EVPN/SDN Assisted Live VM Migration between Geo-Distributed Data Centers

Kyoomars Alizadeh Noghani
Department of Computer Science
Karlstad University
Karlstad, Sweden
kyoomars.noghani-alizadeh@kau.se

Andreas Kassler
Department of Computer Science
Karlstad University
Karlstad, Sweden
andreas.kassler@kau.se

Prem Sankar Gopannan
Ericsson Research
Ericsson
California, USA
prem.sankar.g@ericsson.com

***Abstract*—Live Virtual Machine (VM) migration has significantly improved the flexibility of modern Data Centers (DC). However, seamless live migration of a VM between geo-distributed DCs faces several challenges due to difficulties in preserving the network configuration after the migration paired with a large network convergence time. Although SDN-based approaches can speed up network convergence time, these techniques have two limitations. First, they typically react to the new topology by installing new flow rules once the migration is finished. Second, because the WAN is typically not under SDN control, they result in sub-optimal routing thus severely degrading the network performance once the VM is attached at the new location.**

In this paper, we identify networking challenges for VM migration across geo-distributed DCs. Based on those observations, we design a novel long-haul VM migration scheme that overcomes those limitations. First, instead of reactively restoring connectivity after the migration, our SDN-based approach proactively restores flows across the WAN towards the new location with the help of EVPN and VXLAN overlay technologies. Second, the SDN controller accelerates the network convergence by announcing the migration to other controllers using MP-BGP control plane messages. Finally, the SDN controller resolves the sub-optimal routing problem that arises as a result of migration implementing a distributed anycast gateway. We implement our approach as extensions to the OpenDaylight controller. Our evaluation shows that our approach outperforms existing approaches in reducing the downtime by 400 ms and increasing the application performance up to 12 times.

***Index Terms*—Data Center, Ethernet VPN, Software Defined Networks, Distributed Gateway, VM Migration.**

I. Introduction

Live Virtual Machine (VM) migration between geo-distributed Data Centers (DCs) is a key enabler for cloud federation, supports dynamic and transparent load balancing among DCs, improves the resiliency, and makes DCs more fault-tolerant. Emerging new concepts such as mobile edge cloud [1] benefit from efficient long-haul VM migration because the VM that provides a service may need to migrate to a new edge cloud in order to run the service closer to the end user [2]. While many schemes exist that enable seamless live VM migration, higher network latency and lower available capacity are two key differences between a VM migration inside a DC and a long-haul VM migration across geo-distributed DCs.

978-1-5386-4633-5/18/$31.00 ©2018 IEEE

Live VM migration needs to be seamless by maintaining ongoing connections and providing an overall low service downtime. Consequently, the downtime due to migrating the CPU and memory states needs to be short utilizing an optimal migration strategy [3]–[5]. In addition, the network convergence time, i.e. the time needed to re-establish the connectivity for ongoing connections is also crucial in order to minimize the network interruption and provide a resilient and seamless live VM migration scheme. Finally, after the VM has migrated to a new location, network configurations inside the VM should remain valid which is a challenge as different DCs may have different network settings. After the connectivity is re-established, sub-optimal routing may further reduce the performance of the applications inside the VM, increase the network congestion level, and waste bandwidth.

Several approaches tackle the problem of long-haul migration. A number of studies [4]–[6] optimize the state transfer procedure to ensure that service downtime is not disruptive. Solutions such as Mobile IP [7], L2VPN (e.g., VPLS [8]), and overlay networks to help the VM preserve its ongoing connection after the migration are leveraged in [4], [9], [10]. However, they either result in suboptimal routing problems or overlook the requirements of multi-tenant DCs. Accelerating the network convergence for VM migration can be done by proactively re-routing the flows to new locations of VMs [11], [12]. However, [11] is not applicable to long-haul migration as they lead to the triangular-routing problem and flow bicasting [12] wastes bandwidth in the core network (MPLS/WAN). Furthermore, both studies assume that the whole network is managed by a single controller which is difficult to achieve in geo-distributed DCs.

