Autonomic QoS Management Mechanism in Software Defined Network

WANG Wendong, QI Qinglei, GONG Xiangyang, HU Yannan, QUE Xirong

State Key Laboratory of Network and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, P.R. China

Abstract: With the increase of network complexity, the flexibility of network control and management becomes a nontrivial problem. Both Software Defined Network (SDN) and Autonomic Network technologies are sophisticated technologies for the network control and management. These two technologies could be combined together to construct a software defined self-managing solution for the future network. An autonomic QoS management mechanism in Software Defined Network (AQSDN) is proposed in this paper. In AQSDN, the various QoS features can be configured autonomically in an OpenFlow switch through extending the OpenFlow and OF-Config protocols. Based on AQSDN, a novel packet context-aware QoS model (PCaQoS) is also introduced for improving the network QoS. PCaQoS takes packet context into account when packet is marked and managed into forwarding queues. The implementation of a video application's prototype which evaluates the self-configuration feature of the AQSDN and the enhancement ability of the PCaQoS is presented in order to validate this design.

Keywords: software defined network; autonomic management; context aware; quality of service (QoS)

I. INTRODUCTION

As the increment of various network equipments and services, the complexity of network control and management has risen sharply in recent years. The QoS management is an important part of network management. Various QoS models and mechanisms have been proposed, but there is no one-fit-all algorithm. The advantages and disadvantages of the different combinations of queue management and scheduling, which works at different context conditions (various workloads or transmission mediums), are analyzed [1]. Currently, different services or applications want to obtain the different network resource for fulfilling their requirements. So the self-configurable QoS models and mechanisms based on the context-aware are highly expected.

In order to reduce the network management cost and the probability of network failure, autonomic network technologies is introduced in network control and management, and self-* attributions with the MAPE-K [2] or similar model are defined. The self-* attributions include self-configuration, self-protecting, self-healing and self-optimization/adjusting [3]. The autonomic function is also considered as an important component in many projects [4-8] which focused on future network architecture.

Software Defined Network (SDN) separates the control plane and the data plane, some network control and management functions could be implemented through the software instead of updating huge numbers of network elements. OpenFlow protocol which proposed by Open Networking Foundation (ONF) is the most popular standard for communication between controller and OpenFlow switch in SDN. In order to support QoS management functions, each version of OpenFlow protocol defines the *enqueue* action, but the queue configuration has not been involved. In version 1.3 [9], the meter table, meter band, and two optional band types (i.e. drop and dscp remark) are defined. According to these definitions, other band types for enhancing the meter ability of the OpenFlow switch could be added. As an auxiliary protocol of the OpenFlow protocol, OF-Config[10] could be exploited to configure those functionalities, which are not supported by the OpenFlow protocol, at the lower time-scale , e.g., the minimum and maximum transmission rates of a queue in an OpenFlow switch. And it is also extensible to carry more QoS configurations.

We design an Autonomic QoS management mechanism in SDN (AQSDN) for network QoS guarantee. In this mechanism, the controller undertakes the function of analysis and decision in the autonomic control loop, while the switch acts as collector of the context and executive of the behavior. This mechanism controls adaptively the QoS rules according to the context from data plane and the policy from the operator. In addition, for improving the quality of multimedia service, we also propose the Packet Context-aware QoS model (PCaQoS), which processes packets according to their semantic precedence level.

The remainder of the paper is organized as follows. The related work about QoS control and management for software defined network is described in section II. Then the AQSDN architecture is outlined in section III, and the PCaQoS is introduced in section IV. The self-configuration feature of AQSDN and the perfomance of PCaQoS are evaluated in Section V. Section VI concludes the paper.

II. Related Work

Several solutions for QoS guarantee through adjusting flow routing without extending OpenFlow protocol are proposed [11-13]. A novel optimization model is designed in [11], it distinguishes the specified QoS traffic from normal best-effort traffic when services routing is selected, so that the QoS contract of scalable video streaming could be satisfied. The model in [12] is an extended version of [11]. It opens the API for service providers to interfere the route calculation. The management layer is designed by utilizing the North Bound interface of the SDN controller[13]. The management layer validates and enforces QoS policies which are one of the basements of the route decision in the control layer. However, these works don't involve the configuration of various QoS strategies into SDN switch currently.

An automatic and scalable QoS control architecture is presented in [14]. It adds rate-limiters and queue mapping API to OpenFlow. The rate-limiters API is used to map one or more flows to the special rate limiters, while the queue mapping API is used to map a flow to the special priority queue. QoSFlow [15] extends the OpenFlow protocol 1.0 for improving the QoS control capability (i.e. Shaping by using Hierarchical Token Bucket, Schedulded by using Stochastic Fairness Queuing and Congestion avoidance by using Randomly Early Detection) on OpenFlow/SDN networking. PindSwitch [16] provides Output Queue Control message for configuration of the queue scheduling. However, none of them take the consideration of packet marking. [15, 16] integrate the configuration of queue into OpenFlow protocol rather than explore OF-config to configure the rules of queue management and scheduling as we do.

