Towards a Real Application-aware Network

Leo Caldarola[1], Amine Choukir[1], Davide Cuda[1], Marco Dondero[1], Domenico Ficara[1], Roberto Muccifora[1], Libor Polčák[2] and Antonio Trifilo[1]

[1]*Cisco System Sarl, Rolle, Switzerland*

[2]*Faculty of Information Technology, Brno University of Technology, Božetěchova 2, 612 66 Brno, Czech Republic*

Keywords: Software Defined Network, OnePK, OpenFlow, Application Awareness.

Abstract: As the number of network services and applications increases, each one of them showing different requirements in terms of bandwidth and latency, it becomes critical for network operators to identify these applications. This paper builds up on the principles of Software Defined Networking, proposing a novel Application-Aware Network architecture, which is able to directly handle applications and their requirements; abstracting the complexity of dealing with network flows. Since metadata about applications are signalled either directly from the endpoints or from application managers, the information is precise and the approach can also be used for encrypted traffic. The benefits of the Application-Aware Approach for the network flow management is demonstrated in real network by two proof-of-concept services: aimed on quality of experience and load balancing.

1 INTRODUCTION

The number of network services and applications is constantly growing. As each application shows different requirements in terms of bandwidth and latency, it becomes critical for network operators to be able to map traffic to applications. This paper builds up on the principles of Software Defined Networking; it proposes a novel Application-Aware Network architecture that is able to handle the requirements of different applications. The main feature of the architecture is the abstraction from the complexity of dealing directly with network flows.

Getting the right insight into the different applications allows network operators to plan their network capacity and policies to deliver the right quality of service (QoS) satisfying the application requirements. For example, applications such as VoIP and video conferencing need low jitter and latency whereas peer-to-peer file transfer require high throughput to minimize the download time. In this context, developing ad-hoc solutions is not cost effective and lacks flexibility since a change in the traffic requirements might lead to a complete reconfiguration of all middle boxes.

Currently, the predominant way of managing and configuring network services is on a per-network-node and per-endpoint type basis, irrespective of the commonalities that traffic patterns exhibit. Another painful point is the decentralised application priority/policy configuration. A good example is the deployment of a voice solution. The solution usually require the deployment of a specific application manager (e.g. SIP/H.323 call manager), which allows for QoS setting such as DSCP marking to be configured and pushed to the respective endpoints. However, network administrators choose to trust or distrust the priority set by the endpoint depending on the type of endpoint and the administrative domain they belong to. A typical case is a hardware phone and a software phone: despite both providing fundamentally the same function and present the same requirements, the traffic originating from the software phone is usually distrusted as it runs on desktop PCs.

This configuration and management model is error prone, not flexible and cumbersome. Network vendors proposed different solutions to make the network application-aware, such as Medianet[1], Application Visibility and Control[2] or Junos Application Aware[3]. A common characteristic of these solutions

[1]http://www.cisco.com/web/solutions/trends/medianet/index.html

[2]http://www.cisco.com/c/en/us/products/routers/avc_control.html

[3]http://www.juniper.net/us/en/products-services/network-edge-services/service-control/junos-application-aware/

Caldarola L., Choukir A., Cuda D., Dondero M., Ficara D., Muccifora R., Polčák L. and Trifilo A..
Towards a Real Application-aware Network.
DOI: 10.5220/0005473800050012
In *Proceedings of the 6th International Conference on Data Communication Networking* (DCNET-2015), pages 5-12
ISBN: 978-989-758-112-0

is the decoupling of flow identification from flow policies (prioritization, routing and others) through application specific tags. These tags can be either explicitly signaled as in the case of Medianet or locally produced by deep packet inspection as in the case of the Junos Application Aware solution. Based on the information about the requirements of the flows gathered from these tags, network nodes can decide the policies to be applied; in principle, this allows for a smooth collaboration between the applications and the network. However, today these solutions are mainly vendor specific. The lack of standards leads to poor or non-existing interoperability.

Recently, Software Defined Network (SDN) emerged (McKeown et al., 2008) as a new paradigm based on the separation of the network control logic from underlying physical routers and switches that forward traffic. In more details, SDN utilizes a centralized controller which exposes network functionalities through a set of well-defined APIs for services built on top of the controller. The combination of those two elements allows network operators and service providers to simplify network management operations giving them the possibility to offer new services to their customers. However, SDN based services have no insight on the relations between flows produced by the same application. Indeed, they still refer to traffic through its specific network protocol fields.