This paper presents a novel approach for long-haul live VM migration between geo-distributed DCs that accelerates the network convergence time and optimizes the post-migration traffic routing by coordinating among the multiple SDN controllers. First, the network convergence time is reduced by proactively re-routing flows during the VM migration phase using SDN. When entering the stop-and-copy phase, SDN controller extensions trigger a modified Ethernet Virtual Private Network (EVPN) MAC advertisement message [13] in the control plane using MP-BGP protocol. When receiving such a message, remote SDN controllers are aware of the

new reachability information before the network interface is detached from the old location. As a consequence, they modify corresponding flow rules to pro-actively send traffic to the new VM location. Second, because our SDN controller also acts as a gateway for all VXLAN networks, such *SDN-based distributed anycast gateway* addresses the sub-optimal inter-subnet routing problem after migration. By advertising gateway information, SDN controllers in the remote sites handle inter-subnet VM traffic. As a result, our solution does not result in sub-optimal traffic routing and thus avoids the hair-pinning problem. We have implemented our approach as extensions to the SDN controller OpenDaylight (ODL) [14]. By performing experiments in emulated WANs, we find that our approach is effective in reducing the downtime compared to alternative schemes for inter-DC migration. We also introduce intra- and inter sub-network traffic scenarios and study how our approach improves the available bandwidth of applications running inside the migrating VMs. In summary, we make the following key contributions:

- We propose a new live VM migration scheme for geo-distributed DCs that is based on SDN, EVPN, VXLAN and distributed anycast gateways. It proactively re-routes traffic using SDN and uses control plane signaling to trigger SDN-based re-routing.
- We implement our approach as ODL extensions and evaluate it in a geo-distributed DC setup to show that it reduces the downtime by 400 ms and improves the bandwidth by approximately 12 times.

The remainder of this paper is organized as follows. Section 2 elaborates network challenges of long-haul VM migration. Section 3 presents the relevant background for our work. Section 4 describes the proposed architecture. In Section 5, the results of experiments are described and presented. Finally, this paper is concluded in Section 6.

II. Design Challenges for live VM migration across the WAN

Live VM migration is a key algorithm for modern DCs. In geo-distributed DCs, multiple DC sites are interconnected over the WAN, typically using MPLS networks. In contrast to intra-DC networks, where links have typically less than one millisecond latency and 40 or 100 Gbps link capacity is common, WAN connections have significantly higher latency and lower capacity. Supporting seamless live VM migration between different DCs which are connected over the WAN poses several important challenges:

Maintain Ongoing Connections: The ability to maintain ongoing connections is a prerequisite for seamless VM migration. However, network settings (e.g., IP address space) would typically be different when the VM moves between remote sites, thus making it difficult to seamlessly transfer active network connections. Various solutions have been proposed to aid a migrating VM to maintain its ongoing connections after migration. Authors in [9], [15] have leveraged mobile IP solutions [7], [16] to address the problem. Bradford et al. [5] proposed a combination of dynamic DNS and IP tunneling to maintain the network state during long-haul live migration. Alternatively, layer 2 DC interconnect technologies (e.g., VPLS), overlay networks, and SDN-based methods have been deployed in different studies [4], [10], [17], [18] to address the same problem.

The ideal solution ought to be scalable, easy to deploy, and does not raise sub-optimal routing problems. However, proposed solutions fall short to meet the requirements of seamless migration in different aspects. For example, Mobile IP-based solutions [9] cause triangular routing, require the VM or its corresponding hypervisor to have a modified protocol stack, or need all involved networks to support a specific protocol. Extending an overlay network from one DC to another DC is neither scalable nor efficient as it extends the broadcast domain. Moreover, remote sites may need to leverage different overlay technology. Legacy layer 2 DC interconnect solutions are limited in terms of redundancy, scalability, flexibility, and forwarding policies. Finally, proposed SDN-based solution [17] leverages mapping between the old and new network address to maintain the ongoing connections which is not a scalable solution. Furthermore, SDN-based solutions [17], [19] assume that the controller has a holistic view over all DCs and is capable of managing all DC networks. However, long-haul VM migration involves multiple networks where each may have its own management infrastructure.

Large Convergence Time: In long-haul live VM migration the network convergence time is significantly larger in comparison to live VM migration within a DC. Until the network is not converged, ongoing connections between the migrating VM and its peers continue sending traffic to the former location of the VM and consequently introduce further service interruption. Although the VM can generate a gratuitous or reverse ARP in order to advertise its new location to switches and therefore accelerate the route convergence time in a legacy network, this approach delays the network convergence to after the VM is up and running at the new location. The ideal solution is to conduct flow migration in parallel with the system state migration as proposed in [11], [12], [19]. The key idea is to pro-actively prepare the network for VM migration using SDN-based architecture and OpenFlow-based forwarding rule rewriting. However, unlike these approaches which consider flow restoration using a single network controller, we orchestrate the process over multiple domains controlled by multiple controllers.

