A High Performance VoLTE Traffic Classification Method using HTCondor

Jonghwan Hyun*, Jian Li†, ChaeTae Im‡, Jae-Hyoung Yoo*, James Won-Ki Hong*†
*Department of Computer Science and Engineering, POSTECH, Pohang, Korea
Email: {noraki, styoo, jwkhong}@postech.ac.kr
†Division of IT Convergence Engineering, POSTECH, Pohang, Korea
Email: gunine@postech.ac.kr
‡Korea Internet & Security Agency, Seoul, Korea
Email: chtim@kisa.or.kr

***Abstract*—Voice-over-LTE (VoLTE) is a VoIP-based multimedia service which is provided using All-IP based LTE networks. VoLTE service was first commercialized by Korean telcos in 2012, and now more and more telcos are trying to adopt this technology. With the increased VoLTE service popularity, it is inevitable to have large VoLTE traffic volume (possibly degrading the service quality) and the potential attacks (possibly degrading the service reliability and availability) in the near future. Therefore, in order to avoid such potential issues, we need to perform thorough analysis on VoLTE traffic. As a first step, we propose a VoLTE traffic classification method and its distributed architecture. As the proposed classification method relies on Deep Packet Inspection (DPI) technique, it severely suffers from the large processing time and scalability issues. To overcome these issues, we further propose a distributed architecture for VoLTE traffic classification by adopting a high throughput computing framework - HTCondor. We performed a set of experiments using real-world traces captured from a commercial LTE core network, and have shown that with the proposed architecture, we can achieve up to 23.869 Gbps classification throughput which was almost 35 times faster than the system without distributed processing.**

***Keywords*—HTCondor, Traffic Analysis, VoLTE, RTP, SIP**

I. Introduction

With the growth of Long Term Evolution (LTE) service market and the increased popularity of smart devices, the mobile traffic volume has been growing rapidly in the past several years, and such trend will inevitably continue in coming years. In the worldwide, the mobile traffic volume is expected to grow at least 3 times in 2016 [1]. Moreover, among the overall LTE traffic, the voice and video traffic contributes the most (more than 75 %), as the large bandwidth that the LTE network can provide.

Voice call service was typically realized using a circuit switching method in the existing 2G and 3G networks. However, LTE network is an All-IP based network, and with which all services should be implemented over packet switching networks. Hence, VoLTE service was developed with considering this context, and through which voice and video call service can be provided. VoLTE service was first commercialized by Korean telcos in 2012, and now more and more telcos are trying to adopt it is inevitable to have large VoLTE traffic volume (possibly degrading the service quality) and the potential attacks (possibly degrading the service reliability and availability) in the near future. Therefore, in order to avoid such potential issues, we need to perform thorough analysis on VoLTE traffic.

Because VoLTE is a VoIP-based multimedia service, the underlying communication protocols of two services are almost identical. Consequently, it is non-trivial to classify the VoLTE traffic among LTE traffic using existing VoIP traffic classification methods [2][3][4][5][6]. To effectively perform the classification of the VoLTE traffic, we need to find out the different characteristics of two services (VoIP and VoLTE), and with which we can differentiate VoLTE traffic from VoIP traffic within LTE traffic. We summarized the differences of VoLTE and VoIP services as three parts,

- **Underlying Wireless Networks**: VoIP services are provided over various types of wireless networks (e.g., LTE, 3G, Wi-Fi, WiMAX), while VoLTE service is provided over the LTE network only.
- **Guaranteed QoS Level**: to provide guaranteed QoS to VoLTE service, mobile network operators logically separate the VoLTE traffic and non-VoLTE traffic using different bearer type. VoLTE traffic, which has the highest priority, is delivered over a dedicated bearer, while non-VoLTE traffic including VoIP traffic is delivered over default bearer with which only best effort QoS can be preserved.
- **Application Types**: all of the VoIP traffic over LTE network is generated by third party mobile applications (e.g., Skype, Viber and WeChat), while VoLTE traffic can only be generated by the phone dial application which is typically pre-installed by mobile network operators.

The first difference can be exploited when the VoLTE and VoIP services run over the different underlying wireless networks. Unfortunately, the targeted VoLTE and VoIP traffic is captured from the identical network (e.g., LTE core network); hence, we cannot design the classification method by taking into account the first difference. When it comes to the second difference, since only VoLTE traffic is delivered over dedicated bearer, it is reasonable to exploit the bearer type as the VoLTE classification criterion. Since each User Equipment (UE) holds two bearer IDs (both for default and dedicated bearers), we have to properly manage the dedicated bearer IDs to differ the dedicated bearer type. However, managing the dedicated bearer IDs is challenging, due to 1) synchronizing

the dedicated bearer ID among multiple capture points requires the comprehensive knowledge on overall LTE core network, and 2) removing the expired dedicated bearer ID is a non-trivial problem. Therefore, in this work, we design our VoLTE traffic classification method by exploiting the third difference, which is the application level classification.