The major contribution of this paper is the proposal of the Application-Aware Network (AAN). The AAN network extends the controlled-network boundaries to include endpoints. Its core building block is the Application-Aware Controller (AAC) that combines different sources of information such as the network endpoints, hints extracted from the signaling packets and application managers (e.g.: a SIP gateway) to provide a uniform view of the application running through the network.

We evaluated the feasibility of the AAN approach with a testbed composed of real routers, in which the AAC was orchestrated by two application-aware services. Finally, the paper discusses how AAC can reduce the complexity of deploying a new network service while reducing their time to market.

This paper is organized as follows. Section 2 elaborates on application awareness in todays networks. Section 3 describes the details of the AAN architecture. The feasibility of the AAN is shown in Section 4 and Section 5 highlights the major benefits of the AAN approach. Finally, Section 6 concludes the paper and discusses possible extensions.

2 APPLICATION AWARENESS TODAY

Historically application awareness used heuristics (Dainotti et al., 2012; Bendrath, 2009), which inspect and infer flow characteristics. Heuristics may be based on port ranges (Fraleigh et al., 2003), IP subnetting (special network subnets for specific applications, e.g. phones), or deep packet inspection (DPI) (Moore and Papagiannaki, 2005), e.g. application level gateway. Port based solutions suffer (Moore and Papagiannaki, 2005) from port overloading and inconsistent port usage. IP subnetting solutions are error prone and result in network management hassle. DPI is computationally expensive and becomes a challenge with the wider adoption of encrypted signaling and secured traffic. In addition, DPI-provided insights are hardly shared between network nodes on the path of the flow.

Recently Cisco introduced Medianet (Wilkins, 2011): a solution that makes the network aware of the application traffic that it is carrying. In more details, Medianet endpoints leverage a proprietary middle layer (namely, Media Service Interface or MSI) to advertise flows Metadata (MD) tags, e.g. application name, traffic type, media type and business importance. Network devices such as routers and switches take local decisions to handle the advertised flows in the desired way.

2.1 Current Limitations of the Distributed Solution

To distribute Metadata tags along the path of a flow to different network nodes, an on-path signalling protocol such as RSVP needs to be used. This poses a number of challenges, some of which are exposed below:

- To be able to consume and leverage the MD tags, software (and potentially hardware) of network nodes needs to be upgraded. This entail a higher operational and development cost. It also increases the time to market and slows the adoption of application-aware features.
- The presence of middle boxes, such as firewall, NAT, transcoder, proxies or load balancers, can affect on-path signalling. Indeed, because middle boxes usually receive one input flow and regenerate it into one or more output flows, MD signaling has to be replicated/re-generated as well.
- The endpoint sourcing the flow and any network node along the path can contribute to the MD tags characterising the flow. This poses a consistency

problem as different node along the path get a different view of the flow.

- Resource consumption on the different network nodes and the bandwidth consumed by the on-path signalling: Each flow comes with a set of MD tags describing the flow and each node needs to store the MD tags so that different features on the network node can act upon them.

The centralised nature of SDN overcomes or mitigates all those problems.

2.2 Application Awareness in SDN

The usage of Software Defined Network technology to enable Application-Aware Networks is a fairly novel concept in both industry and academia. Jarschel et al. (Jarschel et al., 2013) showed a QoS boosting application that monitor the Youtube streaming experience. If certain quality thresholds are crossed, the application requests the BigSwitch SDN controller to enforce a routing change for the affected Youtube network flows.

Solutions such as that of Insieme (Application Visibility and Control) also combine Application Awareness and SDN: DataCenter deployment and management of applications, databases and their traffic is unified through a unique controller that is drivable with a set of APIs. Similarly, PlumGrid[4] provides network abstraction for data-centers with comparable promises to those of Insieme. Curtis et al. (Curtis et al., 2011) propose new ways to provide flow scheduling in Data Centers through OpenFlow. Das et al. (Das et al., 2011) adopt OpenFlow to dynamically aggregate flows according to flow properties such as bandwidth, latency, jitter; thus providing different levels of QoS to applications. In the same spirit of using OpenFlow to aggregate traffic according to application knowledge, Zhang et al. (Zhang et al., 2013) propose an extension to the OpenFlow protocol to enable application optimization in Optical Burst Switching networks. Finally, Bredel et al. (Bredel et al., 2013) show a viable way to coordinate Traffic-Engineering for traffic generated by scientific collaborations like the CERN's LHC.

