at around 2 Gbps at high attack rates, Classic SDN switch degrades as the attack rate is increased to drop down to zero.
- *Mitigating DNS Amplification attacks:* Fig. 13(a), DNS server database/"zone file" and "*Scapy* as DNS query generator" are setup. DNS tool spoofs 'victim's IP address and floods the targeted DNS server(VM) with 80-byte length QUERY and DNS server responds for each query with REPLY of 4 kb to victim. Thus, the victim is flooded with unsolicited DNS responses saturating the network & CPU on that VM. We evaluated the detection of DNS attack under various scenarios. In 100 iterations, CloudSDN detects & blocks the DNS flooding attack in 4–12 s).
- *Attack Response:* Fig. 13(b) Shows attack-traffic saturating the link and consequently disrupting the legitimate normal traffic. CloudSDN detects in less than 3 s and mitigates (dropping the attack packets) in the data plane itself.

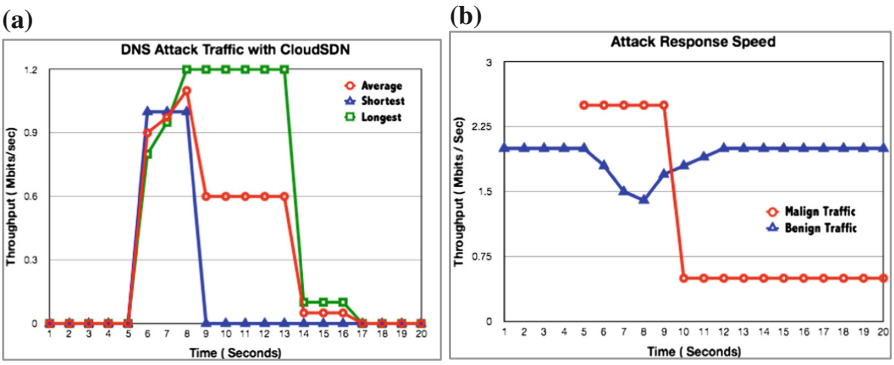

Fig. 13. (a) DNS amplification attack (b) Attack response

Table 2. Latency and packet loss

Metric	Flow table entries duration			
	50 s	100 s	500 s	1200 s
RTT	110.67 ms	56.34 ms	5.28 ms	3.91 ms
Packet loss	3%	1%	0%	0%

Table 3. Memory utilization

Instances/Node	Linux BR FW	OvS classic FW	CloudSDN FW
1	9.2%	22.6%	24.4%
4	14.8%	26.1%	32.2%
8	26.2%	34%	36.2%
16	40%	48.4%	56.8%

– *Latency/Packet-loss:* From Table 2 tests, (a) When DDoS attack begins, number of PACKET_IN events on SDN OpenFlow channel varies between 200 and 2000. Under DDoS attack (i) without security-scheme, Ave. RTT is >100 s and packet loss 100% (ii) With security scheme (a) small duration-the RTT gets affected. (b) longer-duration-the RTT is normal and 0% packet loss. Benign traffic is impacted by attack traffic, but it recovers under longer duration.

5.3 Comparison of Linux Bridge/Classic OpenStack/CloudSDN

To evaluate software nature adopted by CloudSDN, we evaluated the Key Performance Indicators (KPI) computation at various load conditions and networking at various traffic conditions. We set up 5 VM instances/compute-node and sufficient number of CPU cores and memory. We ran "30 tests of netperf TCP STREAM. per node", with one VM instance of server and other clients.

– *CPU, Memory Usage:* Fig. 14(a) shows all three mechanisms (legacy LB, native OvS, CloudSDN) consume similar CPU/memory resources. However, in Table 3, we see that SDN OvS mechanisms consume more memory compared with legacy LB, because of SDN/OvS OpenFlow pipeline tables. As the VMs/node increase, the memory utilization of all 3 mechanisms normalized to a level equal to or even lower than that is used by legacy LB.