Similar to the VoIP service, VoLTE service relies on two types of communication protocols, SIP [7] and RTP [8]. We first analyze the traffic traces of VoIP and VoLTE applications, and then synthesize the VoLTE application payload signature based on our observation. With this signature we classify the VoLTE SIP traffic among all SIP traffic with high accuracy. Finally, we extract the meta-information from VoLTE SIP traffic, to further classify the VoLTE RTP traffic.

Payload signature based application classification method is typically relying on Deep Packet Inspection (DPI) technique; nevertheless, this technique induces high computational overhead [9]. Moreover, vast amount of existing traffic classification methods [10][11] are difficult to be parallelized because they perform flow-based classification, which induces dependency among parallelized workloads. Therefore, to overcome these problems, we propose a distributed architecture for VoLTE traffic classification workload in a way that increases the classification speed.

The remainder of this paper is organized as follows: Section II presents related work on VoLTE and VoIP traffic classification and the distributed architecture for traffic analysis. Section III presents our VoLTE traffic classification method, and its performance for a single machine. A distributed VoLTE traffic classification architecture is proposed in Section IV, and its performance is evaluated in Section V. Finally, conclusion and future work are discussed in Section VI.

II. Related work

A. VoIP Traffic Classification

VoLTE uses SIP to establish a multimedia session, and uses RTP to transfer multimedia data. As VoLTE is an emerging service, there is merely little literature work has been done on classifying the VoLTE traffic among overall LTE traffic. However, both VoIP and VoLTE services rely on the same communication protocols (e.g., SIP and RTP), we can still get some insights by carefully analyzing the literature work on VoIP traffic classification methods.

The RTP traffic classification techniques fall into two categories. The first technique relies on the SIP payload which contains a set of RTP meta information including port number, and with which the RTP traffic can be easily classified with high accuracy. The second technique does not rely on inspecting SIP payload; instead, it requires pre-defined RTP header or payload signatures (e.g., L7-filter [19]). Birke and Sinam et al. in [3][4] proposed heuristic algorithms which are used to classify RTP flows instead of RTP packets for the purpose of further decreasing the computational complexity. In a summary, the first technique inspects both SIP payload as well as RTP header, which induces large computational overhead, yet provides very high classification accuracy. Whereas, the second technique inspects RTP header/payload only, hence the computational overhead is relatively moderate, yet shows poor classification accuracy compared to the former technique. In this work, we choose and further extend the first technique to classify the SIP and RTP packets of VoLTE service. The reasons are: 1) the rich information (e.g., IP, codec information, user-agent and etc.) contained in SIP payload can be utilized as a solid basis for differentiating the VoLTE and VoIP service; 2) it provides much higher accuracy compared to the second technique; and 3) we can overcome the deficiency of the first technique by adopting the distributed architecture to realize the classification parallelism.

B. Distributed Traffic Analysis

Traffic analysis requires vast amount of computing power, especially for DPI technique. Most of the commercial DPI devices require specialized hardwares; nevertheless, expensive price burdens the adoption of those devices. An alternative solution is to adopt a large scale cluster which is comprised of cheap commodity servers. Due to the economic and scalability merits, the cluster based DPI approach becomes more and more popular recently. However, parallelizing the traffic processing operation over a cluster is non-trivial problem.

To make a scalable and distributed traffic analysis system, a lot of research have applied the Hadoop framework [21], which implements a popular parallel programming model, MapReduce [20]. In [22], the authors proposed a scalable Internet traffic measurement and analysis architecture with Hadoop. They evaluated the proposed system using a Hadoop testbed with 30 and 200 nodes. They also demonstrated that the MapReduce based traffic analysis method achieved up to 14 Gbps of throughput for 5 TB input files. RIPE [23] is a packet processing library for Hadoop, and it can be used within MapReduce jobs to natively read pcap formatted files. It took 180 seconds for analysis of 100 GB pcap input file using 100 Amazon EC2 instances. However, it does not consider the parallel processing capability of reading packet records.

HTCondor [24] is a software framework for distributed and parallel processing of workloads that require high computing power. Compared to Hadoop, HTCondor has following advantages, especially on traffic analysis: 1) HTCondor supports C/C++, so legacy codes (e.g., based on libpcap library) can be used to analyze the traffic without additional modification, whereas Hadoop based approach must re-implement the entire logic in MapReduce fashion; 2) Hadoop requires additional I/O operation to read and write pcap (binary) formatted traffic trace while HTCondor supports both of text and binary formatted traffic trace.