The work we present in this paper is similar to that of Jarschel et al. (Jarschel et al., 2013). However, our approach is suited for a broader range of applications. In addition, we adopt carrier-grade IOS-XR hardware and Medianet, application-aware feature developed by Cisco.

[4]http://plumgrid.com

3 APPLICATION-AWARE NETWORK ARCHITECTURE

Controllers manage routers and switches in an SDN network via a southbound interface, whereas services (SDN applications) instruct controllers via a northbound interface. Unfortunately, all current SDN southbound protocols such as OpenFlow or OnePK limit the amount of information exposed to a controller and consequently to SDN services: only basic network indicators and metrics are available. The controller does not have information about the applications that produce the flows or about the relations between flows (e.g. the relation between a SIP control channel and associated RTP streams).

Network endpoints, such as PCs running the applications or application managers (AMs), i.e. the server applications, are a reliable source of information as they produce or handle the traffic. Therefore, a service that can utilize these additional pieces of information can achieve better flow handling than an SDN service leveraging only the network indicators. Application awareness allows for instance:

- To develop an AAN service extending network Quality of Service capabilities to match business-critical applications to flows and assign higher priorities to them;
- To precisely reserve network bandwidth according to the expected traffic rate of the flows generated by specific applications;
- To deploy application-dependent dynamic service chaining, i.e., routing of specific application flows through specific network services (e.g.: IPS/IDS, firewall, load-balancing or caching).

Application-Aware Networking (AAN) extends SDN to route flows in the network in a more effective or desirable way. Effectively, AAN extends the controlled-network by endpoints, end user applications and AMs. Figure 1 depicts the building blocks of AAN which are then described in details in the following subsections.

Our major contribution is in the abstraction on handling the network flows separately. Instead, AAN recognizes the relations between different flows originating from the same application. In this way, it is possible to handle traffic of the same application in a consistant manner. In addition, an application can tag each flow for a specific class of traffic. Consequently, the AAN can distinguish traffic classes of a single application with different flow handling requirements, e.g. a video stream and a stream with text metadata.

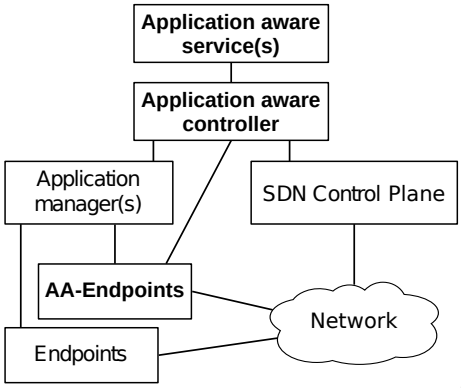

Figure 1: High level architecture of Application-Aware Network.

3.1 Network

In general, AAN is a hybrid network where SDN and non-SDN nodes coexist. SDN nodes constitute an overlay network interconnected by non-SDN network nodes. These SDN nodes are the points where network controlling services can provision resources and enforce policies, adapt to the application requirements and provide feedback to the controller(s).

In this paper, we focus only on AANs constituted entirely of SDN nodes. We leave the specification of the interaction between SDN and non-SDN nodes for future research.

3.2 SDN Control Plane

We do not assume any specific SDN controller. However, we assume that the controller provides at least basic OpenFlow functionalities, i.e., it is able to provide visibility and statistic about the flows crossing the SDN nodes and support basic actions to set QoS or select the output interface.

3.3 Endpoints and Application-aware Endpoints

In this paper, the word endpoint stands for an enduser device running applications. All endpoints communicate with the application manager (AM), which is responsible of managing the application logic, for instance, establishing calls or provide information about the user location.

Application-Aware endpoints (AA-endpoints) are equipped with a middle layer that allows them to directly communicate with the Application-Aware Controller (AAC). AA-endpoints export to the AAC information such as the established flows and application statistics. Furthermore, AA-endpoints export additional information such as the endpoint status (e.g., memory or CPU utilization) and type (e.g., smartphone or desktop device).

