International Journal of Electrical and Computer Engineering (IJECE)
Vol. 9, No. 6, December 2019, pp. 5519~5528
ISSN: 2088-8708, DOI: 10.11591/ijece.v9i6.pp5519-5528

Enhanced IPFIX flow monitoring for VXLAN based cloud overlay networks

Osman Ghazali[1], Shahzada Khurram[2]
[1]InterNetWorks Research Laboratory, School of Computing, Universiti Utara Malaysia, Malaysia
[2]Department of Computer Science, the Islamia University Bahawalpur, Pakistan

Article Info

Article history:

Received Dec 20, 2018
Revised Apr 18, 2019
Accepted Jun 17, 2019

Keywords:

Cloud monitoring
Flow classification
Flow monitoring
IPFIX
Overlay networks

ABSTRACT

The demands for cloud computing services is rapidly growing due to its fast adoption and the migration of workloads from private data centers to cloud data centers. Many companies, small and large, prefer switching their data to the enterprise cloud environment rather than expanding their own data centers. As a result, the network traffic in cloud data centers is increasing rapidly. However, due to the dynamic resource provisioning and high-speed virtualized cloud networks, the traditional flow-monitoring systems is unable to provide detail visibility and information of traffic traversing the cloud overlay network environment. Hence, it does not fulfill the monitoring requirement of cloud overlay traffic. As the growth of cloud network traffic causes difficulties for the service providers and end-users to manage the traffic efficiently, an enhanced IPFIX flow monitoring mechanism for cloud overlay networks was proposed to address this problem. The monitoring mechanism provided detail visibility and information of overlay network traffic that traversed the cloud environment, which is not available in the current network monitoring systems. The experimental results showed that the proposed monitoring system able to capture overlay network traffic and segregated the tenant traffic based on virtual machines as compare to the standard monitoring system.

Corresponding Author:

Osman Ghazali,
InterNetWorks Research Laboratory,
School of Computing,
Universiti Utara Malaysia,
Kedah, Malaysia.
Email: osman@uum.edu.my

1. INTRODUCTION

The traditional cloud providers are struggling to keep up with new cloud computing requirements which include virtual machine migration, scalability and network isolation in a large cloud network environment. To manage a large and complex cloud network infrastructure requires the monitoring system to capture its state precisely [1]. Therefore, network architects should rethink their cloud designs and adopt simpler topologies and new control protocols to achieve better performance and operational agility in multi-tenant cloud networks.

Virtualization plays a vital role in the implementation of cloud computing. However, virtualization technologies add complexity to cloud providers and consumers. It leads to difficult in managing not only physical but virtual resource in cloud infrastructure [2-5]. The complexity of cloud network infrastructure requires root cause analysis of network problems and in-depth troubleshooting when a problem happens. Finding the cause of the problem involves searching into several layers including physical and virtual layers. Therefore, a reliable and real-time monitoring system is required for the cloud providers and consumers to understand the performance issues, and the causes of failure in cloud infrastructure [6]. Some organizations

may have mission-critical applications that are hosted on multiple clouds for high availability and workload sharing concerns. In such situations, monitoring is essential to significantly improve the performance of real-time applications and enable troubleshooting the multiple cloud network infrastructure [7]. This paper presents an enhanced IPFIX flow monitoring system for VXLAN based cloud overlay networks. The proposed monitoring system can capture the VXLAN packets in a cloud environment and differentiate them from other network traffic.

The remainder of this paper proceeds as follows. Section 2 describes the packet observation and selection mechanism. Section 3 explains the 6-tuple based flow processing and classification mechanism. Section 4 describes the enhanced IPFIX messaging system with flow export process. Section 5 presents the flow collection and traffic analysis process. Finally, Section 6 concludes the paper and briefs the future research direction in cloud monitoring.

Related Work, There have been many research and development efforts in the field of cloud monitoring and traffic analysis for the last few years. As a result, many tools have been introduced to meet various needs of cloud traffic measurement [8]. The monitoring systems for resource utilization in virtualized and large cloud environment have recently been proposed [9-10]. However, these mechanisms do not provide the complete picture of monitoring in respect of cloud overlay networks in a virtualized environment. Moreover, these mechanisms have not taken the dynamic nature of cloud overlay network performance into account. For classifying traffic into flows, L. Deri and F. Fusco [11] proposed the real-time cloud monitoring architecture based on network probes. However, it did not include the mechanism of overlay network traffic classification in the proposed architecture. In another research work, Mann et al. [12] proposed a flow-based network service monitoring solution for cloud infrastructure. However, it only analyzed flow monitoring protocols such as NetFlow [13] and sFlow [14] on physical and virtual switches for traffic analysis.

