Optical Switching and Networking ∎ (∎∎∎∎) ∎∎∎–∎∎∎

Contents lists available at ScienceDirect

Optical Switching and Networking

journal homepage: www.elsevier.com/locate/osn

SDN orchestration architectures and their integration with Cloud Computing applications

Arturo Mayoral *, Ricard Vilalta, Raul Muñoz, Ramon Casellas, Ricardo Martínez

Centre Tecnològic de Telecomunicacions de Catalunya (CTTC), Parc Mediterrani de la Tecnologia, Av. Carl Friedrich Gauss 7, 08860 Castelldefels (Barcelona), Spain

ARTICLE INFO

Article history:
Received 16 July 2015
Accepted 28 September 2015

Keywords:
SDN
ABNO
Orchestration
AS-PCE
Cloud Computing
VM migration

ABSTRACT

Emerging cloud-based applications, running in geographically distributed data centers (DCs), generate new dynamic traffic patterns which claim for a more efficient management of the traffic flows. Geographically distributed DCs interconnection requires automatic and more dynamic provisioning and deletion of end-to-end (E2E) connectivity services, through heterogeneous network domains. Each network domain may use a different data transport technology but also a different control/management system. The fast development of Software Defined Networking (SDN) and the interworking with current control plane technologies such as Generalized Multi-protocol Label Switching (GMPLS), demand orchestration over the heterogeneous control instances to provide seamless E2E connectivity services to external applications (i.e. Cloud Computing applications).

In this work, we present different orchestration architectures based on the SDN principles which use the Path Computation Element (PCE) as a fundamental component. In particular, a single SDN controller orchestration approach is compared with an orchestration architecture based on the Application Based Network Operations (ABNO) defined within the International Engineering Task Force (IETF), in order to find the potential benefits and drawbacks of both architectures. Finally, the SDN IT and Network Orchestration (SINO) platform which integrates the management of Cloud Computing infrastructure with the network orchestration, it is used to validate both architectures by evaluating their performance providing two inter-DC connectivity services: E2E connectivity and Virtual Machine (VM) migration.

© 2016 Elsevier B.V. All rights reserved.

1. Introduction

Most emerging internet applications rely upon cloud-based services geographically distributed among different data centers (DCs). DCs are generating an increasing amount of dynamic, variable horizontal traffic [1] (i.e. databases synchronization or virtual instances migration), and vertical traffic between customers and DCs. The augment on the number of dynamic application-driven requests of network resources' reservation is moving network operators to find new network architectures which can provide automatic and efficient management of the setup and release of connectivity services. These new architectures must provide end-to-end (E2E) connectivity across different network domains. Intra-DC networks require a very dynamic, packet-based traffic control, while long-haul optical transport networks, which transport the inter-DC traffic, have carrier-grade, multi-domain control requirements.

* Corresponding author.
E-mail address: arturo.mayoral@cttc.es (A. Mayoral).

http://dx.doi.org/10.1016/j.osn.2015.09.007
1573-4277/© 2016 Elsevier B.V. All rights reserved.

Please cite this article as: A. Mayoral, et al., SDN orchestration architectures and their integration with Cloud Computing applications, Optical Switching and Networking (2016), http://dx.doi.org/10.1016/j.osn.2015.09.007

Network management and control systems are suffering an unprecedented evolution since Software Defined Networking (SDN) has emerged as the future networking paradigm. SDN proposes separating the control logic from the switching infrastructure by removing the 'intelligence' from the forwarding elements and concentrating it into a logically centralized SDN controller. OpenFlow (OF) [2] is a standard protocol developed within the Open Networking Foundation (ONF), which allows to externally define the forwarding behavior of the network infrastructure by characterizing the traffic as a combination of flow rules based on the packet headers. It has become the preferred SDN interface between control and data planes. SDN allows cutting costs from network infrastructure due to dedicated hardware that can be replaced by software-based switches installed on cheaper Commercial Off the Self (COTS) servers. This is one of the main reasons why SDN is attracting so much interest from a wide spectrum of the networking industry (especially on DC operators segment).

On the other hand, Generalized Multi-Protocol Label Switching (GMPLS) in combination with the Path Computation Element (PCE) is a mature technology with more than ten years of standardization progress, which offers a carrier-grade control solution for automatic circuit provisioning in Wavelength Switched Optical Networks (WSON). The Active Stateful PCE (AS-PCE) [3] has been demonstrated as a robust and effective management solution for the dynamic establishment and release of optical circuits or Label Switched Paths (LSPs) in GMPLS-controlled optical networks [4].

Operators which have already deployed GMPLS-based control solutions need to assure the return of investment of their current deployments, thus any network upgrade needs to account on existing control technologies. In this context, it arises the need of coordinating or orchestrating multiple, heterogeneous control planes. Inter-working between different control planes requires a higher, master entity (referred here as Network Orchestrator – NO) which automatically coordinates the processes to establish and release E2E connections through different network domains controlled by different control instances. A graphical description of this network scenario can be viewed in Fig. 1.

Recently, the ABNO architecture [5] has been designed within the IETF, based on standard protocols and components to efficiently provide a network orchestration solution for multi-layer and multi-domain networks. In this paper, we are presenting a full-defined ABNO implementation, with a modular, plugin-based, architecture to orchestrate multiple southbound controllers. Its northbound Application Programmable Interface (API) has been designed following the Representational State Transfer (REST)-ful principles to allow external IT applications [6] (i.e. Cloud Computing management systems) to directly request E2E connectivity services into the network. In addition, a previously presented orchestration approach [7] based on a single SDN controller is compared with the ABNO orchestration architecture.

The paper is structured as follows: in Section 2 an overall description of the orchestration process and different orchestration approaches is presented, also Cloud Computing and the need of integrating network resources into a jointly orchestration are examined and introduced. In Section 3, two different SDN orchestration architectures, ABNO architecture and Single SDN Controller Architecture (SC-Arch), are thoughtfully described and compared. To conclude the section, the SDN IT and Network Orchestration (SINO) application and its integration within the two previously described architectures is presented. And finally, Section 4 presents the experimental validation of both network orchestration architectures in the SDN/NFV Cloud Computing platform and Transport Network of the