Sub-Optimal Routing: When a VM migrates, the network should find an optimal path for its ingress and egress traffic. Using a sub-optimal path significantly degrades the application performance. Triangular routing and hair-pinning effect are the most well-known problems that arise as the result of VM migration. Triangular routing results into higher latency as all traffic destined to a device has to pass through its home agent which degrades the application performance and places unnecessary workload over the WAN. Conducting fast network convergence in conjunction with avoiding triangular routing significantly improves the performance of applications residing in the migrating VM.

When the VM wants to establish a connection with nodes outside of its IP subnet, it requires a gateway, which is responsible for routing the traffic between different subnets. The *default gateway problem* occurs because a VM does not flush its ARP table when relocating from one server to another and continues sending packets with the destination MAC address set to that of the original gateway which is now located in the previous DC. This problem results into the hair-pinning effect which is shown in Fig. 1. The end host with VLAN-1000 has an ongoing connection with the other end host in VLAN-3000. Meanwhile, the VM migrates from DC-1 to DC-2. Since the default gateway is located in DC-1, the traffic leaves DC-2, makes a u-turn in the DC-1, and finally reaches the end-host in DC-2. The hair-pinning problem imposes additional workload on network nodes, increases the congestion level in the WAN, and wastes bandwidth especially across WAN links.

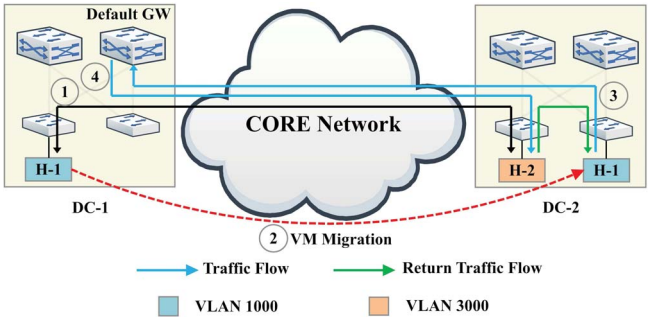


Fig. 1: Hair-pinning effect

Note that the frequency of observing the hair-pinning problem depends on various parameters such as application communication pattern, the frequency of long-haul VM migration, and etc.

III. Background

This section describes the background for the leveraged technologies that help the VM to maintain its ongoing connections after the migration. Moreover, this section elaborates on the EVPN procedure for announcing the VM migration and distributing the gateway information.

*a) **VXLAN and EVPN**:* Extending the L2 broadcast domain ensures that a VM can be reached with the same IP address after the migration while it removes the need to leverage complex L3 overlay methods such as mobile IP.

VXLAN has emerged as a technology which addresses traditional VLAN limits and enables multi-tenant networks at scale. VXLAN provides L2 extension over a shared L3 underlay infrastructure network by using MAC in IP/UDP tunneling encapsulation. As a result, a VM can move everywhere within a VXLAN-based DC network while retaining its ongoing connections. Note that other overlay technologies such as LISP may also help the migrating VM to maintain its ongoing connections. However, this paper considers VXLAN as it is widely adopted and deployed in today's DC networks.

To support VM mobility between DCs, cloud providers might want to stretch VXLAN tunnels between DCs. However, VXLAN is not designed to be a DC interconnect solution as it extends the broadcast domain from one DC network to another which in turn may violate the scalability, efficiency, and security of the DC network. There are a number of L2 DC interconnect solutions available that help to stretch VXLAN between DCs.

EVPN encompasses next-generation ethernet L2VPN solutions and is designed to handle sophisticated redundancy access scenarios. It also provides per-flow load balancing, enhances the flexibility and decreases the operational complexity of legacy L2VPN solutions. Furthermore, EVPN utilizes MP-BGP in the control plane as a signaling method to advertise addresses which removes the need for traditional flood-and-learn in the data plane. Having VXLAN-based DCs that are interconnected through EVPN technology is a scalable and efficient solution that helps VMs to maintain its ongoing connection for migration between geo-distributed DCs. Fig. 2 illustrates such a design with a pair of provider edge (PE)[1] routers. We refer the reader to [20] for detailed information.