The IETF introduced IP Flow Information Export (IPFIX) protocol for exporting per-flow information. However, the IPFIX architecture described in [15-17] has limited functionalities and needs to be enhanced. The enhancement process should provide various functions like aggregation, filtering, or the modification of flow records for the means of saving system resources and providing processing tasks for collecting only traffic data of cloud overlay network.

Traditional IPFIX based Flow Monitoring Architecture Design, The architecture of the IPFIX based Flow Monitoring system consists of several stages that include packet observation and selection, flow metering and export process, flow collection process and traffic analysis. Figure 1 presents traditional IPFIX based flow monitoring architecture design and processing stages. All processing steps act on packets.

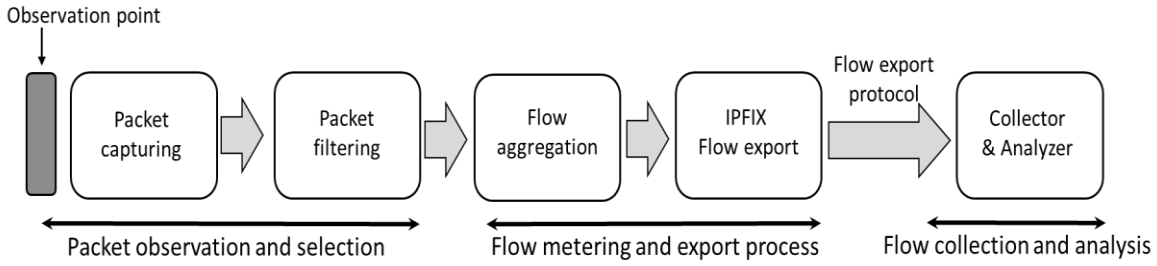

Figure 1. IPFIX based flow monitoring process architecture

2. PACKET OBSERVATION AND SELECTION

The packet observation and selection stage consists of packet capturing, packet filtering and packet sampling. Packets must be read on the line, and the packet observation is the first step of this architecture. Typically, packets capturing is performed on the Network Interface Card (NIC), which carries the packets. Before a packet moves to the receiving host memory, several checkings are performed on the card buffer such as checksum errors to ensure the packet is received in the original form. Due to the high traffic output, most of the packet capturing is performed on wired networks. It can range from a Local Area Network (LAN) to a Wide Area Network (WAN). Figure 2 presents the proposed enhanced IPFIX flow monitoring system design and processing stages.

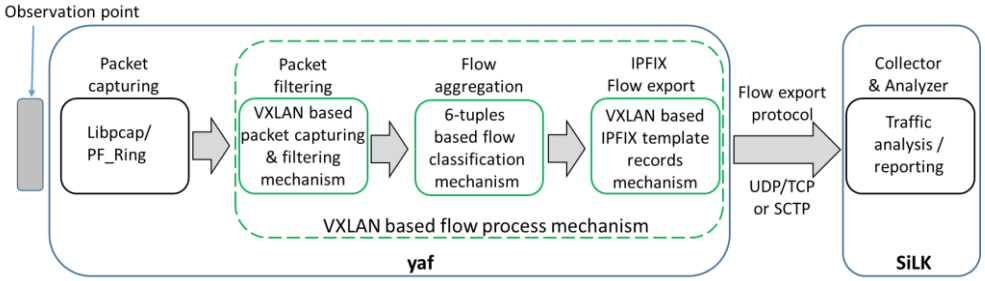

Figure 2. The proposed enhanced IPFIX Flow monitoring system

2.1. Packet capturing in virtual environment

In cloud environments, virtual network rapidly becomes more important due to the widespread deployment of virtual machines. Since the virtual environment is rapidly growing in cloud environments, packet capturing of virtual networks has become more common in cloud environments. Although things get a lot more complicated in virtual environments, the deployment of packet capturing device is very similar to deployment in the wired network. Thousands of virtual machines are interconnected to each other as a virtual network by using virtual switches to interconnect the virtual machines [18]. A virtual switch works the same as a hardware switch that supports virtual network taps and port mirroring. In virtual environments, traffic is captured through in-line mode or mirroring mode. Therefore, mirrored traffic forwarded to physical ports can be captured using a dedicated packet-capturing device outside the virtual environment.