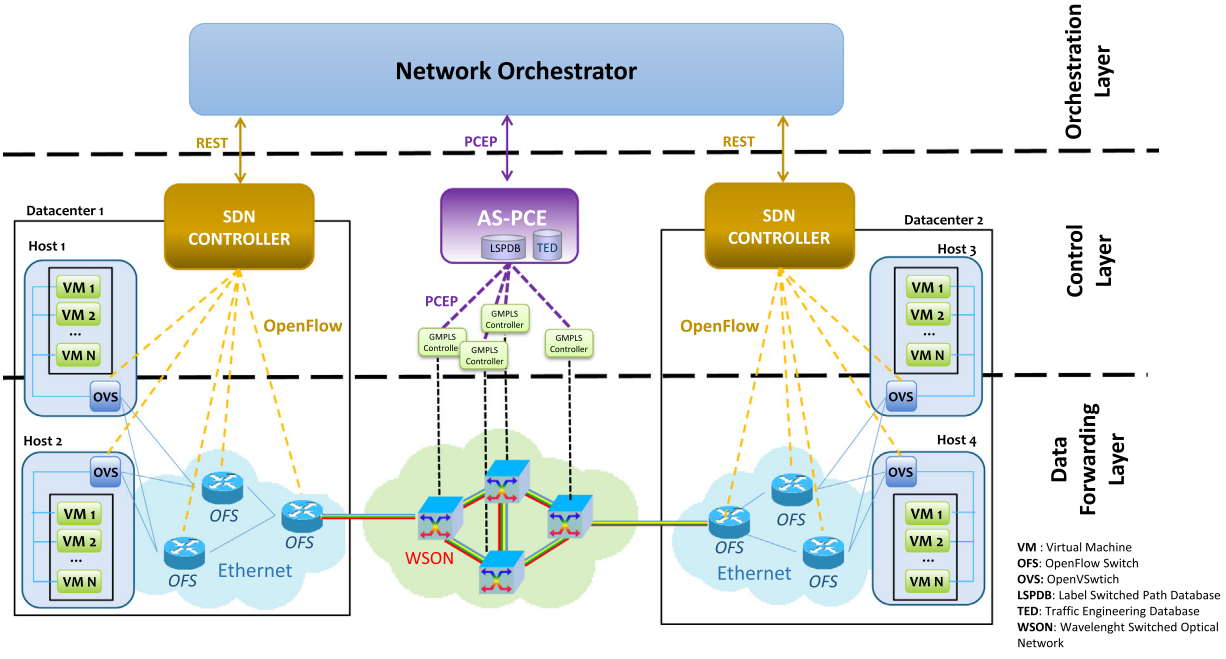

Fig. 1. Network orchestration architecture.

Please cite this article as: A. Mayoral, et al., SDN orchestration architectures and their integration with Cloud Computing applications, Optical Switching and Networking (2016), http://dx.doi.org/10.1016/j.osn.2015.09.007

ADRENALINE Testbed, by measuring and comparing the setup delay of two typical inter-DC communication uses cases: E2E channel provisioning and Virtual Machine (VM) migration.

2. State of the art

This section reviews the current state of the art on SDN orchestration and Cloud Computing. It is organized in two subsections: first one goes into the description of the orchestration process and the global perspective of the current orchestration approach; and second one, it makes a brief introduction to Cloud Computing and the need of integrating network resources into a jointly orchestration to provide E2E network transport services for the interconnection of the Cloud Computing infrastructure.

2.1. SDN orchestration

Firstly, we define here network orchestration as the coordination and automation of the establishment and release of multiple independent network connections (usually performed by different control instances) for the provisioning of E2E connectivity services through heterogeneous network domains (which might be composed of different network technologies).

The SDN orchestration approach is based on a hierarchical architecture where a software-based logically centralized entity provides E2E communication through different transport networks (Ethernet/DWDM) and/or control technologies (SDN/OF, GMPLS/PCE). This hierarchical approach has been proposed before in [8] where multiple OpenFlow-enabled PCEs are orchestrated by a parent-PCE to provision E2E connections. In [9], the ABNO architecture is used in a international Testbed to orchestrate E2E connectivity services across several optical network transport technologies (Optical Packet Switching – OPS, Elastic Optical Networks – EON [10] and WSON) controlled by different SDN controller distributions.

Differently, in this work we explore the orchestration of a network scenario composed of SDN/OF network domains interconnected by a GMPLS/PCE controlled transport network, with a logically centralized Network Orchestrator (NO). Network domains are interconnected by border links shared between two nodes which belong to different domains. This heterogeneity, among the control technologies considered, introduces the need of implementing multiple southbound interfaces in the NO and also introduce new elements of discussion on the design of the orchestration architecture (i.e. level of topology abstraction and multi-layer path computation). Now, we propose to outline the set of requirements that should be fulfilled by the orchestration layer in the previously presented scenario:

1. Translation of the external application connectivity service requests to the configuration of the different control plane instances. Definition of a standard and extensible northbound API to support customer service requests and offering network control abstraction to customer applications.
2. Discovery and inventory of the control instances and the network topology. Full physical network topology information is not strictly required in the orchestration layer, but at least, the inter-domain connectivity and an abstracted view of the network domains are required.
3. E2E path calculation across the different network domains. Domain selection or full path computation depending on the level of topology abstraction.
4. Provisioning and restoration of the E2E connectivity. Programmability of the different network controllers through specific provisioning interfaces depending on the underlying control technology.
5. Event handling, notification support of changes in the network (failures, topological changes, etc.).

In the following paragraphs we present in detail different protocol/interface alternatives for the design and implementation of a SDN orchestration architecture which fulfill the previously outlined requirements.

Topology discovery: The NO may compose its network topology by the cooperation of the underlying network controllers which can advertise its intra-domain topologies using different protocol or interfaces. Most of the SDN controllers implement custom RESTful APIs to offer network topology to external applications, but also other possible interfaces are attracting a lot of interest, this is the case of the NETCONF protocol [11] and its RESTful based version RESTCONF [12], both based on the YANG modeling language [13].

The topology discovery in the orchestration layer can be done in a reactive or proactive manner. In the proactive approach, the NO recovers the network topology to the control plane instance every time it needs to refresh its working copy to perform a new path computation. RESTful interfaces are connection-less interfaces and they do not inherently support asynchronous notifications. This feature may constrain some orchestration implementations which employ RESTful as a topology discovery interface, to operate only in a proactive manner.

Another alternative to expose the network topology by a control instance is the northbound distribution of the Link State and TE information using the Border Gateway Protocol (BGP-LS). The BGP-LS UPDATE message has been recently extended [14] to allow multi-protocol network layer reachability information (NLRI) and can be used to advertise the link-state topology gathered by the control instance. This approach was experimentally validated in [15] for the advertisement of intra- and inter-domain TE Links by child-PCEs (c-PCEs) to a higher parent-PCE (p-PCE) responsible for the inter-domain path calculation. BGP-LS is a session-oriented protocol which can provide to the Orchestration Layer asynchronous notifications about changes into the network topology. A BGP-LS speaker instance is required in both the NC component and into the NO. This approach would represent a possible implementation of the reactive approach of topology discovery.

Regarding the control layer, the preferred solution for most SDN controllers is the combination of the Link Layer Discovery Protocol (LLDP) and OF. This technique consists of sending periodic OFPT_PACKET_OUT messages encapsulating an LLDP

Please cite this article as: A. Mayoral, et al., SDN orchestration architectures and their integration with Cloud Computing applications, Optical Switching and Networking (2016), http://dx.doi.org/10.1016/j.osn.2015.09.007

packet to the network elements (NE) within the network domain. These packets are forwarded to a specific port according to the forwarding rule included in the OF message. When another OFS receives a LLDP packet and there is no specific flow action for that packet, the OFS executes the OFPT_PACKET_IN action and encapsulating the LLDP message and re-send it to the controller. This way the SDN controller can determine the network topology.