Due to the impropriety of Hadoop on traffic processing, in this paper, we propose an HTCondor-based distributed VoLTE traffic classification architecture in Section IV.

III. VoLTE Traffic Classification

In this section, we present a VoLTE traffic classification method. The method is based on traditional application level traffic classification method which relies on not only the application specific payload signature, but also the DPI technique as well. For the DPI technique, we adopt the SIP, RTP based approach which has been mentioned in Section II, because: 1) the VoLTE service relies on the similar underlying communication protocol (e.g., SIP, RTP) as VoIP services; 2) to the best of our knowledge, SIP and RTP based approach

is by far the most simple and effective VoIP classification method. Therefore, by using VoLTE signature and SIP, RTP based DPI technique, we can accurately classify the VoLTE traffic. We will discuss in detail on the proposed VoLTE traffic classification method in the following subsections.

A. VoLTE Signature Generation

Here, we describe how we generated the VoLTE signature for classifying VoLTE traffic. To generate the VoLTE signature, the first step is to collect the VoLTE traffic. We collected the VoLTE traffic from VoLTE enabled smartphones. We chose the Samsung Galaxy S4 LTE as the target device. For LTE service, the LTE communication module provides two virtual interfaces: one for delivering Internet traffic, and the other for delivering VoLTE traffic. By capturing on the interface for VoLTE service, we could collect VoLTE traffic only. We then analyzed VoLTE SIP packets and extracted VoLTE tailored SIP packet signature as a regular expression form from the captured VoLTE traffic as follows:

User-Agent: *-LTE-VoLTE*_AND/1.*

Because VoLTE traffic can only be generated by pre-installed application by mobile network operators, the number of VoLTE application will not be increased rapidly, which means that User-Agent field values will not be changed significantly. Therefore, using User-Agent field value as DPI signature is sufficient to classify VoLTE traffic among LTE traffic.

B. Classification Method

As the VoLTE traffic is comprised of SIP and RTP traffic, we need to classify both types of traffic. The classification of VoLTE SIP traffic is relatively easy and shows more accurate result compared to that of VoLTE RTP traffic, as explained in the previous section. Moreover, since RTP dynamic port information resides inside the SIP payload which is used for establishing a call session, we have to classify the VoLTE SIP traffic first, and then classify the VoLTE RTP traffic. Next, we store the VoLTE RTP port information from the classified VoLTE SIP payload to some temporary space for later reference. We name this temporary space as *VoLTE RTP session table*, and with which we can achieve the VoLTE RTP traffic classification speed improvment because: 1) with VoLTE RTP session table, VoLTE RTP classification no longer relies on the SIP information; 2) we only need to apply the DPI technique to SIP packets only rather than the entire packets; and 3) once a RTP session has been established, it will last for a few minutes; therefore, managing VoLTE RTP session table does not induce much overhead. With considering these benefits, we propose a VoLTE traffic classification method assisted with VoLTE RTP session table. The overall working procedure has been shown in Fig. 1.

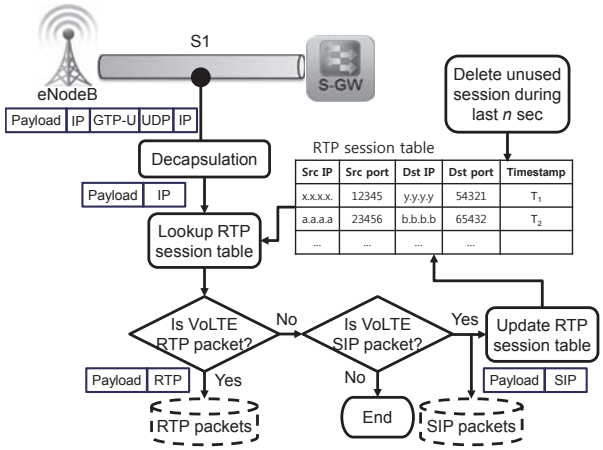


Fig. 1: Proposed VoLTE Traffic Classification Method

The session entry includes RTP source IP address, RTP source port, RTP destination IP address, RTP destination port and the timestamp which stores the time at which its session entry is lastly matched (hit). Each session entry is generated by the first SIP packet that is for establishing the VoLTE RTP session. However, we cannot store all RTP session entries without any bound, due to the limited memory resource and searching efficiency issues. Therefore, we need to find a way to evict the least significant session entry. The significance is defined as follow: 1) validity of a RTP session; and 2) popularity of a RTP session.