2.2. Packet capturing process

The in-line mode has been selected for capturing the packets in high-speed cloud network environments. The development of a reliable monitoring architecture requires a full understanding of packet capturing process. Many applications programming interfaces (APIs) and libraries are available in the open source Linux environment. The most reliable library libpcap [19] is used for packet capturing. Since the operating system network stack is performed for general purpose networking, the libpacp library is used for handover of packets from the NIC to the packet capturing application. The overall packet capturing process depends on the system performance as pre-packet processing overhead is added during the process.

2.3. Packet filtering

Packet filtering is the technique that defines the actions performed on every single packet received from the observation point for the selection of particular packets. The role of packet filtering is defined in RFC 5475 as separating the packets with a specific property from those without it [20]. This step is adopted for selecting the packets that we are interested with, which is VXLAN (Virtual eXtensible LANs) [21] packets. Typically, this type of packet filtering requires property matched filtering technique. Whereby, a packet is selected if a specific field of a packet is equal to a specified value or inside a specified value range [20]. In order to design the filtering technique of cloud overlay packets, the complete structure of VXLAN packet format which is defined in RFC 7348 [21] has to be clearly understood.

2.4. VXLAN based packet filtering mechanism

The cloud overlay is a new technology in which packets are encapsulated in the overlay network. Therefore, the most critical step is retrieving and sampling the VXLAN packets. The proposed technique inspect the captured packets and select only VXLAN packets. All packets are read directly from the observation point with time stamped. Packets are inspected based on the header instead of the whole payload inspection to reduce overhead and minimize the load at the packet selection stage. The selected packet becomes an element of the output packet stream.

For the selection of VXLAN packets, the following steps have to be performed on each arriving packet. This is done without dropping or altering even a single packet [22].

- In the first phase, each arriving raw packet requires packet size check as the minimum. The VXLAN packet size including all headers is 72 bytes without payload size. If the packet size is less than 72 bytes then move the packet to the initial phase. Otherwise, forward the packet to the next phase.
- In the second phase, extract the outer IPv4 header fields and check the packet protocol. By default, the VXLAN packets use UDP for communication. If the protocol is not UDP then move to the initial phase. Otherwise, forward the packet to the next phase.

- In the third phase, open the VXLAN header and check the 5th bit. The valid VXLAN packet 5th bit must be on out of first eight bits. If the 5th bit is not on, then move to the initial phase. Otherwise, forward the packet to the next phase.
- Once all the checks are performed successfully, then select the packet from the input stream, and packet count is incremented by one to account for the just arrived packet. Then, the selected packet will be forward to the flow processing.

3. FLOW PROCESSING STAGE

The flow processing stage consists of packet aggregation, flow cache, flow selection and transport protocol. After the selection of filtered packets, cloud overlay packets will be aggregated into network traffic flows for temporarily stored in the flow cache. Network traffic flow is described as a sequence of packets between two endpoints based on the key fields. Typically, a flow pattern is based on 5 tuples which represent the set of five different key values as described in Figure 3. It includes a source IP address and source port number, destination IP address and destination port number and the protocol in use [15]. If the values of the key fields of the new captured packet match the existing flow, then the packet will be added to the existing flow and information is updated accordingly. On the other hand, if the values of the key fields do not match any of the existing flow, then a new flow will be generated and stored in the flow cache. Flow generation and update steps can be repeatedly performed for flow aggregations. However, typical 5-tuple based flow pattern cannot fulfill the requirements of VXLAN based flow generation. As overlay network traffic involved more layers; therefore additional fields required to identify the encrypted tunnel traffic for flow generation.

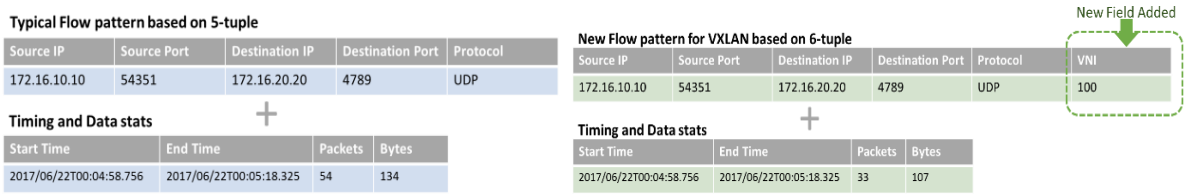

Figure 3. Typical flow pattern based on 5-tuple and new flow pattern for VXLAN based on 6-tuple

