IET Networks

Special Issue on Software Defined Networking

IET Journals
The Institution of Engineering and Technology

Service-level agreement aggregation for quality of service-aware federated cloud networking

ISSN 2047-4954
Received on 30th September 2014
Accepted on 9th December 2014
doi: 10.1049/iet-net.2014.0104
www.ietdl.org

Alexander Stanik ✉, *Marc Koerner*, *Odej Kao*

Complex and Distributed IT Systems, Technische Universität Berlin, Berlin, Germany
✉ *E-mail: alexander.stanik@tu-berlin.de*

Abstract: Since the cloud paradigm becomes increasingly popular for dynamic resources allocation, the flexibility of a cloud is still limited regarding network services and their autonomous federation between different providers. The following architectural approach introduces a generic layered model to orchestrate and federate heterogeneous networks. In particular, an architecture is presented that enables quality of service (QoS) aware parameterisation of network resources in a cloud infrastructure of a single data-centre as well as for a federation. Furthermore, this architecture uses a service-level agreement (SLA) protocol and language to expose key performance indicators and to negotiate appropriate QoS constrains that are applied to the virtually sliced underlying network substrate. In this way, capabilities of the orchestration and the current utilisation of the network are building the foundation for dynamic negotiated SLAs and the within-guaranteed QoS network resources. Therefore an aggregation mechanism is illustrated for merging service-level objectives and for guaranteeing a single SLA that specifies obligations and responsibilities of all participants.

1 Introduction

The cloud paradigm has changed the IT landscape in recent years because of powerful cloud middlewares that provide cloud services on-demand. Although infrastructure-as-a-service (IaaS) builds the foundation, where resources are virtualised and provided to the paying customer, platform-as-a-service and software-as-a-service are built on top. Virtualisation is a powerful enabling technology in this domain, allowing flexible and on-demand provisioning of IT resources from the users' perspective. Moreover, virtualisation is also simplifying the management of the data centre for the cloud operator. However, the flexibility of cloud computing is very limited when it comes to accessing or configuring network resources. For example, the network topology behind the allocated virtual machines (VMs) in the cloud is pretty opaque, inflexible and provides only basic network policies [1].

Network virtualisation is a key technology for deploying next generation networks and applications [2–5]. Virtual networks (VNs) can be deployed over a shared network infrastructure, while ensuring isolation of the traffic among concurrent VNs. This technology allows companies to specify customised virtual topologies according to their requirements. Software-defined networking (SDN) complements the network virtualisation approach by offering programmability features to the owners of VNs [6]. Thus, network functionalities (e.g. routing, firewalls etc.), which were traditionally implemented by vendors within the hardware, can now be realised in software. This provides a great degree of programmability, since data centre providers may develop their own network functions and thus control their VNs according to their needs. However, the full range of SDN capabilities is not utilised in today's cloud middlewares. Furthermore, SDN is used to facilitate future data centre networks, but the benefits are often not passed to the paying customer.

The presented approach will improve cloud middlewares by integrating SDN functionalities with appropriate business models, which will allow cloud providers to sell additional network services to their customers. One issue that this approach has in focus is to support quality of service (QoS) characteristics to the virtual environment of the deployed VMs. Consequently, the provider of VNs can apply guarantees in service-level agreements (SLAs) to their users. Thus, customers will be able to configure the VN according to their requirements, which results in an SLA that not only describes the service levels, the objectives and the prices but can also be used to monitor the fulfilment or violations of advertised guarantees.

This architecture is using programmable networking mechanisms to provide SLA guarantees for users of virtualised networks. Thus, this approach is called ProgNET. To achieve QoS guarantees on VNs, ProgNET strives to introduce a novel data link layer mechanism that will solve the data link layer loss problems that are experienced in today's VNs. Furthermore, in the lossless networking environment of ProgNET, new tools and mechanisms will be implemented to manage and enforce customer SLAs within the virtualised network substrate. The envisaged SLA mechanisms are designed not only for managing the networking resources of a single provider, but also for managing the resources within federations of providers, where the end-to-end user services may involve resources drawn from multiple service providers in a transparent way to the user.

The establishment of a QoS-aware route between two cloud providers involves several independent network providers. Thus, an SLA chain has to be discovered, negotiated step-by-step and instantiated if all parties come to a common agreement. Here, the customer should just define the service-level requirements (SLR) and should have no additional obligations in this process. To facilitate the understanding and to include all information in single agreement that is presented to the customer, an SLA aggregation for network services is presented and further explained.