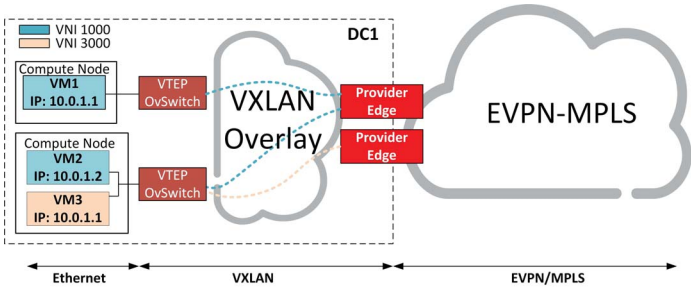


Fig. 2: EVPN-VXLAN and EVPN-DCI interworking

*b) **VM Mobility in EVPN**:* EVPN comprises four types of messages: ethernet auto-discovery, ethernet segment, inclusive multicast, and MAC/IP advertisement route. The EVPN MAC/IP advertisement message is designed to advertise MAC/IP reachability information of a given VM. When an EVPN capable node is informed about a new MAC address (through using local learning, e.g., data-plane), it advertises the information to its peers through the MP-BGP protocol. All remote peers that belong to the same EVPN instance import this route and insert the announced MAC address and its reachability information (e.g., ethernet tag[2]) into their MAC VRF (Virtual Routing and Forwarding) table. This process allows the remote nodes to know where to send the traffic [13].

By adding an additional extended community section to the MAC/IP advertisement message, EVPN capable nodes can update each other about VM movement. Every MAC mobility event for a given MAC address contains a sequence number that increases with each MAC move. This is used by EVPN capable nodes to ensure that the MAC advertisements are processed correctly. An EVPN capable node advertises a MAC address for the first time with no *MAC mobility extended community* attribute. When another EVPN capable

[1]PE provides connectivity between a DC and the MPLS/WAN network.
[2]An ethernet tag identifies a particular broadcast domain, e.g., a VLAN.

node detects a locally attached MAC address for which it had previously received a MAC/IP advertisement route, it advertises the MAC address in a MAC/IP advertisement route tagged with a MAC mobility extended community attribute with a sequence number one greater than the last received sequence number [13]. Fig. 3 shows the EVPN MAC advertisement message structure, its parameters, and extended communities that MP-BGP message may carry in conjunction with MAC advertisement message. The fields with the * symbol are optional.

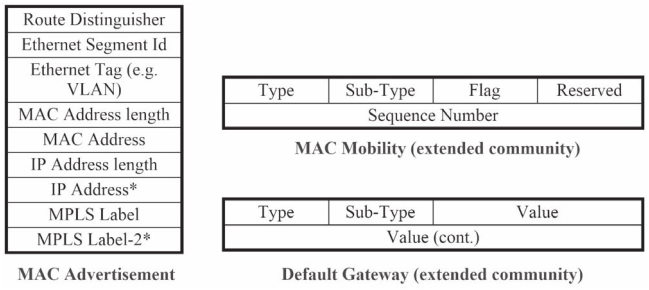


Fig. 3: MAC advertisement message and extended communities

*c) **Distributed Gateway using EVPN**:* To resolve the hair-pinning effect, the end-host default gateway should be present across all locations where the end-host may migrate. The default gateway problem can be addressed in two ways. In the first approach, which is known as distributed anycast gateway, the network administrator allocates a specific IP and MAC address for each locally defined subnet and configures the gateways accordingly. Therefore, the VM has access to the gateway regardless of its physical location. The distributed anycast gateway removes suboptimal routing inefficiencies associated with separate centralized gateways and provides significant added value to VXLAN deployment within the DC. However, deploying anycast gateway requires the network provider to configure all gateways beforehand which is not dynamic and scalable.

EVPN offers a unique and scalable solution which allows gateways to be actively distributed across an arbitrary number of network elements. This is especially relevant in cloud environments where an L2 tenant might exist anywhere in the fabric. Using the combination of MAC/IP advertisement message and *default gateway extended community*, an EVPN capable node distributes its information to its EVPN peers. The remote peers treat the received MAC/IP address as equivalent to their own gateway interface for the purposes of gateway processing. As a result, the gateways are distributed around all DC networks that are part of a given EVPN instance. This approach requires the least manual configuration. However, the main drawback of this approach is a requirement of advertising multiple addresses in the control plane as each EVPN capable node has a different network address.