3.1. VXLAN based 6-tuple flow

The VXLAN based flow classification requires more than 5-tuple fields. The VXLAN based cloud overlay network traffic uses a unique network identifier (VNI) value for communication between two tenants, where each tenant creates a dynamic overlay network for communication with other tenants. The VNI field has 24 bits and can identify a maximum of 16 million VXLAN segments. The key step to monitoring VXLAN based overlay network traffic is to identify the VNI value. Therefore, a new flow pattern called VXLAN based 6-tuple flow is introduced. It represents a set of six field values. A new VNI key field is added on the traditional flow key pattern which makes it a unique 6-tuple VXLAN based flow pattern. The fields include a source IP address and source port number, destination IP address and destination port number, protocol and a newly added field VNI. If any of field change, then a new flow will be generated. Figure 3 presents the new VXLAN based 6-tuple flow pattern. Each separate flow has an entry associated with the non-key fields including flow start time, end time, total number of packets and total bytes. All active network traffic flows information is maintained in the flow cache.

3.2. Flow classification

Flow classification is used to map each input packet to its respective flow. This operation is necessary as the processing of each input packet is done at VXLAN based packet filtering mechanism. After the filtering, each packet that arrives in the flow classifier has the relevant 6-tuples header fields extracted. The 6-tuples header field values in the arriving packet are compared with the existing flow entries. If there is no matching entry found, then a new flow will be created based on the 6-tuples VXLAN pattern. In the event of existing entry matched, the existing flow entry is updated with information from this newly arrived packet and several fields are also updated. The packet count is incremented by one to account for the packet that just arrived. The byte count is incremented by the number of bytes of data present in the packet. The timestamp is also updated with the current time to indicate that a new packet just arrived for this flow. The timestamp is used to age out old flow entries. The pseudocode for the implementation of flow classification mechanism based on 6-tuples pattern is given in Algorithm 1.

Algorithm 1 VXLAN based flow pattern algorithm

```
OPcount is initialized to zero
Call the flow flush timer and initialized to zero
OPScount is initialized to zero     /* Overlay Packet size*/
        Arrival of VXLAN packet vPi
        check the 6-tuple value of the packet vPi
         if ( No VNI based flow-group-pattern found) then
         Make new flow-group-pattern [Source, Destination IP, PortIn, PortOut, VNI]
                    If ( Group-pattern is seen) then
                    Send/Add packet to existing flow-group
                  Increment OPcount
                  Increment OPScount
                  End if
          Else
                    Check the flow flush timer
                  If ( flush timer expire) then
                    Make new flow-group-pattern
                    End if
Else
                    No action
End if
```

4. IPFIX MESSAGE AND FLOW EXPORT

The simplified IPFIX message format consists of version number, message length, export time, sequence number and domain source ID and different set of records [15]. IPFIX is an open source standard, which is defined in RFC7011 by IETF.

4.1. VXLAN based IPFIX template

The IPFIX template is based on a set of fields that can be exported to flow records that are named information elements. The detail of IPFIX information elements available at Internet Assigned Numbers Authority (IANA), which is responsible for maintaining a standard list of IPFIX information elements [23]. The IPFIX information elements can be defined from the data link layer to the application layer. However common information elements belong to the network and transport layer. On the other hand, IPFIX also supports private information elements. The required VXLAN based IPFIX information elements are defined in Table 1. Except for VNILabel information element, all of the other information elements are already defined in IANA standard list of IPFIX information elements. As per the research requirement of cloud overlay network monitoring, a new 3 bytes of information element named VNILabel ia added as a private information element.

Table 1. Vxlan based IPFIX information elements

ID	Name	Description	Byte size
152	flowStartMilliseconds	Timestamp of the flow's first packet.	8
153	flowEndMilliseconds	Timestamp of the flow's last packet.	8
8	sourceIPv4Address	IPv4 source address in the packet header.	4
12	destinationIPv4Address	IPv4 destination address in the packet header.	4
7	sourceTransportPort	Source port in the transport header	2
11	destinationTransportPort	Destination port in the transport header.	2
10	ingressInterface	Interface address where packets in	4
14	egressInterface	Interface address where packets out	4
2	packetDeltaCount	Number of packets for the flow	8
1	octetDeltaCount	Number of bytes for the flow	8
4	protocolIdentifier	IP protocol number in the packet header	1
1001	VNILabel	VXLAN network identifier value in the packet header.	3

In addition, a new VXLAN based IPFIX template based on Table 1. information elements is constructed. For example, Template ID 203 in Figure 4 presents the VXLAN based set of information elements in IPFIX message.