The remaining sections are organised as follows: Section 2 discusses related work in the area of SLA management, aggregation and QoS for cloud federations followed by Section 3 that explains the proposed architectural approach. Section 4 explains SLA negotiation and creation processes and presents automated aggregation mechanisms. Finally, in Section 6, the concept and the mechanisms are concluded.

2 Related work

Xiao and Ni [7] described already in 1999 the gab of on-demand Internet QoS. They presented a framework that used the resource

© The Institution of Engineering and Technology 2015

reservation protocol (RSVP) for dynamical SLA instantiation and described their integration of multiprotocol label switching (MPLS) and constrained-based routing. They compared their framework to an asychronous transfer model (ATM) network and explained how they established SLAs between Internet service provider (ISP) and the customer. The use of RSVP was a less-than-ideal solution, because this protocol was not designed to express SLAs or instantiate SLAs. The reason for choosing this protocol was probably caused by the circumstance that no standard or other suitable protocol for a dynamic negotiation, creation and evaluation of SLAs existed. The approach presented in the following sections uses WS-Agreement and its extension WS-Agreement Negotiation, both of them standardised and applied in several projects like [8–12].

While [7] focuses mainly on bandwidth allocation by using a bandwidth broker, the approach of [13] has its scope on availability. Pongpaibool and Kim again introduced two path selection algorithms: the maximum availability routing (MAR) and the joint maximum availability routing (JMAR). While MAR algorithm finds a single path with the highest end-to-end availability, the JMAR algorithm simultaneously finds two paths (primary and backup paths) with the highest jointly computed end-to-end availability [13]. They assume that each ISP is responsible for its own domain and SLAs are instantiated with just four parameters. In contrast to our approach, they do not specify how SLAs are instantiated and SLA violations are not detected. Additionally, this paper presents an approach for multiple cloud provider with unspecified amount of VMs.

In general, [13] focus on an end-to-end connection between two hosts directly connected to the ISP. Lakshminarayanan *et al.* [14] extended this approach from a connected host to a connected customer or network. They introduced virtual links that allow ISPs to retain control over the flow of traffic in their networks [14]. Also, hardware-based approaches among the commercial solutions for hybrid cloud networks, which are Cisco Nexus 1000 V [15] and Nexus 6000 V [16], InterCloud switches exist. However, although both switches can be used for establishment of VNs across hybrid clouds and thus support easy migration of VMs and security of inter-cloud network traffic, they do not support SLA guarantees within the hybrid environment. Thus, the end-to-end flows spanning remote data centres are not controlled with specific QoS configurations. In comparison, our approach supports QoS support to end-to-end flows within a VN that spans across multiple data centres within a cloud federation.

The work described in [17] presents an approach for providing QoS guarantees to VNs that connect VMs within a cloud federation. In this work, the carrier network that is used to interconnect remote cloud sites is considered as a MPLS transport domain; thus the problem tackled within the paper is how to use MPLS's QoS classes to provide support for isolation of VNs within a certain MPLS QoS class. The solution given is a programmable admission control mechanism, transparent to MPLS, by using OpenFlow (OF) as the admission controller to the MPLS domain. Thus, edge OF controllers combine information of the state of the requested VNs with QoS information from the MPLS carrier network in order to tag appropriately the packets that belong to each individual VN. The benefit of this approach is that it does not require any modification within the MPLS domain. Compared to [17], ProgNET does not make any assumption on the underlying carrier networks that are employed to interconnect cloud sites within a cloud federation. Thus, this approach introduces the abstraction of the ProgNET network manager component within ProgNET's layered architecture so that our approach does not depend on a specific inter-domain protocol as well as it does not make any assumptions about the networks that are used to interconnect remote cloud sites. Therefore in our generic architecture, an implementation of a ProgNET 'WAN Driver' could be the one proposed by [17] for an MPLS domain. In addition, ProgNET would enhance the work presented in [17] with the capability of negotiating network-level SLAs with the MPLS carrier network provider, which would be needed in cases of network congestions within the MPLS network.

The works of [8–11, 18, 19] focus on SLA characteristics of a VM within the federated cloud, such as the amount of memory and the number of cores of the VM or the location of the VM. However, apart from providing SLA support at the resource-level of VMs within a federation, other approaches does not cater for QoS guarantees in networks nor cloud-based network federations. Our work focused on providing QoS guarantees in interconnected VMs within cloud federations by controlling the SDN approach and the QoS support of the OF protocol so that the networking resources can be configured in order to reach the requirements of customers.

3 Architecture

The OF protocol and derivatives enable an innovative mechanism to control large networks and high performance networks, as often used in cloud service provider facilities. Moreover, OF provides the opportunity to create a wide range of new applications and the corresponding use-cases based on providing the network and the related capacity itself as a service. Configuration and behaviour of SDN can be controlled by one entity and changed on demand. The properties of the underlying network and its connected servers can therefore be treated as a resource.