Similarly, in the GMPLS control plane an Internal Gateway Protocol (IGP), i.e. Open Shortest Path First (OSPF) protocol, can be used to share the topological information between GMPLS nodes. The AS-PCE can sniff as well those IGP's packets and building the TED based on the gathered information.

Path computation: The E2E path computation can be performed in a different manner depending on the level of topology abstraction and the number of transport layers. Depending on the level of abstraction, the NO may just have access to an abstracted view of the network consisting in the domain connectivity with inter-domain links an abstract representation of the network domains [16]. In this case, the intra-domain path computation is delegated to the lower, per-domain controllers and the NO only performs the domain selection of the controllers involved in the E2E path calculation.

Another alternative is the discovery of the complete physical network topology by the NO. In this case the NO is responsible for calculating the full path across the network, which in the proposed scenario comprises different layers. In the multi-layer scenario, a separate path computation instance (i.e. a PCE) for each layer topology can be employed, or a single path computation instance, with network visibility of all the transport layer topologies, can use multi-layer aware algorithms to calculate the routes.

Connectivity provisioning: The E2E connectivity provisioning involves the orchestration of different control plane and forwarding technologies. The NO is responsible for the implementation of the interfaces or protocols exposed by the control plane to forward the orders from the orchestration layer to the control layer. Also, the generalization of different transport technology connections into a flexible and common data structure is a key requirement to be able to offer abstracted information to upper layers through the NBI.

Most SDN controllers offer a custom provisioning REST API in which input parameters are aligned with those used on the OFPT_FLOW_MOD messages to insert flows into the OF-enabled switching devices. A flow is defined by one or more matching rules which range from Layer 2 to Layer 4 packet headers, and an action (forward to a specific port, drop packet, forwarding to the controller, etc.) inserted into the forwarding device's OF table.

The AS-PCE can instantiate or remove LSPs into the network using the PCEP initiate request message (PCInitiate) [17]. This message includes the endpoints and the computed explicit route object (ERO), defining the route and resources to be traversed and allocated by the LSP. After the connection is successfully established, a PCEP Report Message (PCRpt) is generated to notify to the AS-PCE the successful LSP establishment and its management (e.g., deletion and modifying attributes). The PCEP protocol can be used as a NBI of the AS-PCE to expose a programming interface to the orchestration entity.

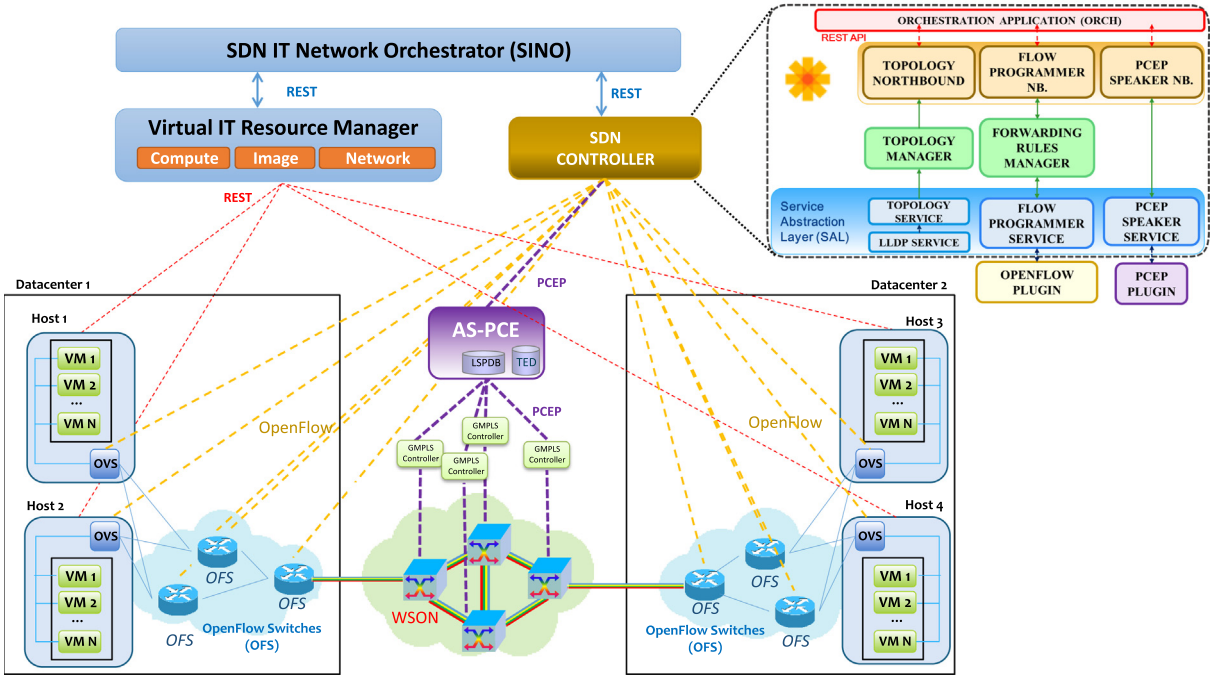

Fig. 2. Architecture based on a single SDN controller.

Please cite this article as: A. Mayoral, et al., SDN orchestration architectures and their integration with Cloud Computing applications, Optical Switching and Networking (2016), http://dx.doi.org/10.1016/j.osn.2015.09.007

2.2. Cloud computing and datacenter interconnection

Cloud computing services are becoming an essential part of any enterprise IT infrastructure. Cloud has introduced a new application paradigm where storage and server infrastructures are hosted and shared. These new paradigm have allowed cost reduction and innovation in services and applications. These new innovations are based on sliced servers, introducing a pay-as-you-go model. In this context, the concept of Platform as a Service (PaaS) is introduced as a new service model for networking the future Internet. Thus, it is one of the faster emerging business for Internet Server Providers (ISP).

A data center (DC) refers to any large, dedicated cluster of computers that is owned and operated by a single organization. DCs interconnection is one of the biggest problems that service providers have to face. DCs have been spread geographically to reduce services' latency to the end user, and that has led into an exponential growth on the inter-DC traffic [18]. Moreover, the DC interconnection cannot rely upon static, coarse granular and expensive connection pipes, but need to be adjusted to the traffic demand for both Cloud and Network providers which can take full advantage of the existing network infrastructure [19].

The integration of the network control and management systems with cloud-based applications relies upon the promotion of open, standard interfaces between network control/orchestration systems and upper applications, which allow gathering the relevant information from the network and directly programming the network behavior. Standard interfaces allow IT and Network applications being developed independently hiding the internal implementation details and offering abstracted services, such as the virtual machine creation/deletion or the E2E connectivity provisioning.

3. Proposed network orchestration architectures

In this section two different SDN orchestration approaches are described and compared in terms of their features and characteristics of each approach; the implementation requirements; and control overhead they introduce in the network orchestration process.