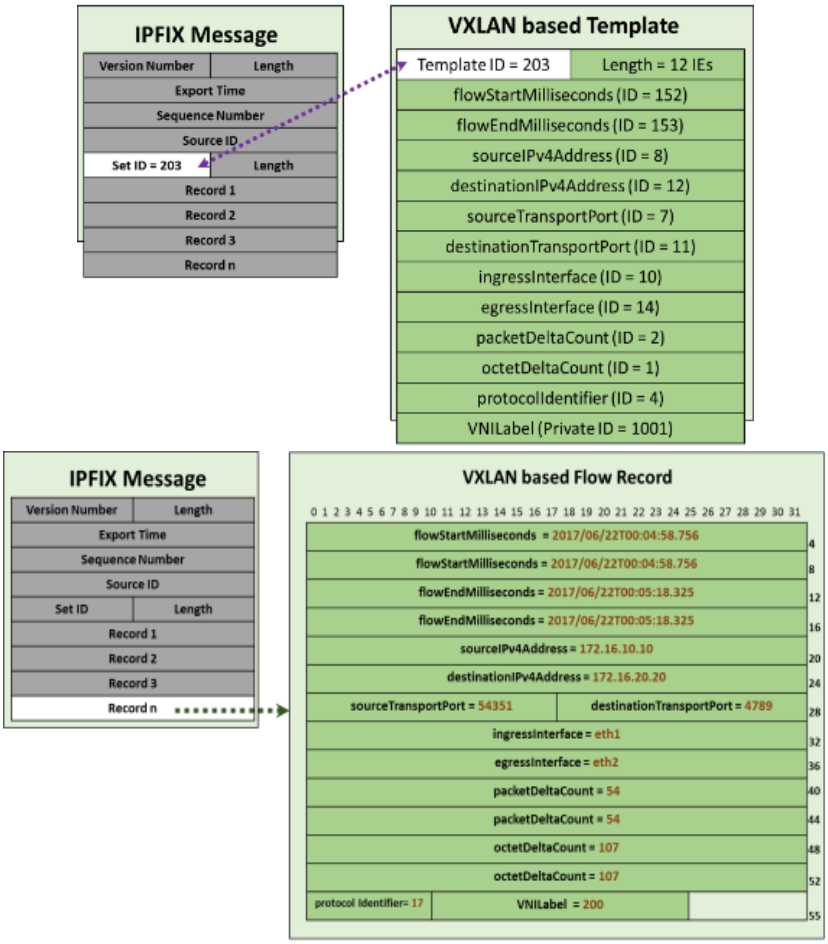

Figure 4. VXLAN based template flow record in IPFIX message

4.2. Data records

Flow entries are maintained in the flow cache tables for a certain period. After the flow entry timeout, whether it is idle or active timeout, flow data is forwarded to a process for building an IPFIX message. An IPFIX message is constructed with template ID that in this study VXLAN based template and flow data are stored in IPFIX records. Moreover, data sets are used in IPFIX to carry data records to be exported to the collector. A data set is based on many different data records, and each data record has flow properties based on the template. Figure 4 presents multiple flow records in the IPFIX message and also presents the VXLAN based flow data record. A 56-bytes required for construction of VXLAN based flow record data set in IPFIX message.

4.3. Flow export process

IPFIX can support multiple transport protocol for flow export [15]. The flow export process defines how to carry VXLAN based IPFIX messages via multiple transport protocols from flow export process to flow collector for further data analysis. After the construction of VXLAN based IPFIX message, UDP has been selected as the transport protocol for exporting the flow record to the flow collector. UDP carries no overhead, and it is a widely deployed transport protocol for flow export process.

4.4. Flow collection and traffic analysis

The flow collector is responsible for collecting flow data which is exported by the flow exporter, and this is an essential part of the flow monitoring system. It works like reception and received data from multiple flow exporters and store them according to the requirement for further network traffic performance analysis. Flow data generally does not contain any payload as the content of end user communications is protected. SiLk [24] is selected as a flow collector. SiLk understands IPFIX message sampled data and supports all transport protocols.

5. SIMULATION AND EXPERIMENT RESULTS

In order to build a cloud overlay network environment, a cloud underlay network was developed. A topology that represents VXLAN based cloud network environment as presented in Figure 5 was designed for the simulation. The topology consisted of three servers on a virtualized hypervisor with different network segments on underlay networks. Two servers, namely Kuala Lumpur and New York, were used for multi-tenancy environment for the virtual machine to virtual machine communications. The third server acted as the router to connect both servers for communication with each other in the underlay network. Linux Ubuntu 16.04 Server edition with minimum packages was installed on all servers. In addition, two different IP network segments were created. The 172.16.10.0/24 was for Kuala Lumpur server, and the 172.16.20.0/24 was for New York server, and both were connected to server-3 (Network Cloud) which performed routing service. Figure 5 demonstrates detail of the underlay network connectivity to all server machines.