SLAs are established and managed between cloud service provider and consumers and are an instrument to describe a service and its quality. This issue is of crucial importance for companies whose success depends on the advertised QoS that the service provider delivers. However, SLAs are mostly formal language written by lawyers that is static and just published on the providers' websites. This might be enough to cover the majority of customers, but it is often an obstacle for companies that would like to rent computing and storage resources with additional representations and warranties. SLAs for network services do not exist in today's cloud landscape, because they are hard to describe in programmatic fashion, and guarantees are not advertised. This architecture enhances IaaS with mechanisms for automated negotiation and creation of SLAs for network services that deliver QoS guarantees (a) between VMs in a single data centre cloud, (b) to external cloud services located anywhere in the Internet and (c) for interconnections between several heterogeneous cloud environments.

To negotiate, create and evaluate SLAs in a machine-readable form, the WS-Agreement standard and its extension WS-Agreement Negotiation, as protocol and language, are used. Thus, customers are able to define their requirement in a standardised way and request services dynamically from an independent service provider. Furthermore, the ProgNET approach will extend cloud middlewares with OF capabilities to apply negotiated and guaranteed quality constrains to the underlying cloud SDN substrate.

The combination of WS-Agreement (Negotiation) and OF offers the opportunity for new use-cases and business models as well. The approach presented in this paper demonstrates new capabilities based on a shared virtual Ethernet circuit overlay. The basic idea is to enable customers to query and book available network routes between local and external hosts including guaranteed network characteristics. To deploy a requested QoS-based network, the customer and the cloud provider are negotiating service-level objectives (SLO), which are applied when both come to a common agreement. To achieve this goal, we designed a layered architecture that is depicted in Fig. 1.

The proposed middleware software stack is composed of three layers. The top layer is the 'Front-end', which is a graphical-user interface that allows customers in a human-friendly way to specify network requirements, configure network topologies and to negotiate SLOs in order to create an SLA and establish an appropriate network environment. Furthermore, this component provides also monitoring mechanism for detecting SLA violations and notification mechanism to inform the customer and the provider about inadequate behaviours of guaranteed resources. The 'Front-end', however, is not managing any network resources directly, but just aspects regarding the establishment of network

© The Institution of Engineering and Technology 2015

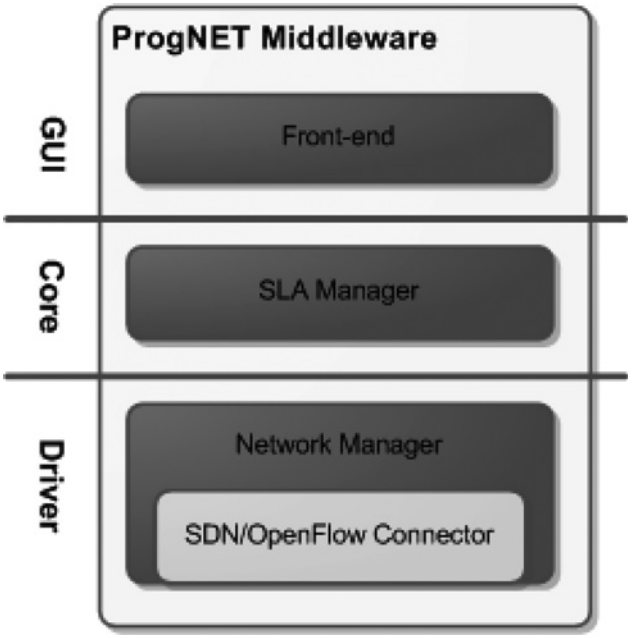

Fig. 1 *ProgNET stack*

services based on SLAs. All properties related to network resources are delegated to the SLA manager, who collects information about the underlying network from the network manager. Based on this information the SLA manager provides functionalities to negotiate and create aggregated SLAs and then appropriately establish QoS-based network environments in a single data centre and to external services. Furthermore, this SLA manager enables federated networking between several autonomous clouds with heterogeneous infrastructures.

The key components to manage the local and global available network capabilities and its particular QoS allocation is the network manager. This component is building the backbone of the entire architecture. This architecture addresses mechanisms based on the centralised control plan in SDN substrates and the opportunity to enforce QoS in data centre as well as comprehensive SDN networks. For pure SDN substrates this architecture considers all opportunities for QoS enforcements based on OF capabilities (e.g. metre tables) of the latest OF standards like OF 1.3 and beyond [20].