3.1. Orchestration based on a single SDN controller (SC-Arch)

Firstly, we start describing an orchestration approach based on a single SDN controller architecture (SC-Arch) (Fig. 2). This architecture handles the network orchestration by a single, logically centralized SDN controller. It directly controls the OF network domains and delegates the control of the optical transport network to the AS-PCE.

On top of the SDN controller architecture a NBI interface is responsible for offering the services to external application which must be general and not constrained to control specifics. A SDN controller can implement different protocol interfaces as southbound plugins. The SDN controller handles the control technology abstraction internally by means of a Service Abstraction Layer (SAL) which exposes device services to applications hiding specific networking protocol plugins characteristics. The SAL determines how to fulfill the requested service irrespective of the underlying protocol used between the controller and the network devices.

Among the different open-source SDN controllers available, we have chosen the OpenDayLight (ODL) project [20] for implementing the SDN controller. OpenDaylight SDN controller already includes the PCEP protocol as a southbound plugin, however a PCEP-Speaker service has been created to manage the dynamic establishment of LSPs into the GMPLS/PCE control plane. The main building blocks of

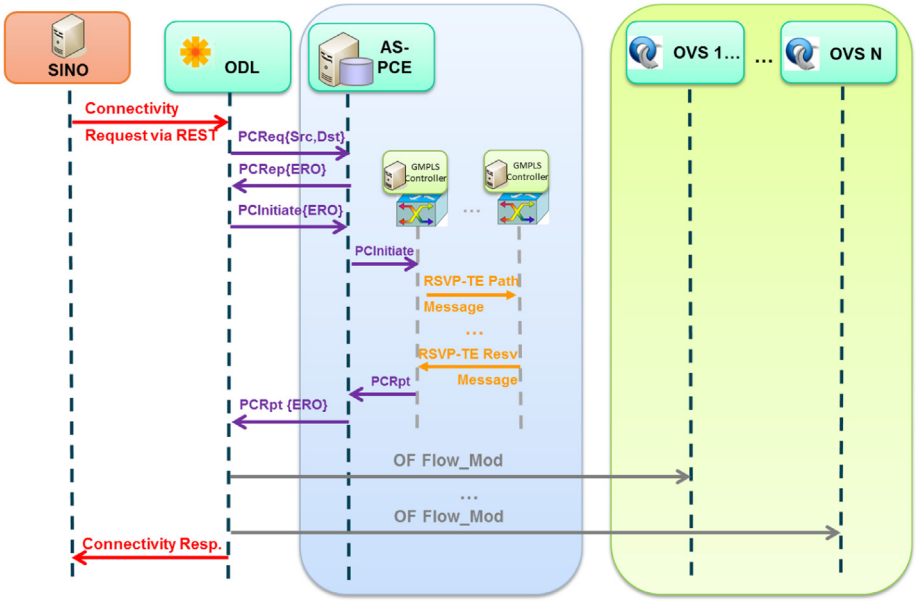

Fig. 3. E2E provisioning workflow with single SDN controller orchestration architecture.

Please cite this article as: A. Mayoral, et al., SDN orchestration architectures and their integration with Cloud Computing applications, Optical Switching and Networking (2016), http://dx.doi.org/10.1016/j.osn.2015.09.007

the internal architecture of the proposed modified ODL controller are shown in the top-right part of Fig. 2.

In our implementation, the SDN controller does not recover the whole L0 optical network topology (BGP-LS could be used to recover this information), the optical domain topology is abstracted as a single-node, whose ports represent the border nodes of the optical transport network. The inter-domain connectivity is dynamically loaded from an external configuration file when the SDN controller starts. Consequently, the path computation, inside the optical domain, is delegated to the AS-PCE.

On top of the SC-Arch, an internal orchestration application (ORCH) has been created in order to handle the required mechanisms to solve the E2E provisioning in the multi-layer network. The E2E provisioning workflow diagram can be seen in Fig. 3.

The ORCH recovers the network topology from the Topology Manager Service (TMS). It is also responsible for the path calculation between the E2E connectivity request's endpoints. If the ORCH fails to obtain a feasible route in the network, it explores the inter-domain connectivity information to detect if the E2E service request failed due to the lack of connectivity through the optical domain. If so, a new virtual L0 connection is established the ORCH sends a Path Computation Request (PCRequest) to the AS-PCE to obtain a feasible route between the border OFS. After a successful response from the AS-PCE, a LSP establishment request with the pre-calculated route encoded into an Explicit Route Object (ERO) is sent to the AS-PCE through the PCEP-Speaker service. After the AS-PCE notifies the PCEP-Speaker Service the effective LSP creation, the new links are discovered by the Topology Manager between the disconnected OFSs through the LLDP mechanism described in Section 2.1.

Finally, the SDN controllers provisioning the E2E connection by calculating the route into the OF network topology managed by the TMS between the E2E request's endpoints and it configures the OFS forwarding tables through OFPT_FLOW_MOD messages containing the match rules (source and destination hosts MAC addresses) and the action (input port/output port).

3.2. ABNO architecture for multi-domain, multi-controller orchestration

In this section, we proposed a network orchestration approach based on the ABNO architecture orchestrating a dedicated control instance for each transport network: a SDN controller and an AS-PCE (Fig. 4). Firstly, the different modules that confirm the ABNO architecture are presented, describing how this approach fulfills the set of requirements enumerated in Section 2.1.

On top of ABNO architecture, the Orchestration Controller (OC) handles the internal workflow between different ABNO modules and it processes the incoming requests from the NBI. The NBI offers, to SINO an other external applications, the CRUD operations for the E2E connectivity service. It has been implemented by a RESTful API.

The Topology Manager (TM) is the component responsible for gathering the network topology from each control domain and building the TED. The TED includes all the relevant information about network links and nodes, and it is used by the Path Computation Element (PCE) for calculating routes across the network. In our implementation, the TM