III. AQSDN Architecture

The AQSDN Architecture is illustrated in Fig.1. The QoS control module which is located in SDN controller decides or chooses QoS rules dynamically. The OpenFlow protocol and OF-config protocol are extended for acquiring the context from SDN switch or issu-

An autonomic QoS management mechanism in Software Defined Network (AQSDN) is proposed in this paper. In AQSDN, the various QoS features can be configured autonomically in an OpenFlow switch through extending the OpenFlow and OF-Config protocols.

ing the QoS rules to SDN switch. For sensing context and instantiating the QoS rules, several functions and components are implemented in OpenFlow switch.

3.1 QoS control module

The QoS control module, which is an application of the ONF controller, has two function models. One is QoS scheme decision model, which determines suitable queue management and scheduling schemes as well as their parameters. This model is activated infrequently, e.g., when a switch is initialized or when the operator forces it. The other one is QoS action decision model, which determines packets marking and designates the queue for each flow adaptively. This model may be activated frequently. QoS schemes and actions are collectively referred as QoS rules. The QoS control module comprises Context manager, Analysis, Rule decision, Requirement DB, Rule DB and Policy DB. The detail functions of these models are described as follows:

Requirement DB: It stores the QoS requirements, which are provided by network operator, end user or other applications running on the controller.

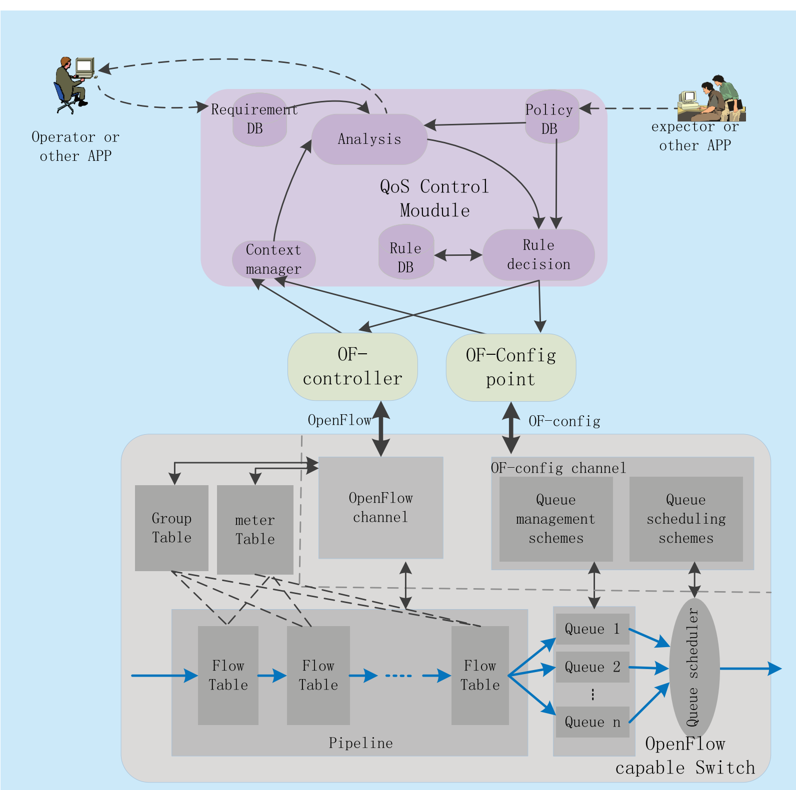

Fig.1 *The AQSDN architecture*

Rule DB: It stores the historical QoS rules which are used for the special requirements or in specified network environments.

Policy DB: It stores the QoS Polices supported by OpenFlow switch, which can be provided by network operator or other applications.

Context manager: context manager is responsible for aggregating, storing and managing context perceived from the data plane. Generally, context could include any information charactering the situation of an entity [17]. In AQSDN, we only focus on QoS management of the network. So the entity is network and service. The network context and the Flow context are defined as Definition 1 and 2.

Definition 1. ***(Network Context)***: network context is an information set characterizing the network performance, which includes the state information of the switch or node (e.g., utilization ratio of CPU and memory on a node and the length of the packet queue), and the link information, (e.g., the Packet Loss Ratio (PLR) delay and jitter, available bandwidth of the link).

The controller attains the node state information through accessing the Management Information Base (MIB), and the link information through analyzing the statistical information of the flow. As we known, the statistical information are divided into multiple granularities: table level, flow level, port level, queue level and meter level.

Definition 2. ***(Flow Context)***: Flow context is an information set characterizing a service flow, which includes the inherent feature (e.g., the service type and the QoS requirement), and the real-time flow feature (e.g., the burst rate).

The inherent features of the service can be carried in the first packet of the flow, and then be achieved by controller with Deep Packet Inspection (DPI) technologies. The controller can achieve the real-time feature through querying the switch for statistical information.

Analysis: The tasks of the analysis module are to analyze whether the QoS requests can be satisfied and if there are conflicts among them. If conflicts are founded or the QoS requests could not be satisfied, these resaults are sent to the administrator. If the QoS requests could be satisfied, the related contexts and requirements are forwarded to the QoS rule decision module.

Rule decision: The Rule decision component can chooses the appropriate QoS rule from the QoS Rule DB if it exits. Otherwise, this component plans new QoS rule based on the context, QoS requirement and QoS policy stored in the QoS Policy DB, and stores the new QoS rule into the QoS Rule DB for use later.