3.4 Application Manager

AMs (e.g., the Microsoft Exchange Server, Microsoft Lync or Cisco Unified Communications Manager for VoIP, VMWare's VSphere and OpenStack) are another source of information about application flows. A general trend is to move AMs toward cloud solutions coupled with provisioning of APIs to advertise and control flows.

3.5 Application-aware Controller and Services

AAC controls application-aware handling of the SDN network. From the perspective of the SDN controller(s), AAC is an SDN application. Nevertheless, AAC uses additional sources of information: AMs, AA-endpoints, or in general, any other source of information about flows. This flow information may include application name, traffic class, business priority and other attributes.

AAC correlates the flow information attributes and passes them to application-aware services (AA-services). An AA-service inserts application policies to the database. AAC transforms these policies into SDN rules and configures the SDN controller(s).

Hence, the AAC abstracts the complexity of handling flows of a single application in a consistent manner. AA-services deal directly with applications instead of the separate network flows.

The AAC provides a consolidated and coherent view of the applications and their flows, merging the information gathered from different sources. As different sources of flow information may provide different information, it is the responsibility of the AAC to merge the information consistently. AAC does so in accordance to the reliability of the source, e.g. information learnt from DPI engine are less reliable than information provided by an AM.

In addition to reliability, each metadata source operates with a specific trust model to prevent the possibility of an injection of false metadata. Therefore, the interface between AM or AA-Endpoints and the AAC should employ a two-way authentication mechanisms. In our experiments, we used Medianet authentication model (Cisco Systems, 2013), which provides two-way authentication.

Examples of AA-services are QoS policies, load balancing, routing, firewalls etc. Section 4 considers AA-services in detail.

4 TESTBED AND USE CASES

This section presents the testbed for AAN realised in our laboratory. The goal of the testbed was to evaluate the time frame required to integrate AAN with real hardware. We ran Cisco ASR 9000 routers running IOS-XR 5.1.1 that provides OpenFlow support as the SDN switches in the testbed.

To evaluate the feature velocity and applicability of AAN, we decided to use both Pox[5] and Open Daylight[6] as SDN controllers. To demonstrate the idea, we implemented AA-services for QoS management and load balancing, both of which benefits from the application-aware approach.

The testbed (depicted in the Figure 2) is composed of different types of endpoint devices and applications, including both proprietary and open sources devices. Two Cisco phones and two Jabber clients are registered with a Cisco Unified Communication Manager and adopted SIP to establish audio/video sessions. Phones and Jabber clients make video calls, thus each producing two media network flows for each call: audio and video. In addition, a video and audio stream were multiplexed in a single network flow from a server to a VLC[7] client. Jabber and VLC have been enhanced with the adoption of the MSI library (Cisco Systems, 2013), thus becoming AA-endpoints.

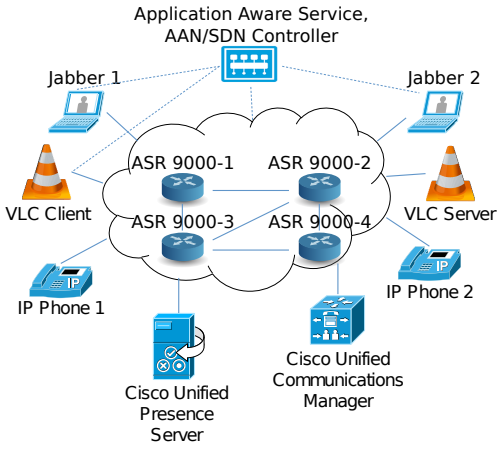

Figure 2: Topology used in our testbed.

The application-aware control plane is detailed in the Figure 3. Three sources of information about applications and their flows are present: AMs, AA-endpoints, and session parsers that gathered additional information from packets passed from the SDN controller.

[5]http://www.noxrepo.org/pox/about-pox/
[6]http://www.opendaylight.org/
[7]http://www.videolan.org/vlc/index.html

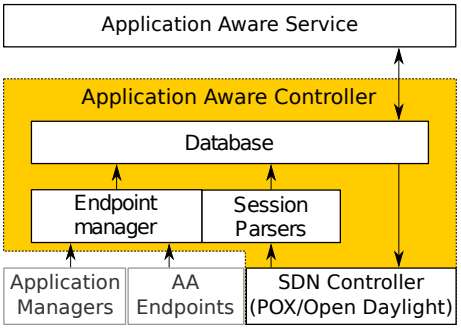

Figure 3: Application-aware control plane in the testbed.