In contrast to a network manager installation in a cloud environment, the network manager in a network provider's infrastructure is an abstraction for the particular external carrier or Internet exchange point (IXP)-related network and directly expose the relevant QoS capabilities as external service through the provider's SLA manager. Besides exposing SLA-based networking capabilities to the outside world, the SLA manager checks routes by discovering external SLA managers. To provide a reasonable solution, ProgNET considers also solutions based on direct circuits for cloud data centres through provider networks to directly connect to the IXP, which is exposing and integrating a service architecture based on this abstraction layer to expose and sell this QoS.

4 SLA aggregation

From a business perspective it is essential that specific services are provided according to predefined service levels (e.g. compliance with minimum bandwidth or service availability). Therefore the ProgNET system gives providers the opportunity to negotiate SLAs among each other in order to provide a single SLA to the customer. This single SLA contains all expectations and obligations for the business relationship.

Each cloud provider has a relationship and basic contracts with the network provider who is providing the network connectivity to the next IXP and the networks in between. This relationship which is known by both sides is based on a fundamental contract. This contract is for instance also formulating network resources that can be requested by the customer through a provider's ProgNET instance. With this information the SLA manager can be configured in order to provide federated networking.

In the simplest case of a single cloud infrastructure, the SLA manager is waiting for requests from customers and negotiates the conditions for the local network environment. The network manager again provides information about the local network utilisation and available capacities to the SLA manager. If both the customer and the provider agree to the negotiated conditions, an SLA is created that serves as a formal contract. Afterwards, the network connection is established between the VMs according to the conditions of the agreement and monitored in order to detect violations. In this scenario, an SLA in a single cloud infrastructure is established between the cloud provider and the customer. This relationship is specified analogue to the WS-Agreement specification in the 'Context' element. Therefore, an SLA is defined as follows:

Definition 1: A service-level agreement denoted as sla consists of a context and an n-tuple of terms denoted as set Term

$$\text{sla} := \{\text{context}, (\text{Term}_1, \ldots, \text{Term}_n)\} \tag{1}$$

In the complex case of federated networks, the SLA needs to be established for a network service that combines the local network capabilities provided by a local network manager with network services of external network provider. In particular, the SLA compliance of a single service may rely on SLA compliance of other services. Therefore, delivered SLAs of single provider can be divided into two classes of SLAs which are defined as follows:

Definition 2: A local service level agreement denoted as $\text{sla}_{\text{local}}$ is established between a customer and a provider, where the provider delivers services that are solely under the control and only owned by the provider itself.

Definition 3: An extended service level agreement denoted as $\text{sla}_{\text{extended}}$ is established between a customer and a provider, where the provider delivers services that are partially under the control of the provider and extended with or depending on third-party services of external provider.

Interdependencies of extended-SLAs are not visible to the customer, who negotiated a SLA with its service provider only. Hence, the customer defines the SLRs for a network environment between several VMs. Depending on the entry point, which is in fact the location of the Front-end and its corresponding SLA manager, the provider have to discover all possible routes between specified endpoints. For example, if a customer requests a route between two VMs in different cloud infrastructures, the customer can inquire an offer from both cloud providers or from an external network provider. Thus, the requested provider's SLA manager is responsible to create a service offer. This SLA manager has knowledge about all SLA managers that are directly connected and responsible for the network connected to the provider's infrastructure.

Fig. 2 illustrates a situation where the left-hand side provider's data centre is connected with two network providers. There, two approaches can be applied to discover appropriated routes: First, a directed acyclic graph (DAG) is built consisting of all SLA managers that can be used to establish the requested route. Secondary, the SLA manager requests both federated SLA manager that in turn requests recursivity route information from further federated SLA manager. In the first case, the entry point or root SLA manager is responsible to negotiate route characteristics, prices and all other conditions for agreement offers for each path

IET Netw., 2015, Vol. 4, Iss. 5, pp. 264–269
© The Institution of Engineering and Technology 2015