The ideal solution is to have a unique gateway address (akin to distributed anycast gateway technology) which is distributed in the control plane between all nodes that participate in the same L2 domain automatically (akin to EVPN technology).

IV. Architecture and Implementation

In our proposed solution, the SDN controller is responsible to stimulate the convergence time and resolve the hair-pinning problem. First, the controller reduces the network convergence time by proactively re-routing flows during the VM migration phase. When entering the stop-and-copy phase, our SDN controller extensions trigger a modified EVPN MAC advertisement message [13] in the control plane using MP-BGP protocol. When receiving such a message, remote SDN controllers are aware of the new reachability information and consequently modify corresponding flow rules to proactively send traffic to the new VM location. Second, because our SDN controller acts as a gateway for all VXLAN networks, such SDN-based distributed anycast gateway addresses the sub-optimal inter-subnet routing problem after migration. By advertising gateway information, SDN controllers in the remote sites handle inter-subnet VM traffic.

The controller modules and their interactions with DC network nodes are shown in Fig. 4. Herein, we assume that all DC networks are SDN-enabled, particularly, the DC on which the VM originally is located and the destination DC to which the VM is going to migrate.

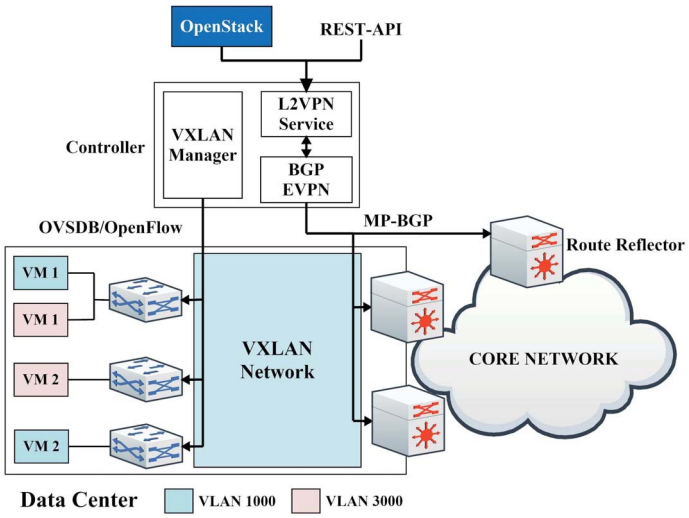


Fig. 4: Proposed architecture

A. Controller Modules

The ODL controller is extended with the following modules.

1) L2VPN-Service: L2VPN-Service collects, stores and updates all parameters related to each EVPN instance and MP-BGP operations, e.g., the remote end hosts MAC/IP addresses belonging to such an EVPN. This module receives the EVPN parameters from OpenStack or network administrator using REST-API. Moreover, the L2VPN-Service interacts with BGP-EVPN module, which receives EVPN control plane messages. When an MP-BGP message concerning EVPN is received, the L2VPN-Service stores the received information fetched by the BGP-EVPN module. Additionally, the L2VPN-Service determines the execution of EVPN control messages such as MAC advertisement messages and provides the BGP-EVPN

module the necessary parameters such as MAC/IP, MPLS label, etc. For further information, we refer the reader to [21].

2) BGP-EVPN: BGP-EVPN module parses and serializes MP-BGP messages related to EVPN. It exchanges EVPN related information with the L2VPN-Service module and communicates EVPN information to external elements such as PEs using MP-BGP extensions. When the BGP-EVPN module receives a new EVPN MP-BGP control message, it parses the information inside the BGP NLRI (Network Layer Reachability Information) and provides that information to the L2VPN-Service and VXLAN-Manager for further actions. Additionally, when the L2VPN-Service module needs to advertise or update information belonging to an EVPN instance such as a new MAC address advertisement, it provides address information (e.g. MAC/IP address) to the BGP-EVPN module, which in turn creates the related MP-BGP message. For further information, we refer the reader to [21].