The ASR 9000 routers are instructed (via OpenFlow) to punt SIP packets coming from the Cisco IP phones and the Jabber clients to the SDN controller. Those SIP packets are then passed to a Session Parser (SP) module, basically a DPI engine. Because Cisco phones are configured not to encrypt signaling traffic, their SIP packets are then parsed by the SP module. The SP module, in turn, collects SIP session details and pushes them to the common database thus mapping network flow details to application.

Jabber flows are instead announced by MSI to the AA-Endpoint manager that, similarly to SP, stores the Jabber application to network flows mapping into the common database. The same mechanism is adopted for the VLC client that announces the RTP flows carrying the video and audio through MSI to AAC.

Because information gathered from all sorts of sources is stored in the same database, the production and the consumption of the information is decoupled. Normalisation is then applied on all the acquired data, according to the degree of reliability and completeness of the sources. For example, SIP packets generated by Jabber clients are not only advertised by MSI but also by the SP module (if they are not encrypted). This happens because the SIP packets are punted to the SDN controller and handled to the SP module. Therefore normalisation enables for merging of the information coming from both sources, giving higher priority to the information provided by MSI as it is more reliable.

The testbed AAC treats the sources of information according to the following priorities:

- Application managers are operated by network administrators and are the most reliable source. Information about both encrypted and unencrypted traffic is visible. They cover only the applications provided inside the testbed.
- AA-Endpoints have a little bit lower reliability as the endpoint can provide false information as a result of a virus infection or malicious tampering with MSI library. Nevertheless, MSI signals

information about relations between flows. AA-Endpoints also provide information about both encrypted and unencrypted traffic.

- SP is the less reliable source as it cannot decrypt encrypted flows and there is a risk of false positive or false negative identification of a protocol.

The rest of this section describes AA-services tested in our testbed.

4.1 QoS Management

To demonstrate the benefits of AAN, we built a service that leverages the AAC in order to assign a QoS policy to enduser applications detected in the network. In two weeks, we were able to develop the AAC and the QoS management AA-service from scratch. Network administrators can influence the network behaviour and assign priorities to specific business critical applications (Choukir et al., 2013). AAN allows to set up the priorities in a convenient and network-wide consistent manner.

4.2 Path Load Balancing

The following experiments (Polčák, 2014) focused on routing and path load balancing. The goal of this AA-service was to show that it is possible to operate the AAN network even if not all flows are recognized by the AAC.

Two sets of applications were operated in the testbed: 1) those running on AA-endpoints and 2) those that initiated flows that were not advertised to the AAC, i.e. those that created background traffic.

The aim of the AA-service was to load balance the traffic of the business-critical applications across all possible paths between the source and destination. In case the AA-service detected that a link carrying priority traffic was congested, the AA-service automatically rerouted the traffic to links with sparse capacity. Hence, even if the link was congested because of traffic from legacy endpoints (that were not application-aware), the AAN architecture allowed to ensure that the priority traffic reaches its destination without unnecessary retransmits.

4.3 Other Services

The presented QoS management and load balancing service is just an example of how network management can be made easy when the network can identify and directly handle applications rather than flows. Our approach can be applied to any network feature by custom services providing path computation, routing, etc. The underlying infrastructure that gives application awareness remains the same.

5 CONSIDERATIONS

In this section, we compare the AAN to a classical distributed network and to an SDN network without application awareness. Moreover, we describe a possible behaviour of an AAN in typical network scenarios.

5.1 Evaluation

Being built on top of an SDN network, the AAN maintains the main advantages of the SDN architecture while reducing the complexity of applying the correct policies to related flows (typically belonging to a single application).

When expressing a policy, a combination of flow characteristics is used to determine the traffic for which the policy is enforced. The combination of these characteristics defines a class of traffic (a.k.a. class-map). IP addresses, port numbers and application names are examples of such characteristics. We define the complexity as the number of network nodes where a class-map must be explicitly declared multiplied by the number of entries in the policy itself. Let us consider a network of N nodes with the number of applications denoted as A, each application carrying F different flows in average. The total number of rules or policies configured and managed in the network may be evaluated in the following way:

- In a conventional network, a rule has to be distributed for each flow of each application on every network node. Therefore, the total complexity can be evaluated as $A \times F \times N$.
- In case of an application-aware distributed network (AA-distributed network), for instance Metadata-aware network, the complexity reduces to $A \times N$ since the network can deal directly with applications instead of flows.
- In SDN, the controller (or the set of cooperating controllers) abstracts the complexity of programming each node separately. However, a rule for each flow must be explicitly declared; thus, the total complexity in this case becomes $A \times F$.
- In AAN, application policies are handled in a straightforward manner since the complexity with flows is abstracted by the AAC; thus, the number of class-maps that need to be specified reduces to A, the number of applications.