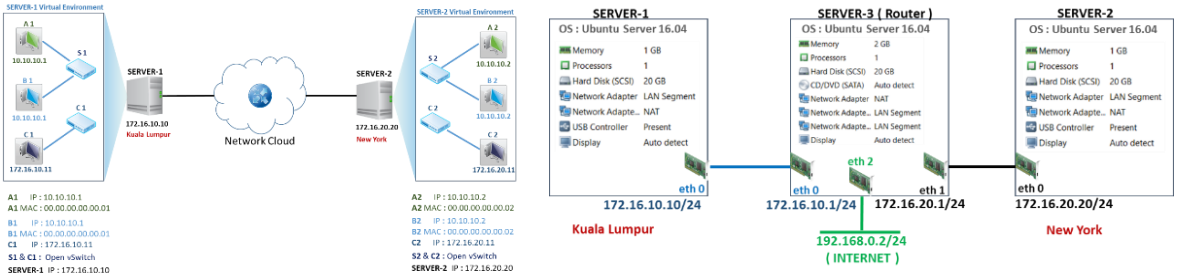

Figure 5. Cloud overlay network environment with underlay network detail

5.1. Dataset for simulation

For cloud overlay network monitoring mechanism, the experiment used Mininet for simulation. Traffic between different virtual machines was generated by the well-known network tool iperf [25] to monitor the performance measurement under different conditions. Figure 5 illustrates the traffic generated between different network segments based on the following dataset.

Transmission duration: 60 minutes
Protocol: ICMP
Virtual Machine -A1 → Virtual Machine -A2 Sending rate: 200 bytes/sec
Virtual Machine -B1 ← Virtual Machine -B2 Sending rate: 100 bytes/sec
Virtual Machine -C1 → Virtual Machine -C2 Sending rate: 200 bytes/sec.

The simulation was performed on a Linux based virtual environment, using Mininet [26] simulation tool, virtual machines, Open Vswitches [27] and different network segments were created for cloud overlay networks environment. A plugin was developed and compiled with the open source tool yaf [28] based on the proposed algorithms to enhance the existing IPFIX flow monitoring mechanisms for VXLAN based cloud overlay networks.

5.2. Experiment results

Figure 6 shows the experiment results of the standard monitoring tool. This tool captures the total number of packets and bandwidth but could not identify the VXLAN based tunnel traffic in a virtual cloud network environment. On the other hand, Figure 7 shows the results of the proposed VXLAN based monitoring system, which manages to capture VXLAN packets and differentiated the traffic based on Virtual Network Identifier (VNI) and other traffic. VNI 100 represents the captured tunneled traffic between virtual machines A1 and A2. Similarly, VNI 200 represents the captured tunneled traffic between virtual machines B1 and B2 with time and date stamp duration, and the remaining traffic is shown as other traffic. However, the standard monitoring tool unable to capture the virtual tunnel traffic and could only identify one traffic or total traffic. On the other hand, the proposed mechanism can capture the live tunnel traffic and also can identify VXLAN packets and distinguish the traffic between Virtual Network Identifier (VNI) and other traffic in a high-speed cloud virtual network environment.

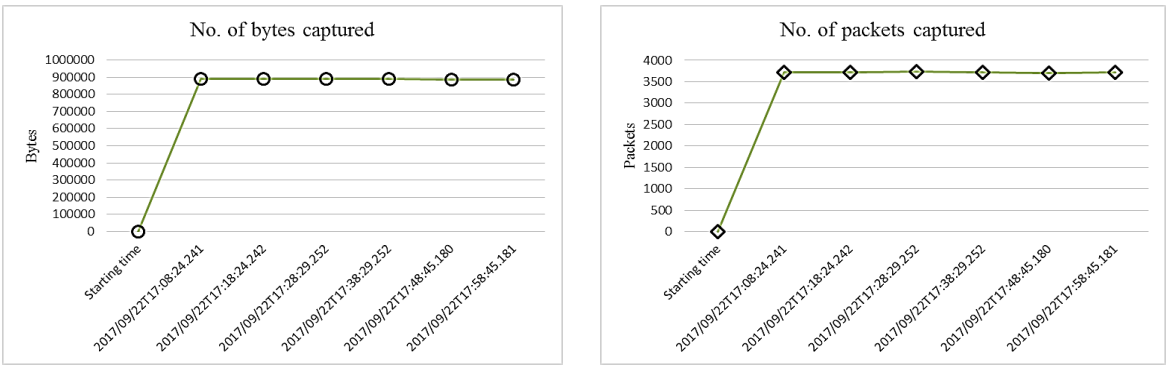

Figure 6. Standard flow monitoring (bytes and packets captured)