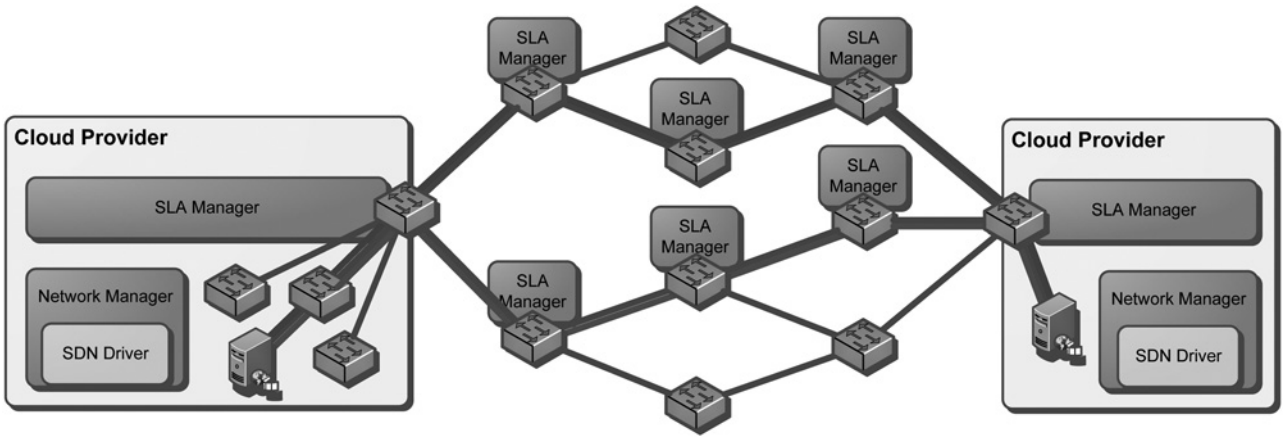

Fig. 2 *Example cloud federation network overlay*

through the DAG and with each provider's network on these routes. Moreover, after finding an appropriate route with agreed quality characteristics, an SLA needs to be instantiated and the promised routes have to be established. Thus, the SLA manager is also responsible in case of SLA violations and therefore have to monitor the route, to pay compensations and should try to switch to alternative routes if the promised guarantees of an SLA cannot be fulfilled. This is very difficult, because monitoring of external networks and above is impossible in terms of detecting the responsible party on the route. Therefore, the second approach is better and easier to be applicable for ProgNET. Here, the SLA manager evaluates a particular SLA with the help of measurements provided by the underlying network manager. In general, only network provider that make use of the ProgNET approach and which have a SLA manager instance up and running can be used to discover a route to an external VM. Furthermore, also the cloud provider on the right-hand side have to support this ProgNET approach in order to establish the last part of that chain between the network edge and the target VM.

If all requirements are met and a set of routes were discovered, the SLA negotiation process can be instantiated. Therefore, the requested SLA manager aggregates its local-SLA offers with the external-SLA offers of both federated network provider. These external-SLA offers are delivered from independent provider, but using similar or identical network paths delivered by the same ISPs, as illustrated in Fig. 3.

Terms in WS-Agreement are the main subjects of an agreement and an agreement offer. These terms can be a set of service descriptions, references to service descriptions and/or guarantee terms that define SLOs coupled with qualifying conditions that must be met. To aggregate terms of local-SLA offer to an extended-SLA offer, service descriptions or references of third-party provider are not relevant for the aggregation. Thus, terms are defined as follows:

Definition 4: A service term denoted as term with a service identifier i consists of a service level objective denoted as slo, a guaranteed service level threshold denoted as slt and qualifying condition denoted as qc

$$\text{Term}_i := \{\text{slo, slt, qc(slt, slo)} \mid \text{slo, slt} \in R,\ \text{qc} \in \text{QCS}\} \quad (2)$$