The pseudo code of the proposed VoLTE traffic classification method is shown in Algorithm 1. The algorithm takes an LTE packet and the global session table as input. We also introduce two auxiliary functions. *IsUDP()* is used to check whether a LTE packet uses UDP, and *SIPType()* is used to extract the SIP type from a LTE packet.

Algorithm 1: VoLTE Traffic Classification Algorithm

```
input : pkt,session_tbl
output: VoLTE SIP and RTP packet, session_tbl
Remove GTP header from pkt and recover original IP header;
if IsUDP (pkt) = FALSE then
    Exit();
if session_tbl contains a matching entry then
    Store pkt into VoLTE RTP repository;
    Update the timestamp of the entry in session_tbl;
    Exit();
if Destination port number ≠ 5060 then
    Exit();
if SIP packet matches with the signature then
    Store pkt into VoLTE SIP repository;
    if SIPType (pkt) = "BYE" then
        Remove corresponding entry from session_tbl;
    else
        if pkt contains SDP message as a payload then
            Store the RTP session info. to session_tbl;
```

To evaluate the proposed VoLTE traffic classification speed, we performed a set of experiments 10 times using a single machine, equipped with Intel Xeon X5650 CPU (2.66 GHz, 6 cores with Hyper-threading) with 24 GB RAM and a 500 GB HDD with no RAID support. The average throughput was around 0.718($\pm$0.0012) Gbps which is far lower than what we expected. To further enhance the classification speed, we present a distributed VoLTE traffic classification architecture in Section IV.

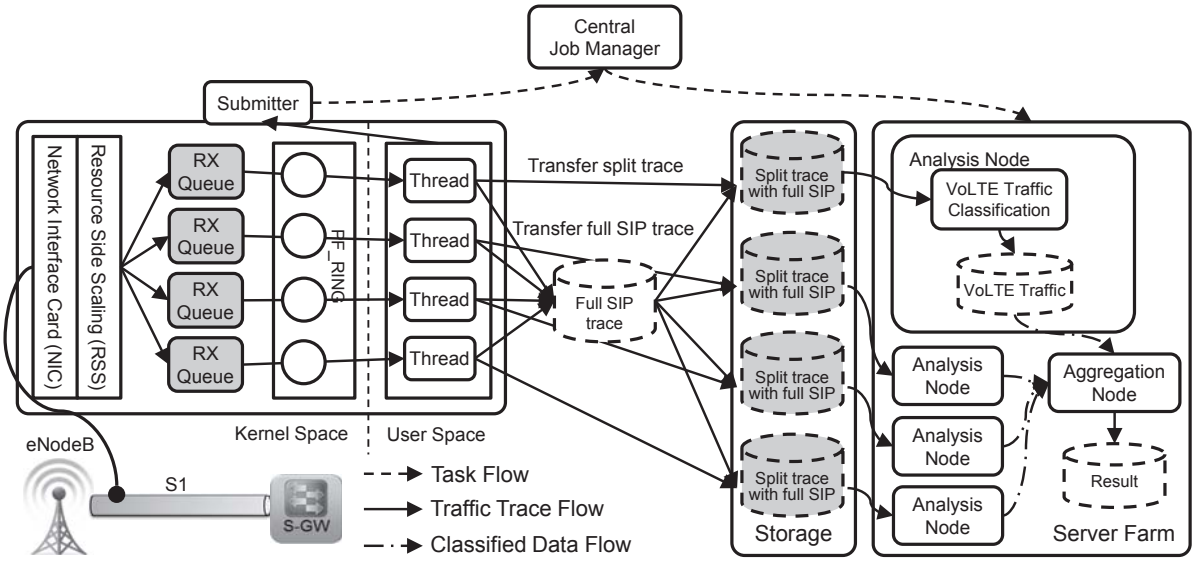


Fig. 2: Distributed Architecture for VoLTE Traffic Classification

IV. Distributed Architecture for VoLTE Traffic Classification

The proposed VoLTE traffic classification method relies on DPI technique, which shows poor classification throughput (less than 1 Gbps with single machine setup), while high classification accuracy in general. Hence, in order to apply the method into real-world deployment in on-line manner (requires 20 Gbps throughput at least), we need to find a way to further improve the classification speed. As a solution, we propose a HTCondor based distributed architecture for the proposed VoLTE traffic classification method.

The classification procedure consists of two main phases which are traffic capturing and traffic processing; and in each phase we may experience the performance bottleneck using the commodity hardware. To maximize the classification performance, we need to mitigate or remove the bottleneck in each phase and realize operational parallelism.