To describe the control procedure clearly, the main process of the QoS control module is presented in algorithm 1.

Algorithm 1 QoS_Control

```
PROCEDURE QoS_Control ( ) {
while (TURE) {
QoS_req:=read_QR (Requirement_DB );
  QoS_policy:=read_QP(QoS_policy_DB);
     Context:=wait_for_context();
Result:=analysis(QoS_req,QoS_policy, Context)
if(Result==Conflict)
exception(THERE_IS_CONFLICT);
else if(Result==Not_Guarantee)
exception (QoS_IS_NOT_SATISFIED);
  else
QoS_rule:=QoS_rule_decision (QoS_req,QoS_policy, Context);
store_QoS_rule(QoS_rule);
    }
  }
```

3.2 The enforcement of the QoS rules

For QoS action in each flow, the packet marking algorithms selection is configured via OpenFlow protocol by the OpenFlow controller. Meanwhile, the QoS schemes for each queue, queue management and schedule schemes are configured through OF-config protocol by the OF-config point.

3.2.1 The enforcement of the QoS actions

The existing OpenFlow protocol defines the message about the configuration of the flow table, the group table and meter table. The *enqueue/set-queue* action specifies which queue attached to a port should be entered by a flow. And the meter table and meter band provide the meter operation for flow. However, the existing *remarking* bands, which only lower the drop precedence level of the packet, does not satisfy various QoS requirement. So it is necessary to extend OpenFlow protocol so that it could support diverse packet marking algorithms. Based on the definition of meter band, the band type, rate, and counters fields are fixed, while other arguments are variable with the band type. For example, there is *prec_level* field in *drop* band and there is not in *remarking* band. On the other hand, the number of specific arguments is unpredictable for new band. So we explore the Type-Length-Value (TLV) format to show the specific arguments (shown in Fig. 2) instead of the predefinition of the structure for each meter band. In the switch, the corresponding code for creating and applying each meter band is needed. we add two packet marking algorithms as new meter bands, Single Rate Three Color Marker (srTCM)and Packet Context-aware Packet Marker (PCaPM), which is described in section 4.1.

3.2.2 The enforcement of the QoS rules

OF-Config is transported using NetConf protocol. Through choosing suitable operation (e.g. <get-config>, <edit-config>), we can achieve and set the configuration of the re-

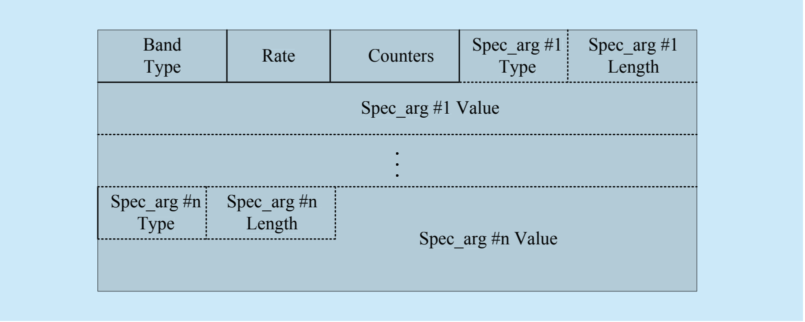

Fig.2 *The format of meter band with variable specific argument*

source in the switch. The existing OF-Config has provided the support of the minimum and maximum transmission rates of a queue. For supporting the configuration of Queue management and Queue scheduling schemes, we enrich the operate set of the OF-Config on the content layer of the NetConf. As shown in Fig. 3, we supplement the queue management and queue scheduling schemes in the configuration of OpenFlow queue. The queue management schemes include Weighted Random Early Detection (WRED) and Packet Context-aware Queue Management (PCaQM), which is described in section 4.2. The queue scheduling schemes include Priority Queuing (PQ) and Weighted Round-Robin (WRR).

We add OF-config channel in SDN switch. Similar with the OpenFlow channel, it is another interface that connects each OpenFlow switch to controller. This interface is responsible for decapsulation or encapsulation of the OF-config message from/to the OF-Config point, and instructing/receiving the configuration of the resource. So the switch can exact the OF-config message and transform it to the switch/router easily. In addition, A FPGA with well-defined interfaces is proposed in reference [1]. It could support the configurability functions of queue management and scheduling in switch/router. This solution brings the bright future for automatic QoS management.