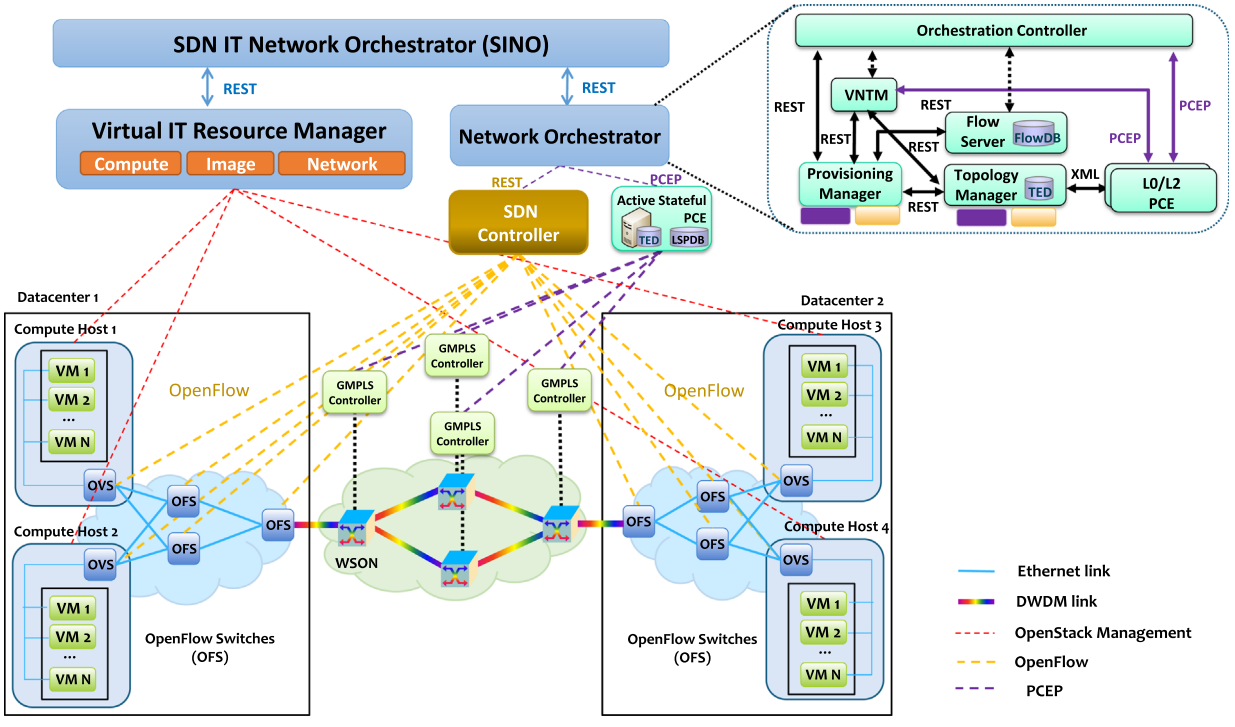

Fig. 4. ABNO architecture for multi-domain, multi-controller orchestration.

Please cite this article as: A. Mayoral, et al., SDN orchestration architectures and their integration with Cloud Computing applications, Optical Switching and Networking (2016), http://dx.doi.org/10.1016/j.osn.2015.09.007

recovers the physical network topology of each network domain and the inter-domain connectivity from Summary. From this information, the TM builds a complete multi-domain topology and a separated topology, built by filtering the whole topology based on the TE information of the links, for each transport layer technology. The TM implements dedicated plugins for (1) the custom SDN controller RESTful interface; (2) a proprietary interface of the AS-PCE based on a raw socket TCP and XML/JSON encoding. The topology discovery is performed in a proactive manner.

The Virtual Network Topology Manager (VNTM) is responsible for the multi-layer management. In the proposed scenario, the VNTM is in charge of the set-up of Layer-0 (L0)/DWDM optical connections to satisfy upper layer's connectivity demands. For instance, if a Layer-2 (L2)/Ethernet connectivity service request cannot be served because there is not an available route through the L2 network topology, the VNTM manages the set-up of a L0 connectivity request between the network domains to which belong the L2 endpoints. To do this, the VNTM requests the inter-domain topology to the TM, to find the L0 border node pair between whom request the connection. After the successful establishment of the L0 connection, the VNTM notifies the TM the creation of a new virtual link and the related virtual ports on the border nodes into the L2 network topology. The mapping of the L0 connections and L2 virtual links is responsibility of the VNTM too.

The Provisioning Manager is the module which translates the connectivity requests, processed by the ABNO controller and the VNTM, into the corresponding provisioning request messages of the underlying network controllers. The Provisioning Manager implements a provisioning plugin for each different network controller connected to it. In the proposed architecture it implements the custom SDN controller's provisioning REST API and the PCEP with Stateful and PCE-initiated LSP Setup extensions, for the communication with the AS-PCE. All the established connections (both L0 and L2) are stored in the Flow server by the Provisioning Manager. The Connection is the data structure exchanged between the different ABNO components in our implementation, and it consists of the following parameters:

- *Endpoints*: Source and destination nodes, described as {Router_ID, Interface_ID}.
- *Path*: List of hops traversed by the connection, each one described as {Router_ID, Interface_ID}.
- *Transport_layer*: (L0, L2).
- *Forwarding_rules*: Matching_Rules and Action similar to the OpenFlow equivalents [2].
- *Connection_type*: (Unidirectional, Bidirectional).

The provisioning workflow of an E2E service, involving the inter-DC network, can be seen in Fig. 5. The internal ABNO workflow of the E2E service involves the creation of a virtual link by the VNTM through the PCEP plugin. A PCI-initiate message is sent to the AS-PCE to setting up the creation of a new LSP to interconnect the OF network domains. This request includes a pre-calculated path through the optical network and the border node's ports connected to the border links which interconnect the OF domains with the optical transport network. These Endpoints are represented as Unnumbered Interfaces composed of the router-ID and the Interface-ID encoded in 64 bits. After the creation of the virtual links the OC requests the path computation into the L2/ Ethernet layer and sends the Provisioning Requests to the SDN controller through the Provisioning Manager interface. These requests contain the MAC addresses of the source and destination E2E service Endpoints as matching rules for the OFSs configuration.

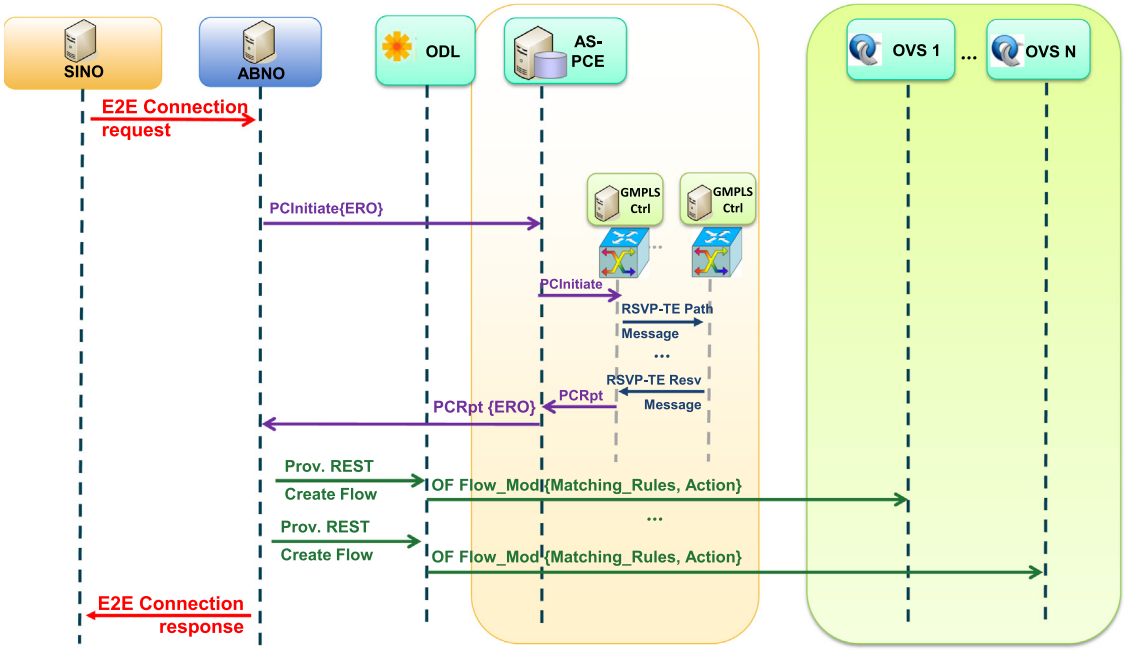

Fig. 5. E2E provisioning workflow with ABNO orchestration architecture.