- **Traffic Capture**: in traditional x86 machine, only one RX queue resides in kernel space to receive the packets from network device, and multiple user level application threads compete each other for accessing the single RX queue. This design is inefficient for parallelizing the traffic capturing operation, thus to be a solution, we adopt the PF_RING [25] and Receiver-Side Scaling (RSS) [26] technology in our design. RSS is a state of art technology, which enables x86 machines to receive the packets using multiple RX queues, while PF_RING binds the each user-level thread to the individual RX queue, so that each thread can directly access RX queue without any competition. Such design can alleviate the single RX queue bottleneck and significantly increase the overall traffic capturing performance. Using PF_RING, we can achieve over 20 Gbps packet capturing speed.
- **Traffic Process**: traffic classification operation requires intensive computing power which typically cannot be fulfilled by using single machine even with the help of multi-core and hyper-threading technology. Therefore, distributed traffic processing becomes popular recently. HTCondor is a high throughput computing framework suited for processing a large job in distributed manner. With HTCondor, a large job can be split into multiple small jobs, and submitted to the *Central Job Manager* using *Job Submitter*. Each job binds to a job description file which specifies the path of binary program (e.g., classification program) that executes the job, the path of file (e.g, LTE traffic trace file) to be processed and the environmental requirements to execute the jobs. Central Job Manager is a matchmaker which assigns the job to a computing machine that fulfills the execution requirement. Each job is then executed by the computing node, and the results are reported to an aggregation node.

With considering the corresponding solutions for resolving the bottleneck in each classification phase, we design a new distributed architecture tailored to VoLTE traffic classification. The detailed design is depicted in Fig. 2. The LTE traffic is captured at s1 point, and distributed to multiple RX queues using RSS. Each RX queue is directly accessed by an application thread using PF_RING. The application thread is in charge of 1) generating the job description file and transferring to Job Submitter, and 2) storing the received LTE traffic into the computing machine where the classification job will be executed. To correlate the LTE traffic trace with the classification job, we generate a random id and assign it to the LTE traffic trace and job description file.

The resulting architecture indeed sounds, however, in order to exploit it to real-world VoLTE traffic classification, we need to further resolve three technical issues as follows:

- **Inefficient Job Scheduling**: the Central Job Manager receives the classification jobs from Job Submitters, and assigns them to the most appropriate computing machine. The job assignment is realized using bin-packing algorithm with default configuration. The key idea of bin-packing is to pack multiple jobs into the minimum number of computing machines. Since many jobs are simultaneously executed in a single machine, we may experience drastic performance

degradation. To avoid this, we implement the load balancing logic inside job scheduler to make the Central Job Manager assign the jobs as even as possible.

- **Dependencies Between Split Traffic Trace**: the RSS based trace splitting scheme only takes into account the packet index number, rather than the SIP session state. Such trace splitting scheme makes the RTP packets with the same SIP session split into different traffic trace, and this in turn results the low classification accuracy. To resolve this issue, we classify the VoLTE SIP traffic from each RX queue in advance, and periodically aggregate and append it to each split traffic trace. By doing so, each traffic trace preserves full SIP traffic information, which in turn ensures that each computing machine has the complete SIP session information. Hence, no matter how the traffic is distributed in different machines, the system can always correctly classify the VoLTE RTP traffic. Note that SIP traffic classification only relies on port based traffic classification technique (e.g., UDP 5060 port), thus the classification overhead is fairly moderate.
- **Multiple Simultaneous HDD Access**: although we can resolve the computational bottleneck by utilizing the HTCondor framework, the HDD I/O may still remain as a bottleneck with multiple simultaneous access. The reason is that with multiple CPU cores and single HDD architecture, multiple concurrent classification jobs compete each other on accessing the traffic trace which resides in a single HDD. To resolve this issue, we proportionally scale the number of HDDs to that of CPU cores, so that each job independently accesses the HDD without sharing.

V. Evaluation

In this section, we present the dataset as well as tesbed setup for evaluating the distributed VoLTE traffic classification architecture. An exhaustive evaluation on the classification speed is conducted by varying the number of concurrent classification jobs with different experiment parameter setup. We also show the classification result on VoLTE SIP and RTP traffic at the end of this section.

A. Dataset and Testbed Setup

We exploited the real-world trace captured from a commercial LTE core network, specifically at the s1 interface. The trace contains around 2 minutes of LTE traffic which has around 19.91 GB traffic volume in total, stored in pcap binary format. In order to preserve the user privacy, the trace has been gone through the anonymization procedure in advance. Since it is infeasible for us to directly integrate our classification program into LTE operator's network to perform the evaluation, we conducted the experiment in off-line manner.