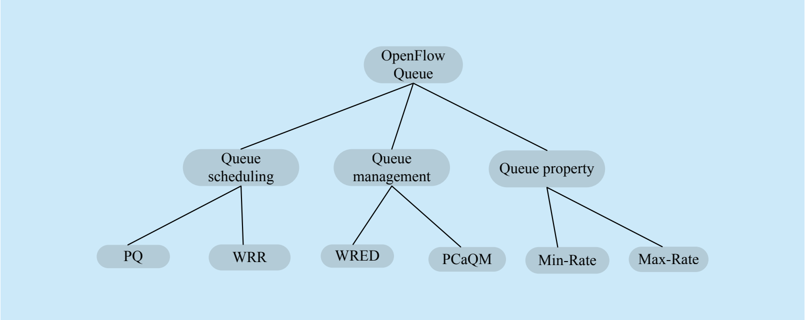

Fig.3 *The structure of configuration items for the OpenFlow queue*

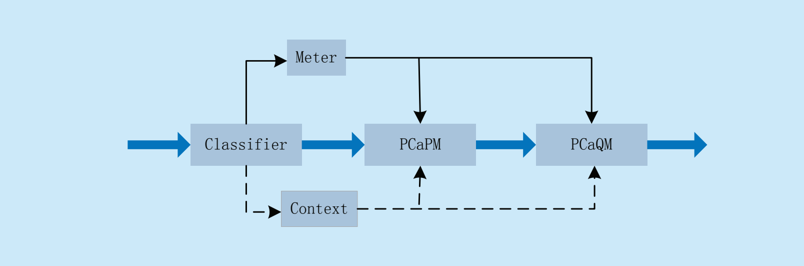

Fig.4 *The PCaQoS model*

IV. Packet Context-Aware QoS Model

Similar with the context definition of network and flow, the packets also have its context, which are defined as definition 3. Take video service as an example, in Group of Pictures (GoP), the precedence level of I frame is the highest, and the precedence level of P frame is higher than B frame's.

Definition 3. ***(Packet Context)***: packet context is an information set (e.g., semantic precedence level) characterizing the packet.

The packet context can be carried by traffic class (TC) field in the IPv6 header or (ToS) field in IPv4 header. In TC field, the format of semantic precedence level follows the definition of the class selector (CS) encoded point in DiffServ model. The QoS guarantee ability would be improved if the SDN take packet context into account. However, it is impractical to deliver the packet context to the controller because this needs the frequent communication between the switch and the controller. So we design the Packet Context-aware QoS (PCaQoS) model enhanced from the DiffServ model, which enables the switch to perceive the packet context and responds it locally. The PCaQoS model is described in figure 3. It is different with the DiffServ model. In addition to the metering information about the flows, the marker and the queue manager also take the packet contexts into account. They are called as Packet Context-aware Packet Marker (PCaPM) and Packet Context-aware Queue Management (PCaQM) respectively.

4.1 Packet context-aware packet marker

A kind of autonomic packet marking (APM) algorithm is presented in [18]. It tries to exploit token reservation and token borrowing for giving high probability of being marked green to the critical packets. The packets are divided roughly two categories: critical and normal packets. And APM still uses three colors to mark the transmission grade, which cannot provide exact differentiation service.

In this section, a new packet marking mechanism PCaPM is presented. It uses multicolor marking [19-21] to distinguish various precedence levels through extending the metering-based marker. PCaPM initiate a multicolor packet marker by remarking the DSCP code based on the packet context and the marking result of the metering-based marker. The marking result is mapped into the Pool 2 (xxxx11). Fig. 5 illustrates a case of the PCaPM. In this case, the metering-based marker is srTCM, the packets of a service present three kinds of priority (high, middle and low) and the mapping relationships from meter result to remarking color are listed in Table I.

Table I *The mapping relationships from meter results to remarking colors*

The marking results by the APM	Semantic priority level	The marking results by the PCaPM
Any	network control	dark green
Green	high	Green
	middle	light green
	low	Cyan
Yellow	high	Cyan
	middle	Yellow
	low	Orange
Red	high	Orange
	middle	Red
	low	dark red

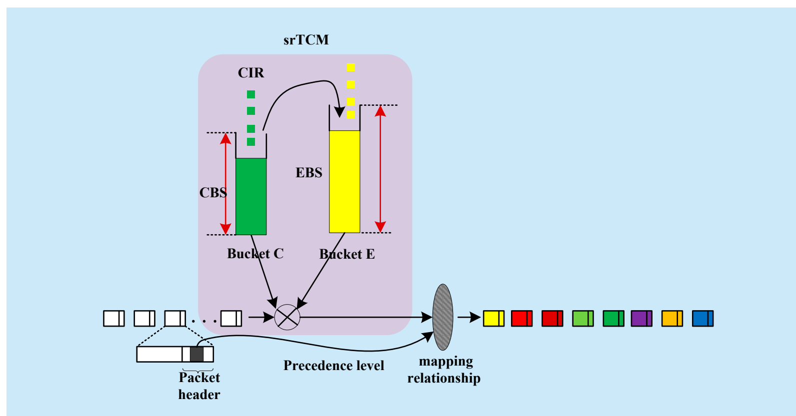

Fig.5 *The PCaPM marker*

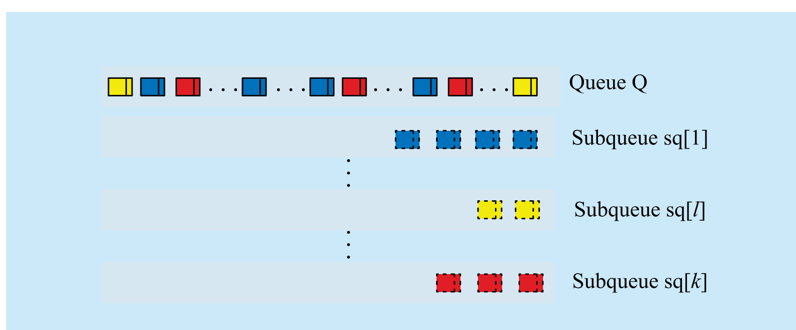

Fig.6 *The mapping relationships between the real queues and the virtual sup queues*

4.2 Packet context-aware queue management

The PCaQM coordinates closely with the PCaPM. If an integrated flow is marked k colors or priorities by PCaPM, PCaQM derives k virtual sub queues from the original queue. All of packets with the same priority $l(1 \le l \le k)$ in original queue Q compose sub queue sq[l]. The sequence of packets in sub queue sq[l] is same as theirs in original queue Q. sq[l] is a link of pointers which point to packets in queue Q instead of requesting new memory for storing packets' copies. The mapping relations between the real queue and the virtual sub queues are shown in Fig. 6.