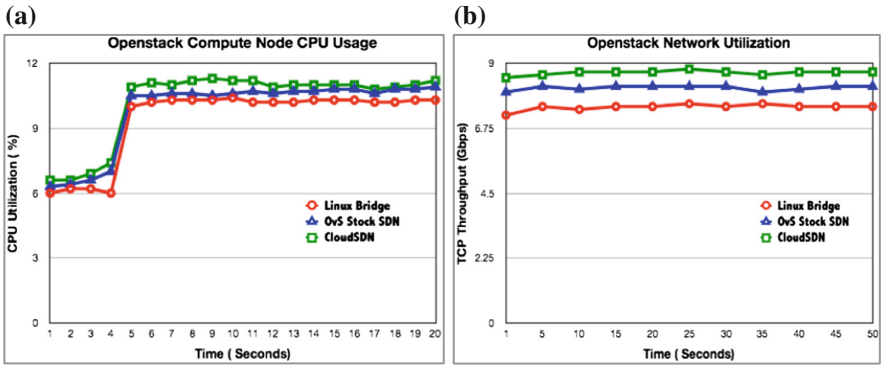

Fig. 14. (a) CPU usage (b) TCP throughput

– *TCP Throughput, Latency:* By varying the number of clients/node and external clients flooding a single server, higher sustained TCP throughput is seen with OvS based firewall than with Linux Bridge approach. This proves that OvS is optimal in the OpenStack Cloud applications. In the long run, the total aggregate throughput for all TCP flows gets closer to the maximum available bandwidth in the network interface. In Fig. 14(b) 4 clients send traffic to 1 server, total TCP throughput is almost 8.4 Gbps. When the number of clients increase, the total aggregated flows utilize the full bandwidth.

6 Conclusions and Future Work

Our work shows potential for software defined networking in achieving one of the paradigm visions i.e. "to provide a programmable capability for global view of the security incidents and respond rapidly", especially in large spatially distributed Cloud networks. In this paper, we presented the integrated view of Cloud computing and SDN under various scenarios especially in the presence of network attacks and DDoS/Botnet attacks. We proposed the CloudSDN Framework, that has the notion of multi-plane collaborative security monitoring, threat analytics, attack detection/mitigation in the emerging SDNFV enabled Cloud computing large-scale applications. We have also contributed key extensions and plugins to OpenStack/SDN based Cloud platform, especially the network architecture, to solve some of the open issues in reliability and security. We demonstrated the feasibility with DDoS/botnet defense applications using novel anomaly detection methods in the control plane and co-operative light weight monitoring/trigger/mitigatory NF mechanisms in the data plane. Our framework is one of the early works to combine the NFV and SDN-enabled Cloud platform, NF service chaining within in the SDN data plane, leading to speed-up and agility, security-awareness in the Cloud networks. CloudSDN design is platform agnostic, extensible to heterogenous network models for any large Cloud applications in IoT, 5G, Industry 4.0.