We implemented the proposed classification method using C language to maximize the classification performance. We utilized 7 high performance machines to conduct the experiments. All machines were equipped with Intel Xeon X5650 CPU (2.66 GHz, 6 cores with Hyper-threading support) with 24 GB RAM and six 500 GB HDD. HTCondor 8.2.2 was installed in 7 machines, and one machine worked as a master node (Job Submitter + Central Job Manager), while 6 machines worked as analysis node which executes the classification job. Since each CPU has 6 cores with Hyper-threading capability, each analysis node can execute up to 12 jobs simultaneously.

B. Experiment Result

We performed three sets of experiments, and each experiment was repeated 5 times. In all experiments, the classification throughput and job completion time were measured, and the averaged result was computed. Moreover, in each set of experiments, we differed the parameters including the number of machines, number of HDDs embedded in each machine and job scheduling algorithm, to figure out how those parameters can affect the overall classification performance.

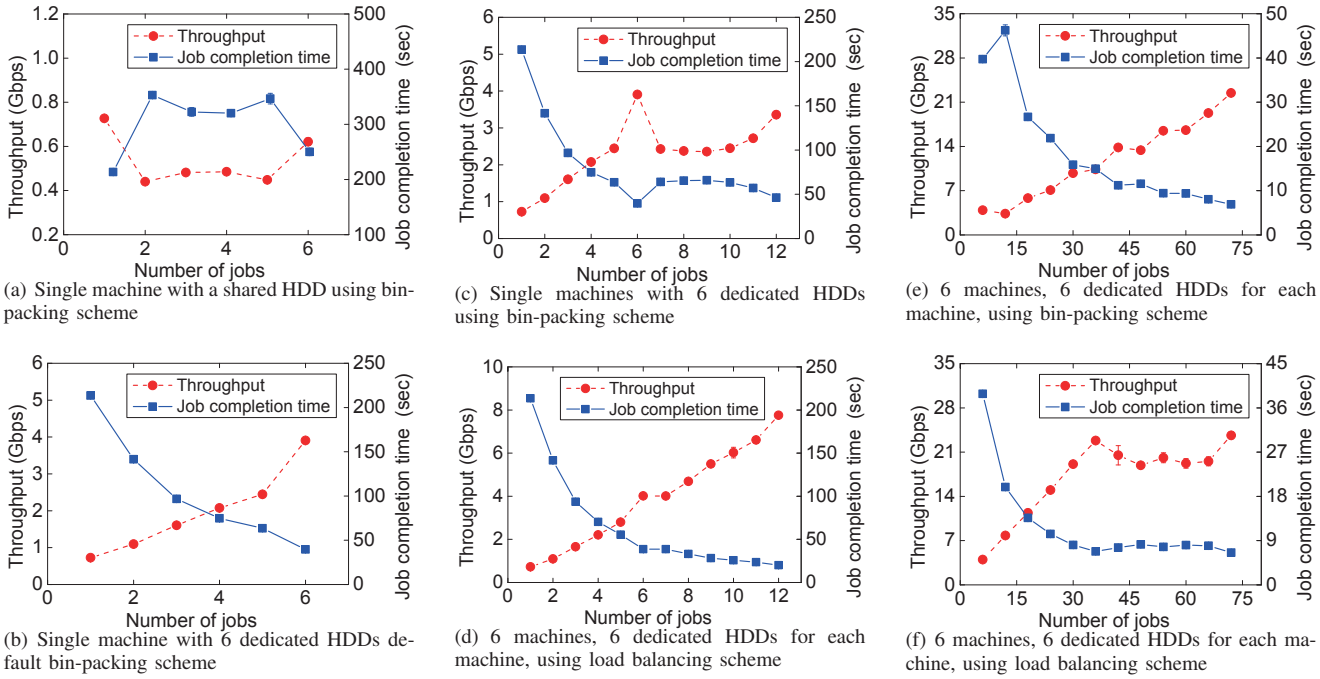
In the first set of experiments, we executed the classification job using single machine with one and six HDDs, while gradually increased the number of concurrent jobs up to 6. The results have been shown in Fig. 3(a) and Fig. 3(b). With one HDD case, no matter how we changed the number of concurrent jobs, the overall throughput was not increased accordingly. However, with 6 dedicated HDDs case, the classification throughput was increased in accordance with the number of concurrent jobs. Such discrepancy was induced due to the HDD bottleneck. With single machine and a shared HDD, we achieved around 0.7 Gbps throughput, while with single machine and 6 dedicated HDDs, we achieved around 4 Gbps throughput which is almost 6 times faster than the shared HDD case.