Please cite this article as: A. Mayoral, et al., SDN orchestration architectures and their integration with Cloud Computing applications, Optical Switching and Networking (2016), http://dx.doi.org/10.1016/j.osn.2015.09.007

Table 1
Network orchestration approaches summary.

Architecture	**ABNO**	**Single SDN controller**
Features		
Topology discovery	Full topology discovery, proactive mode	Abstract optical domain topology, reactive mode
Provisioning connections	Proactive	Proactive
Path computation	Dedicated PCEs for each layer, full path computation	Domain selection, intra-domain path computation delegated to control instances
Interfaces		
Provisioning	REST, PCEP	OpenFlow, PCEP
Topology discovery	REST, BGP-LS	LLDP
Network control overhead	Internal PCEP communication, REST communication between ABNO components	

3.3. Orchestration approaches comparison

After describing the two orchestration approaches we present in this work, we summarize and evaluate the differences between them in Table 1.

In the first place, the two proposed orchestration solutions follow a similar hierarchical approach. The main architectural difference between them is the complete separation between control and orchestration layers in the ABNO approach. The ABNO architecture allows delegating some of the control tasks to underlying control instances, and this feature lets us to think that it will present less scalability problems in large network deployments, in comparison with the SC-Arch architecture. Regarding the SC-Arch approach, the control and orchestration are made by the same SDN controller combining the direct control of network equipment with the integration of other control instances without disclosing the internal domain network details.

Regarding the implementation of these solutions, the E2E service provisioning in a proactive manner in both cases. Topology discovery is made proactively within the ABNO-based architecture by implementing custom REST interfaces between the TM and the per-domain controllers and reactively in the SC-Arch approach. While the ABNO architecture recovers the complete, non-abstracted, physical topology, the SC-Arch approach recovers only an abstracted view of the optical domain and delegates the intra-domain path computation to the AS-PCE. Finally, let us mention that the ABNO-based orchestration approach introduces some control overhead by implementing each ABNO architecture component as independent entities and using PCEP and REST interfaces for the communication between them.

3.4. Cloud computing integration with SDN orchestration architectures

In this section we are going to describe the integration of the Cloud Computing platform with the already presented SDN orchestration architectures. The purpose is to provide integrated IT and Network resources orchestration.

The Cloud Controller is named Virtual IT Resource Manager (VIRM) platform, which is responsible for the management of the creation/ migration/deletion of VM instances (computing service), disk images storage (image service), and the management of the VM network interfaces (networking service). The VIRM handles multiple compute servers geographically distributed in the DC locations.

The computing service manages the VM into the compute hosts (Hosts 1–4 in Figs. 2 and 4). A compute service agent is running in each host and controls the computing hypervisor (e.g., KVM) used for the creation/deletion of the VMs. The image service handles the disk images which are used as templates for VM file systems. The connectivity between VMs and virtual switches within the hosts is managed by the networking service. The networking service is responsible for the handling of all network elements, such as switches and firewalls. We propose to modify the networking service, in order to limit the networking services to the creation of the virtual interfaces, the attachment of the virtual interfaces to the virtual switches and finally the offer of a DHCP service for the VMs to get the assigned IP address. The NO will be responsible for the control and management of the connectivity services between the different network domains. The SINO is introduced in order to coordinate the VIRM and the NO.

The SINO controls the VIRM through a RESTful API, used to both trigger the VIRM actions and get the necessary information about the running VM instances.

The SINO has been previously presented in [21], for the orchestration of IT and Network resources. The SINO exposes a variety of services, among which are VM Create, Read, Update, Delete (CRUD) operations; E2E CRUD connectivity; and VM migration. It exposes all these services through a NBI implemented through a RESTful API. In Section 4 we use the SINO for the selected use cases to evaluate the performance of the two NO architectures described in the following sections

4. Experimental validation

The experimental validation of the previously described architectures has been carried out by the implementation of two typical uses cases demanded by Cloud Computing applications: E2E connectivity provisioning and seamless virtual machine migration.

The SDN/NFV Cloud Computing platform and transport network of the ADRENALINE Testbed, which consists of the SINO, the VIRM and the NO, is the reference scenario used for obtaining all the experimental results that will be shown

Please cite this article as: A. Mayoral, et al., SDN orchestration architectures and their integration with Cloud Computing applications, Optical Switching and Networking (2016), http://dx.doi.org/10.1016/j.osn.2015.09.007

below. The overall architecture was described in Section 3.4 and shown in Figs. 2 and 4.

The Cloud Computing platform has been implemented using the Openstack Havana release, specifically, the VIRM corresponds to the Cloud Controller node in the OpenStack nomenclature. The VIRM controls four Compute Hosts located in two different Data Center (DC) locations, which are responsible for storing the VMs created in the cloud. All cloud components have been deployed into physical servers with 2x Intel Xeon E5-2420 and 32 GB RAM each.

The proposed network scenario consists of two intra-DC network domains based on Ethernet transport technology and an inter-DC optical transport network. Each intra-DC network consists of four OpenFlow Switches (OFS) deployed using standard Custom Off The Shelf (COTS) hardware and running OpenVSwitch (OVS), which are all controlled by a single SDN/OpenFlow controller (OpenDaylight in both architectures). Each DC border switch has been implemented using COTS hardware, a 10 Gb/s XFP tunable transponder and OVS technology. The inter-DC network is a GMPLS-controlled Wavelength Switched Optical Network (WSON) which consists of 2 ROADMs and 2 OXCs providing re-configurable (in space and in frequency) E2E lightpaths, deploying a total of 610 km of G.652 and G.655 optical fiber, with six DWDM wavelengths per optical link.

4.1. E2E connectivity service

For this use case, four different VMs have been deployed on each of the compute hosts distributed in the two DC locations. The experiment will measure the setup delay introduced, by both orchestration architectures, by providing an E2E channel between two of these VMs. Each service request received in the NO demands a L2 connectivity service between the OFS' ports connected to two randomly chosen VMs. Each connection created in the network is removed before the next service request arrives, thus the blocking probability is theoretically zero and all requests have uniform network conditions.

In Fig. 6, the setup delay distribution of 150 requests experiment are shown for both architectures. The results show two different groups of setup delays in both architectures: a set of L2 service requests between co-located VMs (same DC) provisioned in less than 1.5 s; and another group involving the setup of an optical connection into the GMPLS domain (inter-DC communication), the setup delay in this case varies in the range of 2.5–5 s.

If we analyze the Intra-DC requests results (Table 2), we can see better performance achieved by the SC-Arch (lower mean value). This result is motivated by the fact that ABNO introduces more orchestration overhead mainly by the usage of the PCE in the path computation.