Table 1: Comparison of available solutions to enforce policies for specific applications.

	Conventional	**AA-Distributed**	**SDN**	**AAN**
Policy complexity	$A \times F \times N$	$A \times N$	$A \times F$	A
Feature Velocity	Slow	Slow	Fast	Fast
Consistent behavior	No	No	Yes	Yes
Operational cost	High	Medium	Low	Lowest

Feature velocity in distributed environments depends on the number of different operating systems and different hardware platforms to be supported. In the case of SDN and AAN, the decoupling of the capabilities exposed by the network platforms and the flow characteristics allows for an increased velocity in delivering application-aware features. This is achieved through writing software (service) once rather than integrating software into different network platforms. Hence, the feature velocity of SDN and AAN is fast compared to conventional or AA-distributed networks.

The control plane abstraction ensures that there is a single set of policies applied to all SDN switches in the network. In comparison, the cumbersome management of consistent rules on each node might result in inconsistent configuration and unpredictable behaviour of the network. Therefore, the operator has to maintain N configurations.

In a conventional deployment, the policy complexity often results to a high number of class-map rules. The class-map rules has to be managed directly on the network nodes in consistent manner. Maintenance of the consistency incorporates high operational costs accompanied with with slow feature velocity. Despite lower complexity of AA-distributed deployment, it still requires medium operational costs as it requires big effort to maintain consistent behaviour. The decoupling of control and data plane and the abstracted complexity of the network lowers the operation costs of both SDN and AAN. Nevertheless, AAN additionally treats application flows in a consistent manner.

The comparison is summarized in Table 1.

5.2 Typical Scenarios

Let us consider some specific scenarios to illustrate the benefits deriving from the deployment of an AAN.

5.2.1 Network Congestion

Although congestion of a link is easy to detect in a distributed network, none or few network technologies exist to react and reroute flows in a timely way.

With SDN, the controller can immediately reroute the traffic depending on the network data as queue occupancy or interface load increase. However, the information available to an SDN controller are limited to the transport layer.

Compared to an SDN network, AAN offers two main advantages. Firstly, the routing operations can be based on application requirements; thus all flows of a single application are handled in a consistent way without the complexity of matching flows by the SDN application. The second, and more significant advantage, is that the AAC can potentially access application specific metrics and eventually, it can detect the looming suffering of an application in advance.

5.2.2 Topology Change

In the event of a topology change that impacts the path of flows with specific requirements, a regular Medianet solution requires a new distribution of Medianet tagging in the new path. If the change reoccurs with some frequency because of physical impairments, it can result in a flapping behavior and degraded quality. AAC with complete network and application view can instead choose a stable path that avoids the abovementioned issues.

5.2.3 Application Policy Change

In a Medianet network, whenever application requirements change, the new requirements are pushed by administrator to the network devices through configuration. Administrator has to foresee possible problems in advance so that the change targets only affected devices; and avoids changes to unrelated traffic so that a single change does not trigger an avalanche of changes. In AAN, the new application requirements are enforced on all affected devices by the AAC, which has the complete view of all flows and their paths.

5.3 Migration

As AAN raises several requirements on AA-endpoints, it is not expected that the migration is instant and straightforward. Instead, several progressive steps can ease AAN deployment.

Firstly, plug-ins to critical application managers can advertise instantly information about flows of all hosts related to the specific application. As discussed

in Section 4, the benefits of AAN may be utilised even if the metadata are available for only a limited number of flows. Hence it is not necessary to migrate all endpoints to leverage the benefits of AAN.

In case it is not possible to deploy a plug-in for a business critical application, unreliable DPI-based source of information or similar techniques can provide some information about flows in the network.