3) VXLAN-Manager: The VXLAN-Manager has following key responsibilities: 1) managing VXLAN tunnels, 2) managing multi-destination traffic in the overlay network, and 3) providing L2/L3 gateway functionality. Since the VXLAN-Manager module is located at the SDN controller, it can retrieve a holistic view of a DC network and consequently be informed about the location of all end hosts. As a result, the VXLAN-Manager is an ideal entity to manage the VXLAN tunnels between end-points inside a DC network. When one of the end-points are located in a remote DC, the controller steers the traffic to the PE router using a VXLAN tunnel. Then the PE router decapsulates the VXLAN packet and sends the packet to a remote PE router that advertised the remote end host MAC address. Second, the VXLAN-Manager can effectively manage multi-destination traffic in the overlay network by reducing the amount of that traffic and establishing an efficient multicast tree inside a DC. Finally, thanks to the SDN controller the VXLAN-Manager would be an ideal point to provide L2/L3 gateway functionality as it is logically centralized but physically distributed over the whole DCs and all VXLAN Tunnel End Points (VTEPs) can easily reach this module using an out-of-band channel. The L2 gateway functionality is deployed in our proof-of-concept implementation.

Considering the SDN controller modules that are described in Section IV-A, herein we discuss the following aspects:

1) How does the SDN architecture decrease the downtime by accelerating the convergence time?
2) How does the SDN architecture increase the application performance by removing the hair-pinning effect?

B. Improving Network Convergence Time across DCs

As mentioned in Section III, the EVPN-enabled node triggers the MAC mobility message when the migration is actually conducted. However, this approach delays the network convergence to after the VM is up and running at the new location. In our proposed solution, the SDN controller in the source location withdraws the MAC address reachability when the end host enters the stop-and-copy phase. The rationale behind this approach is to stop receiving more packets since the node is not processing any packet when it enters the stop-and-copy phase.

The SDN controller can be noticed about the stop-and-copy phase through OpenStack or hypervisor triggering a specific API call to the controller. Adding such functionality into the hypervisor is more beneficial as it manages the migration procedure and has precise information about the migration states. Furthermore, such functionality does not decrease the hypervisor performance as the migration of VMs that the hypervisor hosts is not an event that happens each and every moment. Once the hypervisor triggers the notification it can start the stop-and-copy phase without waiting for an acknowledgment from the controller. On the other hand, signaling the controller on each VM migration may impose an overhead when a single controller is used. However, DCs widely utilize distributed SDN controllers to address their scalability requirements. In such an environment signaling a local SDN controller does not lead to any performance degradation in the control plane.

EVPN has mass withdrawal feature that is used when there is a link failure on the Ethernet Segment Identifier (ESI) [13] or when the ESI configuration changes. However, the withdrawal simply invalidates the MAC entries for the whole ethernet segment. To withdraw only a single MAC address, a fine granular procedure is required. Herein we propose to leverage the reserved bit in the MAC mobility extended community (Fig. 3) to distinguish between the *actual* and *in progress* migration. MAC mobility message with the reserved bit set to 0 means that the VM is totally migrated. In contrast, reserved bit set to 1 conveys that the migration is in stop-and-copy phase. Upon receiving this message, the remote sites stop sending the flow to the old location. This approach can be further improved if the SDN controller in the source location advertises the new VM location information through the MAC advertisement message. As a result, the remote sites can start updating their routing table and steer the traffic towards new VM location.

Consider the scenario in Fig. 5, where two end-hosts in remote DCs (DC-1 and DC-3) have an ongoing connection with the VM in DC-2 (step 1). When the SDN controller is informed about the migration destination (step 2), it triggers the MAC advertisement message comprising the new VM destination information with MAC mobility extended community with the reserved bit set to 1 (step 3). Remote DCs receive the control plane message and start to steer traffic flows to the new VM location (step 4). This method prevents packets from being routed to the old location, saves bandwidth in the network (including both the WAN and DC networks), and improves the network convergence time.