The inter-DC group of requests shows more differences. Firstly, mean value of setup delay under ABNO architecture is almost one second less, however, it is remarkably the big dispersion obtained in the SC-Arch results (Std.Dev=0.589 and CV=0.16). The dispersion of the setup delay in the SC-Arch is caused by the responsiveness of the SDN controller to the effective connectivity through the inter-DC connection. On the other side, when ABNO NO receives the response from the GMPLS control plane of the effective establishment of the LSP starts the Virtual Link creation and the provisioning of the L2 connection. Therefore it shows a more deterministic and predictable behavior and globally better performance than the SC-Arch.

4.2. Seamless virtual machine migration

The second use case proposed to evaluate the overall network orchestration architectures is the seamless migration of a VM. In the proposed scenario, the VM1 is connected to the VM2 (both initially running into the DC-1), the experiment has been designed to migrate VM1 to DC-2. The migration comprises the use of the inter-DC network and it will be evaluated by measuring the disruption time of the connectivity between VM1 and VM2. The process involves six steps, detailed in Fig. 7a:

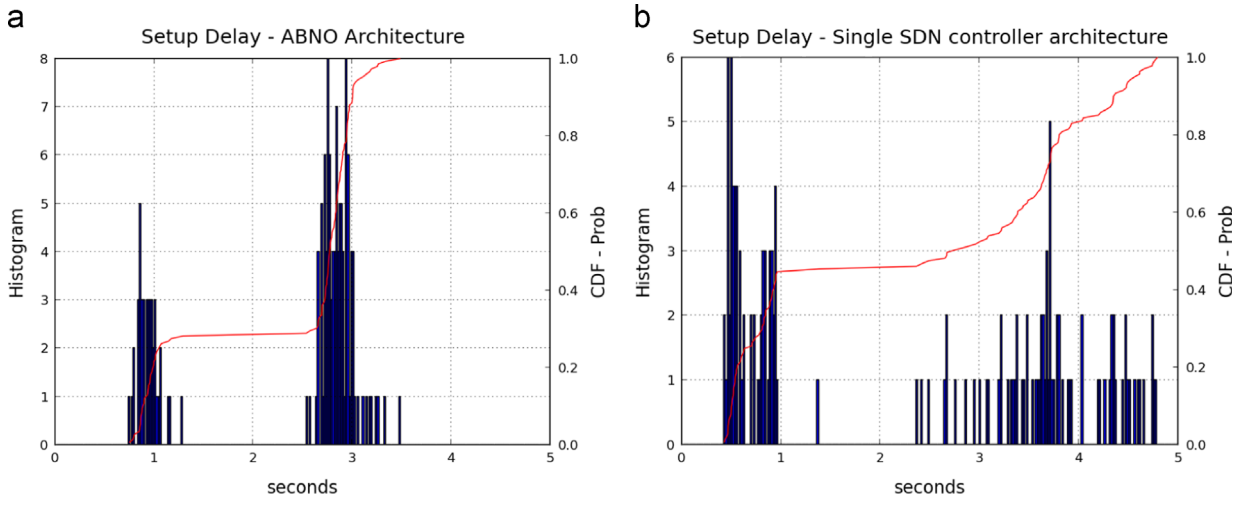

Fig. 6. E2E workflow: (a) setup delay distribution under the ABNO-Arch and (b) setup delay distribution under the SC-Arch.

Please cite this article as: A. Mayoral, et al., SDN orchestration architectures and their integration with Cloud Computing applications, Optical Switching and Networking (2016), http://dx.doi.org/10.1016/j.osn.2015.09.007

1. *Virtual machine migration request trigger*: SINO receives a VM1 migration request through HTTP request.
2. *Deletion of VM connections*: All the flows established in the network in which the target VM is involved must be removed before start of the migration. The SINO requests to NO the deletion of all flows involving VM1. The NO deletes all of them by sending the orders to the SDN controller which at once will send the OFPT_-FLOW_REMOVED messages to the OFSs.
3. *Inter-DC connectivity*: The VIRM, the source and destination hosts must be interconnected during all the migration process in order to exchange the VM images data using the VIRM image service. The SINO adds each of the three hosts (VIRM, Host1 and Host3) as external hosts into the NO, and request an end-to-end path between each pair. The inter-DC connectivity between hosts involves the establishment of a Label Switched Path (LSP) into the WSON by sending a Path Computation Initiate Message to the AS-PCE.
4. *IT migration*: First, VM1 is paused and snapshot in the source host and it is saved into the VIRM. Once the VM1 snapshot is ready, VM1 is removed and a new instance based on VM1 snapshot is deployed into destination server.
5. *Restore VM connections*: The SINO restores the connections between the migrated VM and all the rest of VMs to which it was connected. The SINO requests a new flow between VM1 and VM2 to NO. The NO's PCE computes a path, but since the previous optical LSP between DCs is reused, it is only necessary to create the flows of the OFSs.
6. *Acknowledgement of virtual machine migration*: The SINO notifies the client of VM1 successful migration.

Table 2
Results statistics.

Architecture	ABNO		Single SDN controller	
Request group	Intra-DC	Inter-DC	Intra-DC	Inter-DC
Mean	0.945 s	2.867 s	0.681 s	3.730 s
Std.Dev[a]	0.091 s	0.142 s	0.178 s	0.589 s
CV[b]	0.111	0.053	0.291	0.161

[a] Standard deviation.
[b] Coefficient of variation = (Std.Dev)/(Mean).

Fig. 7b shows the network traces captured at the SINO's server. It includes the HTTP communication through the REST interfaces between SINO, VIRM and NO. These modules are running in the same server and that is why the IP address is the same in the messages exchanged between those entities. For the network orchestration, it can be seen how the NO request the AS-PCE the creation of a LSP through the optical network for the inter-DC communication and finally, the OFPT_FLOW_MOD messages are sent first, for the deletion of flows and for creating them again at the end of the instance migration.

This use case has been evaluated only with the ABNO architecture. Looking at the results (Fig. 7b), the major part of the migration's disruption time is introduced by the IT orchestration (VM snapshotting and creation) and not by the network orchestration process. As it is seen in the previous subsection, the differences of the E2E connectivity provisioning setup delay, between both architectures, are in the order of hundred of milliseconds. In comparison with the whole disruption time measured in the migration experiment (Fig. 8), those differences can be negligible and we believe do not justify a comparison in these terms for this use case. Besides, except for the network orchestration part, which was detailed for both architectures in section 3, the migration workflow would be the same for both architectures.