When packet p is appended into the queue Q, its pointer pointing to it is adding the tail of sub queue sq[$p.PL$], of which $p.PL$ indicate the precedence level of packet p. Reversely, when packet p is moved away from queue Q, its pointer is also moved away from sq[$p.PL$]. PCaQM algorithm is indicated as algorithm 2.

In many queue management schemes, the queue length is the only parameter for computing the packets loss probability. This design is reasonable when all the packets are in an equal status. However, this is unfavorable when the packets have different priorities. For example, in a congested route, if lots of lower priority packets arrive suddenly in a short time, the queue length increases sharply so that the higher priority packets coming later will be dropped. So PCaQM takes packet priority as another metric for the incoming packets processing.

Algorithm 2 PCaQM

```
PROCEDURE PCaQM (Packet p, Queue Q) {
1: l:= p.PL;
2: qlen:= av_len(Q);
3: Pr:= Calc_Discard_Probability(qlen, Q.thmin[l], Q.thmax[l], Q.pmax[l]);
4: if ( Packet_Should_be_Discarded (Pr) ) {
5:   ReplaceDiscard(p, Q);
 }
   else {
6:      Q. AppendPacket(p) ;
7:  Q.sq[l]. AppendPacket (prep) ;
      }
 }
```

Algorithm 3 ReplaceDiscard

```
PROCEDURE ReplaceDiscard (Packet p, Queue Q) {
1: l:= p.PL;
2: sq_low:= null;
3: for (i:= k to l+1 step -1) do {
4:      if (Q.sq[i] == null) continue;
5:         sq_low:= Q.sq[i] ;
6:         break;
    }
7: if (sq_low == null) {
8:  Discard(p);
   }
   else {
9:         p_rep:= sq_low.GetQueueHeader ();
10:       Q.RemovePacket(p_rep) ;
11:       sq_low.RemovePacket (p_rep) ;
12:       Q. AppendPacket(p) ;
13:       Q.sq[l]. AppendPacket (p_rep) ;
}
}
```

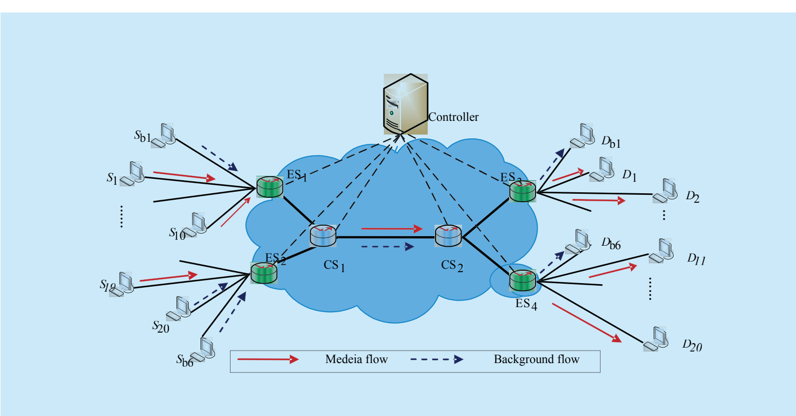

Fig.7 *The implementation of prototype system*

The detail dropping packet procedure of PCaQM is indicated by algorithm 3. If there is packet that should be dropped when the packet p arriving, PCAQM will drop the header packet p' in sq[$p.PL$] that is the lowest priority sub queue in all non-empty sub queues whose priority is lower than $p.PL$. If the packet satisfying the condition cannot be found, packet p is dropped.

V. Evaluation and Results

5.1 The implementation of prototype system

The prototype system of AQSDN, which config QoS polices in SDN according to the policy of multimedia service autonomically, is illustrated by Fig.7. It is composed of three Dell R410 and four R710 servers. One R710 server acts as the controller through running NOX platform [22], while other servers act as either core or edge OpenFlow-enabled switches by running Ofsoftswitch13 [23]. These switches enable to support both DiffServ and PCaQoS models. In addition, several PCs serve as source and destination hosts. The host connects the edge switch directly. The bandwidth of the bottleneck between two core nodes CS1 and CS2 is 20Mbit/s, and the delay is set to 5ms. The bandwidth and delay of any other link are 100Mbit/s and 2-10ms respectively.