References

1. Rubio-Loyola, J., et al.: Scalable service deployment on software-defined networks. IEEE Commun. Mag. **49**(12), 84–93 (2011)
2. Hu, F., Hao, Q., Bao, K.: A survey on software defined networking (SDN) and openflow: from concept to implementation. IEEE Commun. Surv. Tutorials **16**(4), 2181–2206 (2014). vol. PP, no. 99, p. 1
3. Han, B., et al.: Network function virtualization: challenges and opportunities for innovations. IEEE Commun. Mag. **53**(2), 90–97 (2015)
4. Singh, A., Chatterjee, K.: Cloud security issues and challenges: a survey. J. Netw. Comput. Appl. **79**, 88–115 (2017). https://doi.org/10.1016/j.jnca.2016.11.027
5. Singh, S., et al.: A survey on cloud computing security: issues, threats, and solutions. J. Netw. Comput. Appl. **75**, 200–222 (2016)
6. Kanagasabapathi, K., Deepak, S., Prakash, P.: A study on security issues in cloud computing. In: Suresh, L.P., Panigrahi, B.K. (eds.) Proceedings of the International Conference on Soft Computing Systems. AISC, vol. 398, pp. 167–175. Springer, New Delhi (2016). https://doi.org/10.1007/978-81-322-2674-1_17
7. Scott-Hayward, S., O'Callaghan, G., Sezer, S.: SDN security: a survey. In: 2013 IEEE SDN for Future Networks and Services (SDN4FNS), pp. 1–7, November 2013
8. OpenStack. https://www.openstack.org/
9. Open Virtual Network Project. https://www.openvswitch.org/
10. Son, J., Buyya, R.: A taxonomy of Software-Defined Networking (SDN)-enabled cloud computing. ACM Comput. Surv. (CSUR) **51**(3) (2017). Article. 59, 36 Pages. https://doi.org/10.1145/3190617
11. Banikazemi, M., et al.: Meridian: an SDN platform for cloud network services. IEEE Commun. Mag. **51**(2), 120–127 (2013)
12. Du, X., Lv, Z., Wu, J., Wu, C., Chen, S.: PDSDN: a policy-driven SDN controller improving scheme for multi-tenant cloud datacenter environments. In: IEEE International Conference on Services Computing, pp. 387–394 (2016)
13. Mayoral, A., et al.: SDN orchestration architectures and their integration with Cloud Computing applications. Opt. Switching Netw. **26**, 2–13 (2017)
14. Giotis, K., et al.: Combining OpenFlow and sFlow for an effective and scalable anomaly detection and mitigation mechanism on SDN environments. Comput. Netw. **62**(5), 122–136 (2014)
15. Kumar, S., Kumar, T., Singh, G., Nehra, M.S.: Open flow switch with intrusion detection system. Int. J. Sci. Res. Eng. Technol. **1**(7), 1–4 (2012)
16. Jafarian, J.H., Al-Shaer, E., Duan, Q.: Openflow random host mutation: transparent moving target defense using software defined networking. In: The Workshop on Hot Topics in Software Defined Networks, pp. 127–132. ACM (2012)
17. Zanna, P., O'Neill, B., Radcliffe, P., et al.: Adaptive threat management through the integration of IDS into software defined networks. In: Network of the Future, pp. 1–5. IEEE (2014)
18. Xing, T., Xiong, Z., Huang, D., et al.: SDNIPS: enabling software-defined networking based intrusion prevention system in clouds. In: International Conference on Network and Service Management, pp. 308–311. IEEE (2014)
19. Chi, Y., et al.: Design and implementation of cloud platform intrusion prevention system based on SDN. In: IEEE International Conference on Big Data Analysis, pp. 847–852 (2017)

20. Shin, S., Gu, G.: Cloudwatcher: network security monitoring using openflow in dynamic cloud networks. In: 20th IEEE International Conference on Network Protocols, pp. 1–6 (2012)
21. Yan, Q., et al.: Software-defined networking (SDN) and distributed denial of service (DDoS) attacks in cloud computing environments: a survey, some research issues, and challenges. IEEE Commun. Surv. Tutorials **18**(1), 602–622 (2016)
22. Chowdhary, A., et al.: Dynamic game based security framework in SDN-enabled cloud networking environments. In: ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization (SDN-NFVSec 2017)
23. Foresta, F., et al.: Improving OpenStack networking: advantages and performance of native SDN integration. In: 2018 IEEE International Conference on Communications (ICC) (2018)
24. Yang, C.-T., et al.: Implementation of a real-time network traffic monitoring service with network functions virtualization. Future Gener. Comput. Syst. **93**, 687–701 (2018). https://doi.org/10.1016/j.future.2018.08.050
25. Krishnan, P., Najeem, J.S., Achuthan, K.: SDN framework for securing IoT networks. In: Kumar, N., Thakre, A. (eds.) UBICNET 2017. LNICST, vol. 218, pp. 116–129. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-73423-1_11