In the second set of experiments, we compared overall classification performance using different scheduling schemes with small number of concurrent jobs (e.g., up to 12). The results are depicted in Fig. 3(c) and Fig. 3(d). In single machine case, the classification throughput was rapidly increased until the number of concurrent jobs reached 6. With 7 concurrent jobs, the overall throughput was dramatically decreased. The reason was that with the bin-packing scheme, because the 7th job had to share one of the HDD with the other existing jobs, HDD I/O became bottleneck. Contrarily, the overall classification throughput improved proportionally with the number of concurrent jobs using load balancing scheme. This was because with load balancing scheme, 12 concurrent jobs were evenly distributed to 6 machines, and by doing so, each machine only needed to execute 2 concurrent jobs, so that we cannot experience any performance degradation. Overall, in the latter case, we achieved around 8 Gbps classification throughput.

In the third set of experiments, we further enlarged the number of concurrent jobs up to 72. Similar to the second set of experiments, we compared the classification performance using different scheduling schemes, and the results are shown in Fig. 3(e) and Fig. 3(f). In bin-packing case, we observed that the throughput increased linearly in the range of $[12n+1, 12n+6], n \in \{0, 1, 2, 3...\}$, while the increasing trend was relatively gentle in the range of $[12m-5, 12m], m \in \{1, 2, 3, 4...\}$. With bin-packing scheme, since the first 12 jobs were all assigned into the single machine, only the first 6 jobs could access the HDDs without sharing (increasing section in graph), while from the 7th arriving job had to share the HDD resource (standstill section in graph). Whereas, using load balancing scheme, the linearly increasing trend lasted up to 36 concurrent jobs, and performance improvement became sluggish after 37th concurrent job arrival. Although there were large discrepancies between two scheduling schemes in terms of throughput

TABLE I: VoLTE Traffic Classification Results

	Total Trace	Packets		Ratio	
		VoLTE RTP	VoLTE SIP	VoLTE RTP	VoLTE SIP
Volume (Bytes)	20,882,089,648	20,588,426	357,861	0.098 %	0.0017 %
Number of packets	26,548,792	142,348	251	0.536 %	0.0009 %



(a) Single machine with a shared HDD using bin-packing scheme

(c) Single machines with 6 dedicated HDDs using bin-packing scheme

(e) 6 machines, 6 dedicated HDDs for each machine, using bin-packing scheme

(b) Single machine with 6 dedicated HDDs default bin-packing scheme

(d) 6 machines, 6 dedicated HDDs for each machine, using load balancing scheme

(f) 6 machines, 6 dedicated HDDs for each machine, using load balancing scheme

Fig. 3: Throughput and Job Completion Time of the Distributed Traffic Classification

increasing trend, however, in both cases the system shown 23.689 Gbps classification throughput which is approximately 35 times faster than the system without distributed processing.

We also compared the number of classified SIP and RTP packets as well as the volume of classified SIP and RTP traffic with and without the distributed architecture. In both cases, it showed the identical classification results (Table I). As we can notice from the results, the VoLTE traffic occupies quite small portion among overall LTE traffic. The reasons are as follow: 1) only recently released smartphones support VoLTE feature; 2) in most of VoLTE enabled smartphones, the VoLTE function is disabled by default; and 3) VoLTE phone call only can be established between two VoLTE enabled smartphones. Overall, we can conclude that our distributed architecture significantly improves the classification throughput without sacrificing the classification accuracy.

VI. Conclusion

In this paper, we proposed a VoLTE traffic classification method and its distributed architecture. Among various types of classification methods, we chose the DPI based application classification method as a reference, generated the SIP payload signature of the VoLTE application, and classified the SIP as well as RTP traffic using the synthesized payload signature. Moreover, to improve the RTP traffic classification speed, we introduced a VoLTE RTP session table, with which no additional reference to the SIP payload is required within the same session. We applied the proposed classification method to the trace captured from real-world LTE core network, and achieved around 0.718 Gbps classification speed. To further enhance the classification speed, we presented a HTCondor based distributed architecture and its solution for VoLTE traffic classification. We deployed our solution over a small scale cluster to evaluate the classification throughput. Our solution scales well, and with 6 computing node, we achieved around 23.689 Gbps throughput, which was approximately 35 times faster than the result without the distributed processing.

By far, to the best of our knowledge, there is only one type of VoLTE application traffic (for voice call) delivered through each telco's LTE network; hence, as long as we have the correct payload signature of VoLTE application, we can achieve very high classification accuracy. We did not perform quantitative evaluation on the accuracy of VoLTE traffic classification in this work, due to the difficulties on collecting ground truth data from LTE core network. Therefore, as a future work, we will find a way to synthesize the VoLTE traffic marked with ground truth information to evaluate the accuracy of the proposed VoLTE classification method as well as its distributed architecture. We also plan to deploy the entire solution into LTE core network to perform the evaluation in on-line manner.