Ten different videos are explored as the experiment service, each of which is transmitted on two pairs of hosts separately (e.g., $S_1 \rightarrow D_1$ and $S_{11} \rightarrow D_{11}$). These videos are provided by Faculty of Electrical Engineering and Computer Science, Technische Universität Berlin. Their characteristics are illustrated in Table II. These videos are coded and encoded by H.264/AVC JM 18.5 [24]. The video format is Common Intermediate Format (CIF, 352x288 solution, 30fps). The packet semantic precedence level of video flow is known as: $P_I > P_P > P_B$, where P_I, P_P and P_B are the priority of I frame packets, P frame packets and B frame packets in one Group of Picture (GOP). The frame order of H.264 is IBPBPB…, and the interval between two I frames is 20 frames. In order to change the congestion level, we inject some background traffic flows into the network. Through adjusting the background traffic flows, we can achieve higher congestion level in the bottleneck link.

In whole process of the emulation, the video between each pair host is transferred many times circularly (3~20 times based on the duration of the video), and the background traffic flow is also sent to make the bottleneck link in congestion status.

5.2 The self-configuration feature of the AQSDN

In order to compare PCaQoS and DiffServ, we derive two virtual networks from the prototype system. One virtual network comprises the switch ES_1, CS_1, CS_2 and ES_3, and another comprises the switch ES_2, CS_1, CS_2 and ES_4. We input their QoS policy into the Policy DB. These two virtual networks use same flow classification and queue scheduling algorithm: the video flows are mapped to AF class and the background flows are mapped to AF and BE classes. Each class corresponds to one queue. The queue scheduling algorithm exploits PQ+WRR mode: the EF queue is superior to the AF queue; the AF/BE is scheduled by Weighted Round Robin (WRR).

These two virtual networks explore different packet marker and queue management. In the first virtual network, the *Single Rate Three Color Marker* (srTCM) is configured on the switch ES_1, and the *Weighted Random Early Detection* (WRED) is configured on the other switches. This virtual network is referred to as DiffServ based network. In the second virtual network, the PCaPM is configured on the ES_2, and PCaQM is configured on the other switches. This virtual network is referred to as PCaQoS based network. The parameter value of these QoS schemes depends on the service and the network context. The parameter *CIR* in the srTCM and PCaPM is set based on the average rate of the service during a period of time, and the *CBS* and *EBS* are set based on the max burst of the service. The minimum threshold th_{min}, the maximum threshold th_{max} and the maximum drop probability p_{max} are set based on the congestion on the link. The configurations of the srTCM and PCaPM are set

Table II *The flow attributions after encoding of ten video services*

No.	video	frames	AR1	AR2cri	Burstiness
1	akiyo	300	139	86	low
2	news	300	282	127	low
3	hall	300	302	110	low
4	Container	300	266	151	low
5	Highway	2000	254	65	middle
6	bridge_close	2000	446	151	low
7	Foreman	300	488	145	high
8	Paris	1065	535	258	high
9	Coastguard	300	966	192	high
10	Mobile	300	1525	407	low

Note: 1. AR - the average information rate of the stream (KB/s).
2. ARcri- the average information rate of critical packets (KB/s).

Table III *The configurations of the srTCM and PCaPM for ten video flows respectively*

video	CIR (Kbit/s)	CBS (bytes)	EBS (bytes)
Akiyo	140	9000	15000
News	285	15000	22000
Hall	310	12500	18500
Container	280	17000	34000
Highway	270	20000	40000
bridge_close	475	20000	30000
Foreman	490	30000	50000
Paris	560	45000	70000
coastguard	1000	50000	100000
mobile	1600	80000	160000

Table IV *The configurations of WRED and PCaQM* (*in the case that the maximize queue length is 100*)

Color		Mnimum *Threshold* (th_{min})	*Maximum Threshold* (th_{max})	*maximum* drop probability (p_{max})
WRED	PCaQM			
green	dark green green light green	55	90	0.01
yellow	cyan yellow	35	60	0.2
red	orange red dark red	10	40	0.8

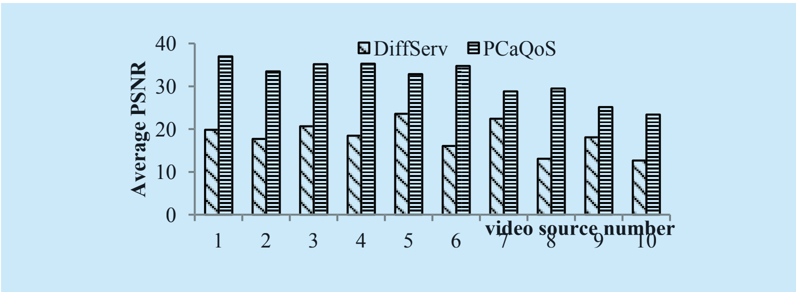

Fig.8 *The average PSNR comparisons of ten pairs of video services across PCaQoS and DiffServ based network respectively*