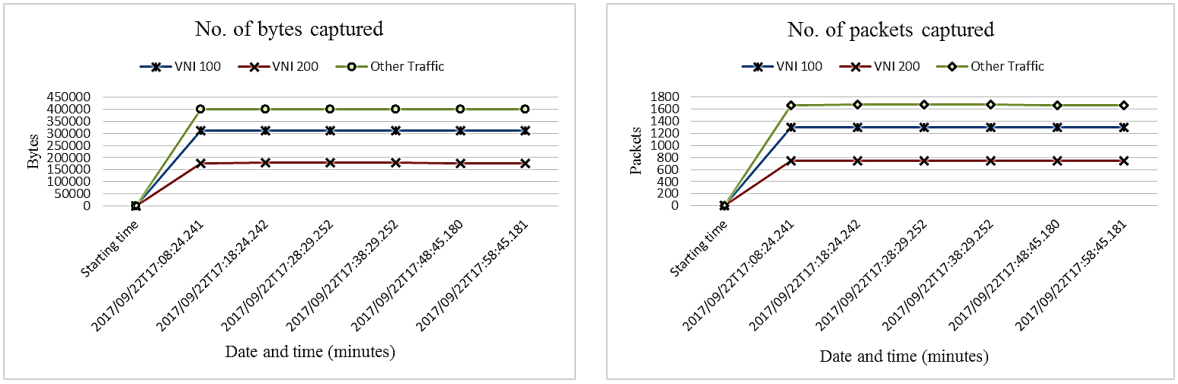

Figure 7. VXLAN based enhanced IPFIX flow monitoring (bytes and packet captured)

6. CONCLUSION

The study proposed VXLAN based enhanced IPFIX flow monitoring system for cloud overlay networks. A flow classification mechanism based on 6-tuples pattern and VXLAN based flow record IPFIX message to identify the virtual traffic was proposed. The proposed system can capture the invisible cloud overlay network traffic to identify, track, analyze and monitor the performance of cloud overlay network services. As the performance of a system is dynamic and depends on multiple parameters, the proposed system is capable to continuously tracking, quantifying and updating the monitoring results. The proposed monitoring system can provide network operators with detailed information about the traffic traversing a linked and related information especially suited to the modern cloud-scale data center. It would help cloud network operators and users to quickly and proactively resolve any network-based performance issues with end-to-end visibility and actionable insights.

ACKNOWLEDGMENTS

This research is funded by the Fundamental Research Grant Scheme (FRGS) 13144 (2014). The authors would like to thank the Ministry of Education Malaysia and Universiti Utara Malaysia for supporting and funding this research.

REFERENCES

[1] A. Viratanapanu, *et al.*, "On-demand fine grain resource monitoring system for server consolidation," *Beyond the Internet- Innovations for Future Networks and Services*, pp. 1-8, 2010.
[2] N. Chandrakala and B. Rao, "Migration of Virtual Machine to improve the Security in Cloud Computing," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 8(1), pp. 210-219, 2018.
[3] A. Buchade and R. Ingle, "Ternary Tree Based Approach For Accessing the Resources by Overlapping Members in Cloud Computing," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 7(6), pp. 3593-3601, 2017.