In Fig. 8a is shown the incoming traffic to DC-2 sampling each 5 s the packets received in the border OFS of DC-2. A significant increase of the number of packets occurs after the trigger of creation of a VM1 into Host3 due to the download of the VM1 image from VIRM. Finally, a traffic capture of the dialog between VM2 and VM1 is shown in Fig. 8b which provides the total disruption time occurred due to the seamless migration process. We can observe that after a downtime of 158 s the connectivity

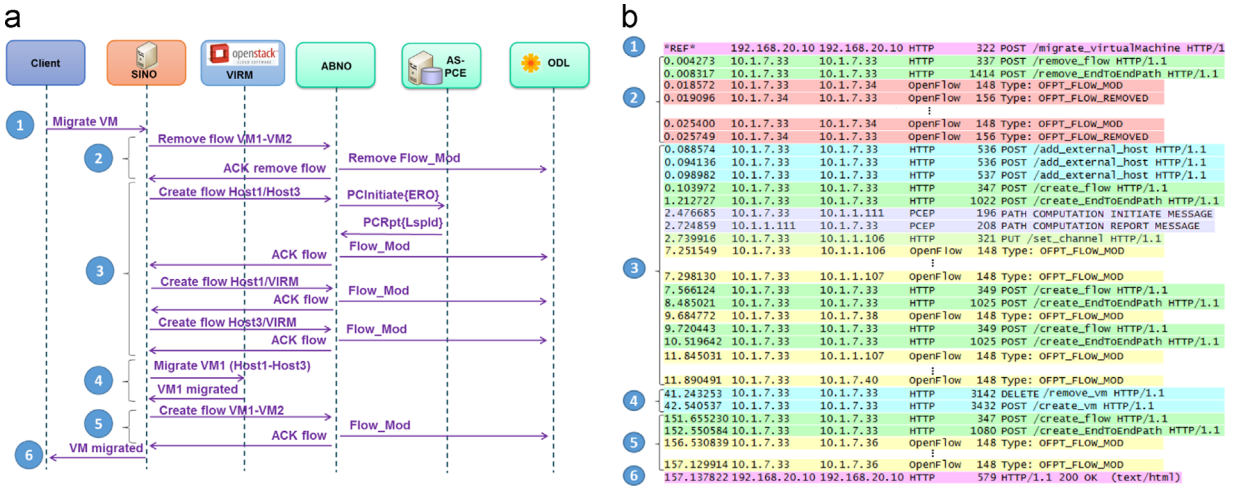

Fig. 7. (a) VM migration flow diagram and (b) wireshark capture of SINO commands.

Please cite this article as: A. Mayoral, et al., SDN orchestration architectures and their integration with Cloud Computing applications, Optical Switching and Networking (2016), http://dx.doi.org/10.1016/j.osn.2015.09.007

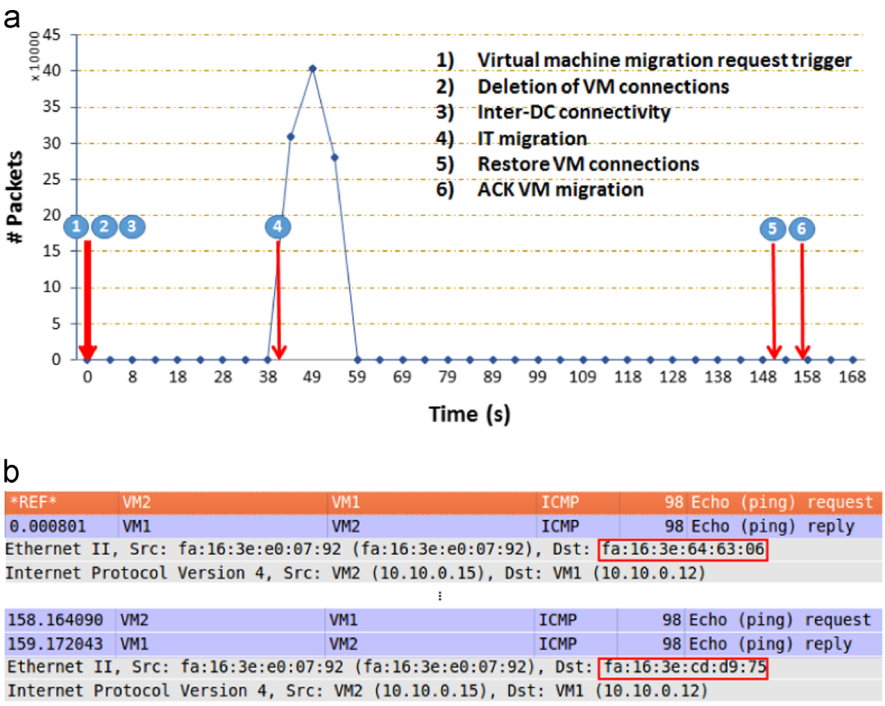

Fig. 8. (a) VM migration traffic (packets/s) received in DC2 over time, and (b) ICMP traffic capture between VM1 and VM2 during VM1 seamless migration.

between VM1 and VM2 is recovered. It can be observed that the MAC address of VM1 has changed due to the block migration.

5. Conclusions and future work

This work has present an in-depth discussion about the network orchestration problem in multi-domain networks comprising different transport technologies and control planes. The integration between network orchestration and potential client applications (i.e. Cloud Computing) has also been considered by presenting IT and SDN orchestration architecture (SINO-VIRM-NO) and validated through all the experiments carried out in the study.

Two possible network orchestration approaches, ABNO and SC-Arch, have been discussed and compared through the definition of their architectures and implementations. They have been experimentally validated by measuring the setup delay introduced in the orchestration process of the provisioning of E2E connectivity services in the reference scenario.

ABNO architecture has presented more deterministic and predictable results in the setup delay distribution measured in the E2E connectivity use case than the SC-Arch approach. In terms of absolute performance the results have not shown significant differences. ABNO architecture also provides a more scalable solution for the orchestration problem because it completely separates orchestration and control layers. This separation really matters in network scenarios where each domain belongs to different administrative entities which may not want to disclose their internal information with other management systems.

To continue the work presented in this paper, the integration of resilience mechanisms in the orchestration architecture will be a necessary step to provide a more close to reality solution for the problem assessed here, the reactive approach on topology discovering is a key-element for this integration. Another important consideration is the heterogeneity of the interfaces to communicate the Orchestration Layer and the Control instances, a standardized solution for a common interface is needed when different SDN controllers (provided by different providers) will be considered for the control of multiple SDN domains.

Acknowledgment

This work was funded by the European project EU FP7 STRAUSS (FP7-ICT-2013-EU-Japan 608528), the Spanish MINECO project FARO (TEC2012-38119) and PACE Coordinated Support Action (Grant agreement no. 619712).

References

[1] T. Benson, A. Akella, D. Maltz, Network traffic characteristics of data centers in the wild, in: Proceedings of the 10th Annual Conference on Internet Measurement, ACM, Melbourne, Australia, 2010, pp. 267–280.
[2] Open Networking Foundation, Open Networking Forum: OpenFlow Switch Specification, ONF v.1.3.0, June 2012.
[3] E. Crabbe, I. Minei, J. Medved, R. Varga, PCEP Extensions for Stateful PCE, draft-ietf-pce-stateful-pce-09, IETF, 2014.
[4] R. Casellas, R. Martinez, R. Muñoz, R. Vilalta, L. Liu, T. Tsuritani, I. Morita, Control and management of flexi-grid optical networks