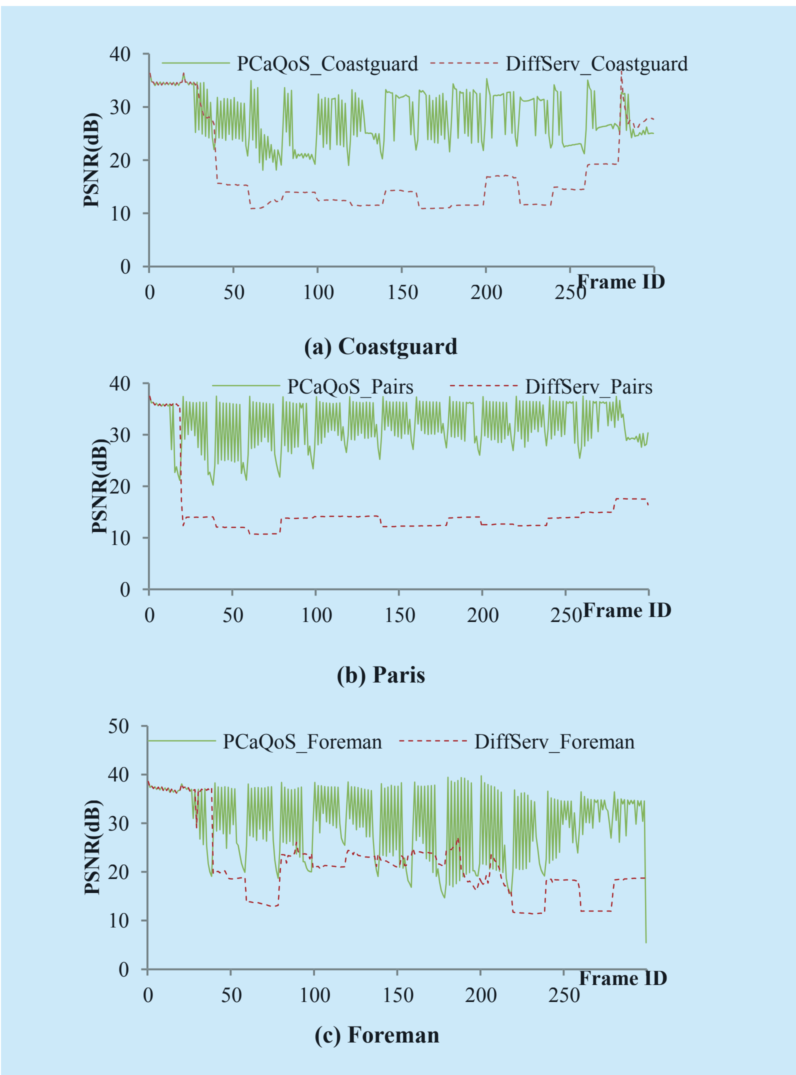

Fig.9 *The PSNR comparisons of 300 continuous frames of three pairs videos across PCaQoS and DiffServ basded network respectively*

as shown in Table III, and the configurations of the WRED and PCaQM are set as shown in Table IV.

5.3 The comparison of PCaQoS and DiffServ

Fig. 8 presents the average PSNR of each video after across PCaQoS and DiffServ based network respectively. It shows that the PCaQoS provides the higher video quality (avgPSNR) than the DiffServ in general. Furthermore we select 300 continuous frames from three videos after across the PCaQoS and DiffServ based network respectively, and analyze their PSNR. Fig. 9 presents the PSNRs of these frames. We also capture the 155th frames of these videos, and compare them with the corresponding frames of the original files. Fig. 10 presents the snapshot of the 155th frame of these three sets of videos. Obviously, Fig. 9 and 10 further support the conclusion that the PCaQoS provides the higher video quality than DiffServ.

VI. Conclusions

In the traditional IP network, resource utilization improvement and network QoS guaranteeing are very complicated for network operators. For solving this problem, research communities propose lot of ideas on it. One of them is to upgrade the network nodes with autonomic abilities. In addition, Software Defined Network provides the capability to implement network control and management functions by software. In this paper, the AQSDN architecture, which combines the advantages of the autonomic network management and the SDN technologies, is proposed. A novel QoS model which is called Packet Context-aware QoS (PCaQoS) model is also presented based on the AQSDN architecture. At the end, the self-configuration feature of the AQSDN and the enhancement of video quality of the PCaQoS model are verified.

Although our algorithms have shown the feasibility in a simple network environment, further verification research in more complicated network environments and the extendibility evaluation of PCaQoS model in AQSDN architecture are our future research works.

Acknowledgements

This work was supported in part by the National High Technology Research and Development Program (863 Program) of China under Grant No. 2011AA01A101, No.2013AA013303, No.2013AA013301and National Natural science foundation of China No. 61370197 & 61271041.

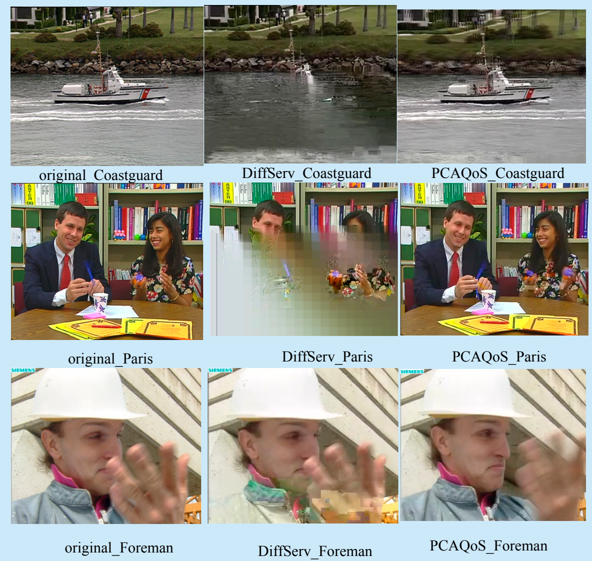

Fig.10 *The screenshot comparisons of the 155th frame of three sets of videos across PCaQoS and DiffServ based network respectively and their original images*

References

[1] Sivaraman, Anirudh, et al. No silver bullet: extending SDN to the data plane. Proceedings of the Twelfth ACM Workshop on Hot Topics in Networks. ACM, 2013.