[4] S. Deshpande and R. Ingle, "Preferences Based Customized Trust Model for Assessment of Cloud Services," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 8(1), pp. 304-325, 2018.
[5] J. Shao, *et al.*, "A Runtime Model Based Monitoring Approach for Cloud," *2010 IEEE 3rd International Conference on Cloud Computing*, pp. 313-320, 2010.
[6] D. Zissis and D. Lekkas, "Addressing cloud computing security issues," *Futur. Gener. Comput. Syst.*, vol. 28, pp. 583-592, 2012.
[7] J. Schad, *et al.*, "Runtime measurements in the cloud," *Proc. VLDB Endow.*, vol. 3, pp. 460-471, 2010.
[8] S. Khurram, *et al.*, "A Survey of Cloud Monitoring: High Level, Low Level, Underlay and Overlay," *Netapps2015*, pp. 1-7, 2015.
[9] S. Clayman, *et al.*, "Monitoring virtual networks with Lattice," *2010 IEEE/IFIP Network Operations and Management Symposium Workshops*, pp. 239-246, 2010.
[10] J. S. Ward and A. Barker, "Varanus: In Situ Monitoring for Large Scale Cloud Systems," *IEEE 5th International Conference on Cloud Computing Technology and Science*, pp. 341-344, 2013.
[11] L. Deri and F. Fusco, "MicroCloud-based network traffic monitoring," *IFIP/IEEE International Symposium on Integrated Network Management (IM)*, 2013.
[12] V. Mann, *et al.*, "Living on the edge: Monitoring network flows at the edge in cloud data centers," *Fifth International Conference on (COMSNETS)*, pp. 1-9, 2013.
[13] "NetFlow," 2018, [Online], Available: https://www.cisco.com/c/en/us/products/ios-nx-os-software/iosnetflow/index.html.
[14] S. Panchen, *et al.*, "InMon Corporation's sFlow: A Method for Monitoring Traffic in Switched and Routed Networks," *RFC Editor*, 2001.
[15] B. Claise, *et al.*, "Specification of the IP Flow Information Export Protocol for the Exchange of Flow Information," *RFC 7011*, 2018, [Online], Available: https://tools.ietf.org/html/rfc7011.
[16] T. Zseby, *et al.*, "Requirements for IP Flow Information Export (IPFIX)," *IETF RFC 3917*, 2004, [Online], Available: http://tools.ietf.org/html/rfc3917.
[17] T. Zseby, *et al.*, "IP Flow Information Export (IPFIX) Applicability," *RFC 5472 Internet Engineering Task Force*, 2009.
[18] B. Pfaff, *et al.*, "Extending Networking into the Virtualization Layer," *Proceedings of the 8th ACM SIGCOMM*, 2009.
[19] S. M. V. Jacobson and C. Leres, "libpcap: Packet capture library," Berkeley, CA, Lawrence Berkeley Laboratory, 2009.
[20] T. Zseby, *et al.*, "Sampling and filtering techniques for IP packet selection," *RFC 5475 (Proposed Standard) Internet Engineering Task Force*, 2009.
[21] M. Mahalingam., "VXLAN: A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks," *RFC 7348 Internet Engineering Task Force*, 2014.
[22] S. Khurram and O. Ghazali, "Design and Development of VXLAN Based Cloud Overlay Network Monitoring System and Environment," *Information Technology – New Generations*, Springer, vol. 738, pp. 141-147, 2018.
[23] "IP Flow Information Export (IPFIX) Entities," 2013, [Online], Available: https://www.iana.org/assignments/ipfix/ipfix.xml.
[24] "SiLK," *(CERT NetSA) Carnegie Mellon University*, 2018, [Online], Available: https://tools.netsa.cert.org/silk/.
[25] "iPerf - The TCP, UDP and SCTP network bandwidth measurement tool," 2018, [Online], Available: https://iperf.fr/.
[26] "Mininet: An Instant Virtual Network on your Laptop or PC – Mininet," 2018, [Online], Available: http://mininet.org/.
[27] "Open vSwitch," 2017, [Online], Available: http://openvswitch.org/.
[28] "YAF - Yet Another Flowmeter," 2018, [Online], Available: https://tools.netsa.cert.org/yaf/.

BIOGRAPHIES OF AUTHORS

Osman Ghazali is an Associate Professor and the Deputy Dean of School of Computing, Universiti Utara Malaysia. Osman holds a Ph.D. degree in Information Technology (Networking) from Awang Had Salleh Graduate School, Universiti Utara Malaysia (AHSGS). He did his post-doctoral as a research scientist at the School of Engineering and Applied Science, Aston University (EAS) in 2012. In 2011, Osman was the Head of the Computer Science Department, School of Computing, Universiti Utara Malaysia. Prior to that, from 2009 to 2011, he was the Technical Chairperson at the University Teaching and Learning Center, Universiti Utara Malaysia. Dr. Osman has more than 100 publications as refereed book chapters and refereed technical papers in journals and conferences. He is the co-founder and senior member of the InterNetworks Research Laboratory. He is also a member of the IEEE and the ACM.

Shahzada Khurram is a Ph.D. candidate in the field of Computer Networks at Universiti Utara Malaysia. He is currently serving as an Assistant Professor in the Computer Science Department in Islamia University Bahawalpur, Pakistan. His research interests include Overlay networks, IoT and Blockchain Technologies.