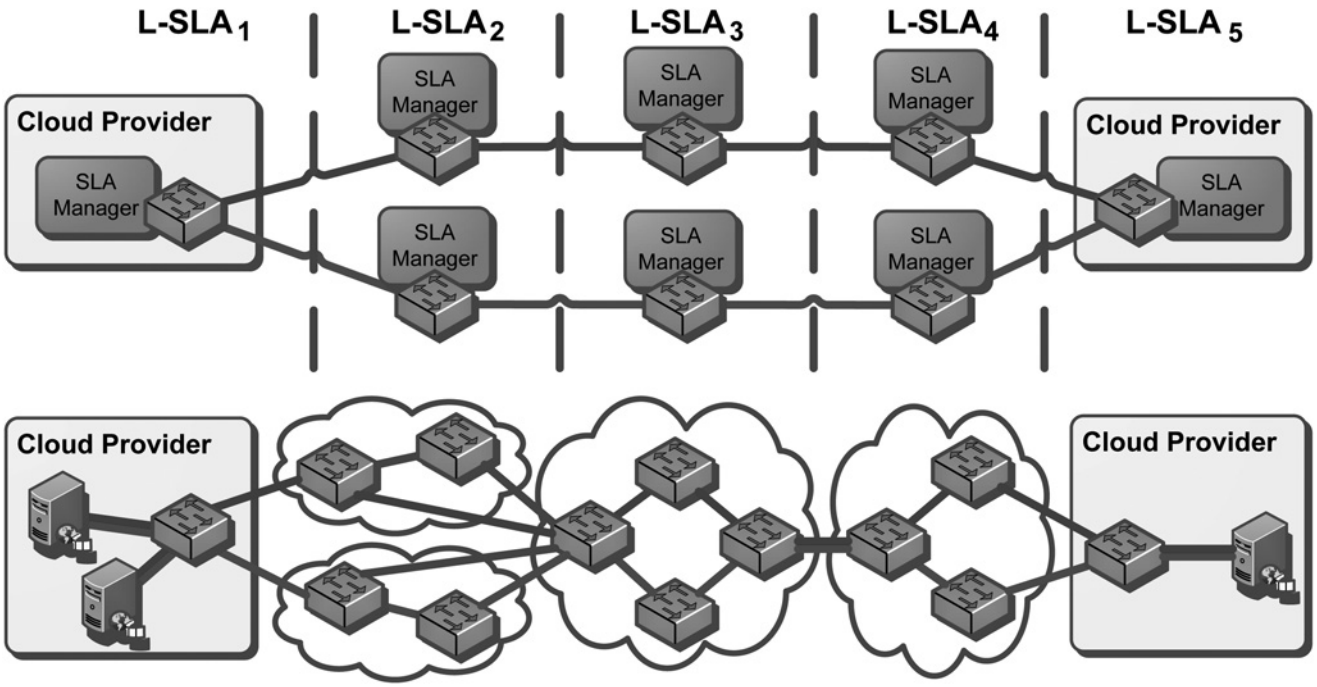

Fig. 3 *Example network SLA aggregation chain*

© The Institution of Engineering and Technology 2015

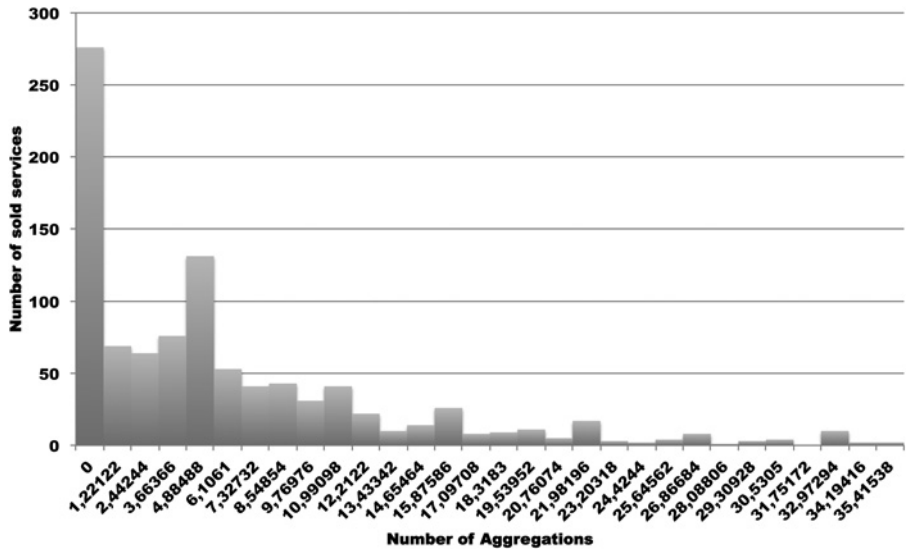

Fig. 4 *Expected probability distribution of federated SLA aggregations*

The qualifying condition is an operation that is used to evaluate an SLA and to express a guarantee

$$\text{QCS} := \{<,\ \leq,\ =,\ \geq,\ >\} \tag{3}$$

To aggregate local-SLA offer to an extended-SLA offer that is passed to the customer, the following aggregation operations OPS are defined

$$\text{OPS} := \{\text{min},\ \text{max},\ \text{sum},\ \text{avg}\} \tag{4}$$

The number of aggregations that need to be performed to aggregate all local-SLAs offer to a single extended-SLA offer is equal to the number of hops of each route. Depending on the kind of service, the aggregation operation has to be applied statically, which allows to aggregate all agreement offer recursivity by requesting all federated SLA managers. Thus, the aggregation function for each aggregation step is defined as follows

$$\begin{aligned}\text{sla}_{\text{extended}} = \big\{&(\text{op}(\text{slo}_1, \text{slo}_2), \text{op}(\text{slt}_1, \text{slt}_2))| \\ &(\text{slo}_1, \text{slo}_2), (\text{slt}_1, \text{slt}_2) \in \text{sla}_1 \times \text{sla}_2,\ \text{op} \in \text{OPS}\big\}\end{aligned} \tag{5}$$

Each agreement offer has an 'Expiration Time' element specified in the 'Context' element of each local-SLA. This expiration time is an absolute date at which this agreement offer is no longer valid and the resources are no longer reserved. Thus, the customer gets a specific time frame in which a decision have to be done.