REFERENCES

[1] Cisco, "Global mobile data traffic forecast update, 2010-2015," *Cisco Visual Networking Index (VNI) Forecast*, 2011.

[2] T. Yildirim and P. Radcliffe, "VoIP traffic classification in ipsec tunnels," in *Electronics and Information Engineering (ICEIE), 2010 International Conference On*, vol. 1, Aug 2010, pp. 151–157.

[3] R. Birke, M. Mellia, M. Petracca, and D. Rossi, "Experiences of VoIP traffic monitoring in a commercial isp," *International Journal of Network Management*, vol. 20, no. 5, pp. 339–359, 2010.

[4] T. Sinam, I. T. Singh, P. Lamabam, N. N. Devi, and S. Nandi, "A technique for classification of VoIP flows in udp media streams using VoIP signalling traffic," in *Advance Computing Conference (IACC), 2014 IEEE International*, 2014, pp. 354–359.

[5] B. Li, M. Ma, and Z. Jin, "A VoIP traffic identification scheme based on host and flow behavior analysis," *Journal of Network and Systems Management*, vol. 19, no. 1, pp. 111–129, 2011.

[6] I. Fauzia and A. K. Uzma, "A generic technique for voice over internet protocol (voip) traffic detection," *International Journal of Computer Science and Network Security*, vol. 8, pp. 52–59, 2008.

[7] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, "Rfc 3261: Sip: Session initiation protocol," IETF, Tech. Rep., 2002. [Online]. Available: www.ietf.org/rfc/rfc3261.txt

[8] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "Rfc 3550: Rtp: A transport protocol for real-time applications," IETF, Tech. Rep., 2003. [Online]. Available: www.ietf.org/rfc/rfc3550.txt

[9] S. Valenti, D. Rossi, A. Dainotti, A. Pescape, A. Finamore, and M. Mellia, "Reviewing traffic classification," in *Data Traffic Monitoring and Analysis*, 2013, vol. 7754, pp. 123–147.

[10] B. Park, Y. Won, J. Chung, M.-s. Kim, and J. W.-K. Hong, "Fine-grained traffic classification based on functional separation," *International Journal of Network Management*, vol. 23, no. 5, pp. 350–381, 2013.

[11] M. sup Kim, Y. J. Won, and J. W. ki Hong, "Application-level traffic monitoring and an analysis on ip networks," *ETRI Journal*, vol. 27, 2005.

[12] *IP Multimedia Subsystem (IMS)*, 3GPP, 2013.

[13] *GPRS Tunnelling Protocol (GTP) across the Gn and Gp Interface*, 3GPP, 1998.

[14] *Evolved Universal Terrestrial Radio Access (E-UTRA) and Evolved Universal Terrestrial Radio Access Network (E-UTRAN); Overall Description; Stage 2*, 3GPP, 2014.

[15] D. Bonfiglio, M. Mellia, M. Meo, D. Rossi, and P. Tofanelli, "Revealing skype traffic: When randomness plays with you," in *Proceedings of the 2007 SIGCOMM*, 2007, pp. 37–48.

[16] D. Adami, C. Callegari, S. Giordano, M. Pagano, and T. Pepe, "A real-time algorithm for skype traffic detection and classification," in *Smart Spaces and Next Generation Wired/Wireless Networking*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2009, vol. 5764, pp. 168–179.

[17] P. A. Branch, A. Heyde, and G. J. Armitage, "Rapid identification of skype traffic flows," in *Proceedings of the 18th International Workshop on Network and Operating Systems Support for Digital Audio and Video*, ser. NOSSDAV '09, 2009, pp. 91–96.

[18] R. Alshammari and A. N. Zincir-Heywood, "An investigation on the identification of VoIP traffic: Case study on gtalk and skype." in *CNSM*, 2010, pp. 310–313.

[19] "L7-filter," http://l7-filter.sourceforge.net/protocols/.

[20] J. Dean and S. Ghemawat, "MapReduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[21] "Hadoop," http://hadoop.apache.org/.

[22] Y. Lee and Y. Lee, "Toward scalable internet traffic measurement and analysis with hadoop," *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 1, pp. 5–13, 2013.

[23] "RIPE Hadoop PCAP," https://labs.ripe.net/Members/wnagele/large-scalepcap-data-analysis-using-apache-hadoop.

[24] "High Throughput Computing Framework - HTCondor," http://research.cs.wisc.edu/htcondor/.

[25] L. Deri, "Improving passive packet capture: Beyond device polling," in *In Proceedings of SANE 2004*.

[26] "Msdn: Introduction to receive-side scaling," http://msdn.microsoft.com/en-us/library/windows/hardware/ff556942(v=vs.85).aspx.