Finally, IETF aims to establish Application Enabled Open Networking working group (IETF, 2014). The current charter proposal includes standardisation of Metadata signaling. Once standardized, the amount of applications supporting Metadata signaling should raise, and consequently, the deployment of AAN for arbitrary application should be easier.

6 CONCLUSION

User expectations of network applications quality are high even for multimedia streaming. The ever growing number of applications communicating through the network environment highlights the need for constructing networks that are capable of achieving application-level control of traffic to ensure appropriate QoE. Traditional means of ensuring appropriate QoS for applications, such as DPI, have a high complexity and operational costs while the behaviour is not consistent across the whole network. SDN-based approach, on the other hand, provides consistent behaviour and fast feature velocity but it does not automatically treat related flows in a consistent way.

This paper presented a novel paradigm of Application-Aware Network (AAN) based on industry and open standards. The solution we presented achieves a number of major goals:

- AAN improves final user QoE based on the feedback from applications received through MSI,
- AAN brings application awareness to Cisco IOS-XR routers such as ASR 9000,
- AAN extends network to include endpoints and endpoint applications,
- AAN brings fast application velocity (the whole project has been developed within two weeks),
- AAN enables application-based policy enforcement.

Future extensions and improvement of this work are under consideration and development. Among these, we are investigating the ability to provide hints to the application mediator through MSI and additional abstraction levels that can help in solving contention and inconsistencies between applications in the network.

REFERENCES

Bendrath, R. (2009). Global technology trends and national regulation: Explaining variation in the governance of deep packet inspection. Technical report, Delft University of Technology. Paper originally prepared for the International Studies Annual Convention.

Bredel, M., Barczyk, A., and Newman, H. (2013). Application-aware traffic engineering for wide area networks using openflow. In *SuperComputing Conference, Emerging Technologies*.

Choukir, A., Caldarola, L., Cuda, D., Dondero, M., Ficara, D., Muccifora, R., Polčák, L., and Trifilo, A. (2013). *Towards a real application aware network*. http://youtu.be/QHYPhAhIwVw.

Cisco Systems (2013). Cisco MSI Deployment Guide. http://www.cisco.com/web/solutions/medianet/docs/Cisco_MSI_Installation_Guide.pdf.

Curtis, A. R., Kim, W., and Yalagandula, P. (2011). Mahout: low-overhead datacenter traffic management using end-host-based elephant detection. In *IEEE INFOCOM*.

Dainotti, A., Pescape, A., and Claffy, K. (2012). Issues and future directions in traffic classification. *Network, IEEE*, 26(1):35–40.

Das, S., Yiakoumis, Y., Parulkar, G., and McKeown, N. (2011). Application-aware aggregation and traffic engineering in a converged packet-circuit network. In *Optical Fiber Communication Conference and Exposition (OFC/NFOEC) and the National Fiber Optic Engineers Conference*.

Fraleigh, C., Moon, S., Lyles, B., Cotton, C., Khan, M., Moll, D., Rockell, R., Seely, T., and Diot, S. (2003). Packet-level traffic measurements from the sprint ip backbone. *Network, IEEE*, 17(6):6–16.

IETF (2014). Application Enabled Open Networking (AEON). https://sites.google.com/site/aeonsignalling/.

Jarschel, M., Wamser, F., Hohn, T., Zinner, T., and Tran-Gia, P. (2013). SDN-based Application-Aware Networking on the Example of YouTube Video Streaming. In *European Workshop on Software Defined Networks*.

McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S., and Turner, J. (2008). Openflow: Enabling innovation in campus networks. *SIGCOMM Comput. Commun. Rev.*, 38(2):69–74.

Moore, A. W. and Papagiannaki, K. (2005). Toward the accurate identification of network applications. In Dovrolis, C., editor, *Passive and Active Network Measurement*, volume 3431 of *Lecture Notes in Computer Science*, pages 41–54. Springer Berlin Heidelberg.

Polčák, L. (2014). *Integration of SDN and Medianet Metadata*. http://youtu.be/CqDYn4-DKn8.

Wilkins, S. (2011). *Designing for Cisco Internetwork Solutions (DESGN) Foundation Learning Guide (CCDA DESGN 640-864)*. Pearson Education.

Zhang, D., Mai, S., Guo, H., TakehiroTsuritani, Wu, J., and Morita, I. (2013). Openflow-based control plane for the application-aware lobs network. In *OptoElectronics and Communications Conference*.