5 Evaluation

To evaluate the ProgNET approach and especially the SLA aggregation, an expectation about the distribution of SLA instantiations is presented in this evaluation. For the prevision of this approach, the Monte-Carlo simulation was selected as ideal candidate. In general, the Monte-Carlo simulation is a stochastic method describing the performances of many random experiments, whose results build the foundation for decision-making. It uses thereby the probability theory to numerically solve analytically or only consuming solvable problems. The goal is to identify the importance of ProgNET for ISPs in relation to cloud provider.

As already mentioned in the previous section, the number of network hops is not equivalent to the number of overlay SLA manager hops. However, the calculation of SLA manager hops depends on the network hops that are provided by ISPs. As notable research work, the authors in [21] collected hop-count data with a small programme that is based on the traceroute utility. This programme makes use of the time-to-live (TTL) that is set in the packet header. For the Monte-Carlo simulation, the maximum hop-count was set to 255, because the TTL in the packet header is an eight binary-digit field.

In another research work [22], the authors measured also the hop-count in the Internet but calculated additionally the mean and variance for the total sample. Here, the result is an average hop-count of 16, 51 with the variance of 14, 96. Based on these results the normal distribution can be applied with these values.

While both measurements present the number of network hops, the ProgNET approach has just to respect the number of overlay hops. Therefore, the number of network hops resulting from the normal distribution are divided by a random number between one and five, because the average number of hops inside the infrastructure of a single provider is in most cases three [22].

However, the number of hops depends on the position and the target position; therefore an appropriate probability distribution has to respect not only the geographical position but also the intention of the customer. This intension is mapped in this evaluation via a lower bound mapping. In particular, all negative values of the Gaussian distribution are set to zero and represent thus the intention to use the ProgNET capabilities just for VMs inside a single data centre.

The zero for number of aggregations means that no aggregation is performed and just a local-SLA is created for network services of a single provider. Fig. 4 illustrates the results of 1000 runs. Here, it is very well cognisable that most of the customers will probably use ProgNET for establishing QoS characteristics in networks of a single cloud environment. Another peak is located at 4.8 aggregations that display the average overlay hop-count of SLA managers.

6 Conclusion

The presented architecture describes an initial approach for a layered network cloud federation management stack. This particular software stack enables the integration of network federations on heterogeneous cloud middlewares.

The concept addresses the federation of SDN-based cloud networks by an orchestration of SDN and traditional network substrates. Moreover, the highest layer presents an SLA interface for the network resources allocation based on QoS parameters, which are finally enforced by the bottom layer. The bottom layer is built by a network substrate dependent manager with a unified API to the SLA layer. This network management stack delivers

IET Netw., 2015, Vol. 4, Iss. 5, pp. 264–269
© The Institution of Engineering and Technology 2015

the opportunity to integrate the network-based interconnections between VMs or physical hosts into the cloud middleware. Hence this approach creates an equivalent behaviour like the 'get what you pay' concept already used for, for instance, CPU or memory allocation just for network resources.

Altogether the described architecture presents a reasonable software stack approach for the integration of network-based cloud federation over heterogeneous network substrates and cloud middleware software. In the next step we will validate and demonstrate ProgNETs novel mechanisms and tools by means of experiments with the ProgNET integrated prototype implementation.

7 References

1 Korner, M., Stanik, A., Kliem, A.: 'An approach for QoS constraint networks in cloud environments'. 2013 Fourth Int. Conf. on the Network of the Future (NOF), October 2013, pp. 1–3

2 Sherwood, R., Gibb, G., Yap, K.-K., *et al.*: 'Can the production network be the testbed?'. USENIX Symp. on Operating Systems Design and Implementation (OSDI), 2010

3 Sherwood, R., Gibb, G., Yap, K.-K., *et al.*: 'Flowvisor: a network virtualization layer'. Technical report Openflow-tr-2009-1, Stanford University, July 2009

4 Salvadori, E., Corin, R., Broglio, A., Gerola, M.: 'Generalizing virtual network topologies in OpenFlow-based networks'. 2011 IEEE Global Telecommunications Conf. (GLOBECOM 2011), December 2011, pp. 1–6

5 Corin, R., Gerola, M., Riggio, R., De Pellegrini, F., Salvadori, E.: 'Vertigo: network virtualization and beyond'. 2012 European Workshop on Software Defined Networking (EWSDN), October 2012, pp. 24–29