[2] Kephart J O, Chess D M. The vision of autonomic computing[J]. Computer, 2003, 36(1): 41-50.

[3] Miller. B. The autonomic computing edge: Can you CHOP up autonomic computing? Technical report, IBM, 2008. http://www.ibm.com/developerworks/autonomic/library/ac-edge4.

[4] EFIPSANS project. http://www.efipsans.org/.

[5] ANA (Autonomic Network Architecture) Project. http://www.ana-project.org/.

[6] HAGGLE Project. http://www.haggleproject.org/

[7] CASCADAS project. http://www.cascadas-project.org/.

[8] BIONETS project. http://www.bionets.eu/.

[9] OpenFlow Specification 1.3.0. https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.3.0.pdf.

[10] OpenFlow Management and Configuration Protocol 1.1.1. https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow-config/of-config-1-1-1.pdf.

[11] Civanlar S., Parlakisik M., Murat Tekalp et al, A QoS-enabled OpenFlow environment for scalable video streaming[C]. in Proc. IEEE GLOBECOM Workshops (GC Workshops), 2010: 351–356.

[12] Egilmez H E, Dane S T, Bagci K T, et al. OpenQoS: An OpenFlow controller design for multimedia delivery with end-to-end Quality of Service over Software-Defined Networks[C]//Signal & Information Processing Association Annual Summit and Conference (APSIPA ASC), 2012: 1-8.

[13] Bari M F, Chowdhury S R, Ahmed R, et al. PolicyCop: An Autonomic QoS Policy Enforcement Framework for Software Defined Networks[C]. Software Defined Networks for Future Networks and Services Conference (SDN4FNS), 2013, Trento, Italy.

[14] Kim, Wonho, et al. Automated and scalable QoS control for network convergence. Proc. INM/WREN 10 (2010): 1-1.

[15] Ishimori, Airton, et al. Control of Multiple Packet Schedulers for Improving QoS on OpenFlow/SDN Networking. Software Defined Networks (EWSDN), 2013 Second European Workshop on. IEEE, 2013.

[16] Tong ZHOU, GONG Xiangyang, Yannan HU, et al. PindSwitch: A SDN-based Protocol-independent Autonomic Flow Processing Platform. Globecom Workshops (GC Wkshps),2013:842-847.

[17] Abowd G D, Dey A K, Brown P J, et al. Towards a better understanding of context and context-awareness[C]//Handheld and ubiquitous computing. Springer Berlin Heidelberg, 1999: 304-307.

[18] Xiangyang, Gong, Wang Wendong, and Li Yan. "A Novel Packet Marking Algorithm in Context-aware Autonomic QoS Framework." CHINA COMMUNICATIONS, 2009: 98-107.

[19] M. Allalouf, Y. Shavitt, A simulation study of multi-color marking of TCP aggregates, in Proc. of 32nd IEEE Conference on Local Computer Networks (LCN 2007),. 2007:.376-386

[20] Z. Cao, Z. Wang, and E. Zegura, Rainbow fair queuing: Fair bandwidth sharing without per-flow state, in Proc. of INFOCOM 2000, the 19th Annual Joint Conference of the IEEE Computer and Communications Societies, 2000(2):922-931

[21] M. Allalouf, Y. Shavitt, A comparison of token-bucket based multi-color marking techniques, in Proc. of the 2006 ACM CoNEXT Conference, 2006:1-2

[22] N. Gude, T. Koponen, J. Pettit, B. Pfaff, M. Casado, N. McKeown, and S. Shenker. "NOX: towards an operating system for networks." ACM SIGCOMM Computer Communication Review 38.3 (2008): 105-110.
[23] CPqD. Ofsoftswitch13, available at https://github.com/CPqD/ ofsoftswitch13, Jan. 2014.
[24] JM 18.5, available at http://iphome.hhi.de/suehring/tml/, Jan. 2014.

Biographies

WANG Wendong, is a professor of State Key Lab of Networking and Switching Technology at Beijing University of Posts and Telecommunications (BUPT). His research interests include network QoS, future Internet architecture, and internet services.

QI Qinglei, is currently a Ph.D candidate at State Key Laboratory of Networking and Switching Technology of BUPT. His research interests include future Internet, and autonomic network management.

GONG Xiangyang, is a professor of the State Key Lab of Networking and Switching Technology of BUPT. His research interests are IP QoS, network security, advanced switching technologies and next generation Internet.

HU Yannan, is currently a Ph.D. candidate at State Key Laboratory of Networking and Switching Technology of BUPT. His research interests include future Internet, Software-Defined Networking and network optimization.

QUE Xirong, is currently an associate professor of State Key Laboratory of Networking and Switching Technology at Beijing University of Posts and Telecommunications (BUPT). Her research interests include: IP QoS, future internet architecture and internet services.