6 Koerner, M., Almus, H.: 'HLA – a hierarchical layer application for OpenFlow management abstraction'. Proc. Fourth Int. Conf. on Network of the Future (NoF'13), Pohang, Korea, October 2013, pp. 1–4

7 Xiao, X., Ni, L.: 'Internet QoS: a big picture', *IEEE Netw.*, 1999, **13**, (2), pp. 8–18

8 Comuzzi, M., Kotsokalis, C., Rathfelder, C., Theilmann, W., Winkler, U., Zacco, G.: 'A framework for multi-level SLA management'. Service-Oriented Computing, ICSOC/ServiceWave 2009 Workshops, 2010 (*LNCS*, **6275**), pp. 187–196. Available at: http://dx.doi.org/10.1007/978-3-642-16132-2_18

9 Happe, J., Theilmann, W., Edmonds, A., Kearney, K.T.: 'A reference architecture for multi-level SLA management', in Wieder, P., Butler, J.M., Theilmann, W., Yahyapour, R. (Eds.): 'Service level agreements for cloud computing' (Springer, New York, 2011), pp. 13–26. Available at: http://dx.doi.org/10.1007/978-1-4614-1614-2_2

10 Rasheed, H., Rumpl, A., Wäldrich, O., Ziegler, W.: 'A standards-based approach for negotiating service QoS with cloud infrastructure providers'. eChallenges e-2012 Conf. Proc., 2012

11 Armstrong, D., Djemame, K., Nair, S., Tordsson, J., Ziegler, W.: 'Towards a contextualization solution for cloud platform services'. 2011 IEEE Third Int. Conf. on Cloud Computing Technology and Science (CloudCom), November 2011, pp. 328–331

12 Stanik, A., Kao, O., Martins, R., Cruz, A., Tektonidis, D.: 'Mo-bizz: Fostering mobile business through enhanced cloud solutions'. 2014 14th IEEE/ACM Int. Symp. on Cluster, Cloud and Grid Computing (CCGrid), May 2014, pp. 915–922

13 Pongpaibool, P., Kim, H.S.: 'Providing end-to-end service level agreements across multiple ISP networks', *Comput. Netw.*, 2004, **46**, (1), pp. 3–18. Available at: http://www.sciencedirect.com/science/article/pii/S1389128604000672

14 Lakshminarayanan, K., Stoica, I., Shenker, S., Rexford, J.: 'Routing as a service' (Electrical Engineering and Computer Science Division, University of California at Berkeley, 2004). Available at: http://www.eecs.berkeley.edu/Pubs/TechRpts/2006/EECS-2006-19.pdf

15 'Cisco', 'Cisco Nexus 1000v series switches for VMware VSphere', Data Sheet June 2014, http://www.cisco.com/c/en/us/products/collateral/switches/nexus-1000v-switch-vmware-vsphere/data_sheet_c78-722606.pdf

16 'Cisco', 'Cisco Nexus 6001 Switch', Data Sheet June 2014, http://www.cisco.com/c/en/us/products/collateral/switches/nexus-6001-switch/data_sheet_c78-726128.pdf

17 Khan, A., Kiess, W., Perez-Caparros, D., Triay, J.: 'Quality-of-service (QoS) for virtual networks in OpenFlow MPLS transport networks'. 2013 IEEE Second Int. Conf. on Cloud Networking (CloudNet), November 2013, pp. 10–17

18 Carlini, E., Coppola, M., Dazzi, P., Ricci, L., Righetti, G.: 'Cloud federations in contrail'. Euro-Par 2011: Parallel Processing Workshops, 2012 (*LNCS*, **7155**), pp. 159–168. Available at: http://dx.doi.org/10.1007/978-3-642-29737-3_19

19 Harsh, P., Jegou, Y., Cascella, R., Morin, C.: 'Contrail virtual execution platform challenges in being part of a cloud federation'. in 'Towards a service-based Internet' (*LNCS*, **6994**) (Springer, Berlin/Heidelberg, 2011), pp. 50–61. Available at: http://dx.doi.org/10.1007/978-3-642-24755-2_5

20 Open Network Foundation: 'Openflow switch specification/version 1.4.0', October 2013, https://www.opennetworking.org/

21 Fei, A., Pei, G., Liu, R., Zhang, L.: 'Measurements on delay and hop-count of the internet'. IEEE GLOBECOM'98-Internet Mini-Conf., Citeseer, 1998

22 Begtaševiü, F., Van Mieghem, P.: 'Measurements of the hopcount in internet'. Proc. of Passive and Active Measurement, PAM, 2001, pp. 23–24

© The Institution of Engineering and Technology 2015