But how does the controller learn about remote sites information? EVPN is designed to have a simple configuration procedure. The administrator configures the router with essential parameters (e.g., route distinguisher, VLAN, etc.). Afterwards, the router itself spreads its reachability information by sending various types of EVPN control plane messages. If other routers belonging to the same EVPN instance and their routing policy

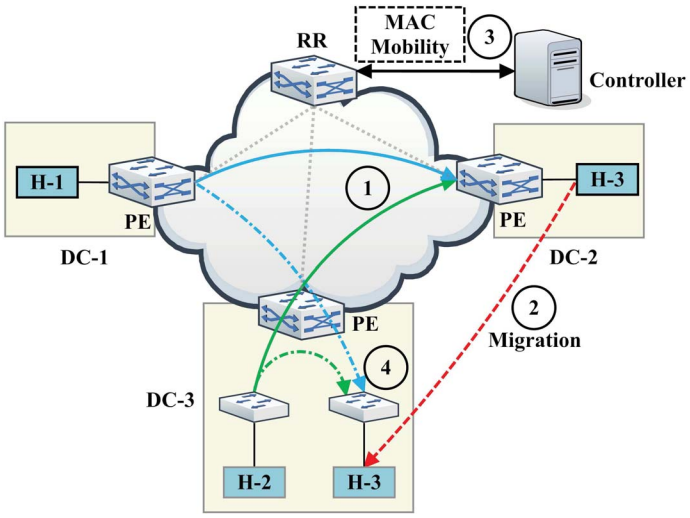


Fig. 5: Withdraw MAC address

allows them to import the messages, they consequently learn the information of other routers. As our developed SDN controller is capable of parsing and serializing the EVPN messages, it can import and interpret the EVPN messages and update its data structure akin to EVPN capable routers. As a result, at the time of migration, the SDN controller at the source location has already captured a list of parameters about the remote sites. The only challenge is to migrate the node to the network that does not have any other nodes in that L2 domain. In such a case, the SDN controller should wait until the EVPN instance is configured on the remote peer and the remote information reaches the controller.

Controller Work Flow: When the VM enters the stop-and-copy phase, the L2VPN-Service retrieves the information about the destination site from its internal data structure and feeds them to BGP-EVPN module. Consequently, BGP-EVPN module serializes the MAC/IP advertisement message and sends it to its peers (e.g., through a route reflector). The BGP-EVPN module of the remote SDN controller receives and deserializes the message. Then it provides the information inside the MP-BGP message for both L2VPN-Service and VXLAN-Manager. VXLAN-Manager either updates the edge router (DC-1 in Fig. 5) or modifies the VXLAN tunnels within the DC network (DC-3 in Fig. 5). L2VPN-Service updates its local tables, accordingly as it is explained in [21].

C. Addressing the Hair-Pinning Problem

In a VXLAN network, the gateway node ought to bridge or route the traffic depending on the packet destination address. When the source and destination nodes belong to the same subnet but are located in different L2 domains (different VLANs), the traffic should be bridged. When the source and destination end-hosts are located in different subnets, then VXLAN routing is required. In both cases, the source end-host sends the packet to its associated gateway. Then the gateway node performs an L2/L3 lookup and encapsulates the packet with the appropriate virtual network identifier. In our proposed architecture, the VXLAN-Manager module implements the legacy gateway node functionalities. When the controller receives a packet from a VTEP, it decides on how to transfer the packet. Accordingly, the controller installs the corresponding rules on the VTEP.

Controller Work Flow: When the administrator defines the gateway configurations for a specific L2 domain (via OpenStack or Rest API), the L2VPN-Service asks the BGP-EVPN module to distribute that information to its peers by triggering the MAC advertisement message carrying the default gateway extended community. Consequently, the controller in the remote sites import the message through BGP-EVPN module, parse the parameters, and feed the extracted information to the VXLAN-Manager and L2VPN-Service modules. When the node migrates, it can continue sending packets to its default gateway address. The SDN controller in the new location receives the packets and checks the destination IP and MAC address. If the packet destination address matches with the gateway address which was received through MAC advertisement message, the controller deploys the gateway functionality by installing proper flow rules on both source and destination VTEP nodes.

V. Evaluation

We have conducted several experiments to answer the following questions:

- How does our proposed framework decrease the downtime and consequently increase the application performance hosted on the migrating VM?
- How does our proposed framework increase the application performance hosted on the migrating VM by addressing the hair-pinning effect?

To address the two aforementioned questions, the following evaluation scenarios are developed:

1) Intra subnet: This scenario illustrates the effectiveness of our proposed solution in decreasing the downtime. In this scenario, the migrating VM and its corresponding peers, which are located in remote sites, belong to the same broadcast domain. Our proposed solution is compared with two alternative methods: 1) mobile IP, and 2) EVPN. In mobile IP method, the traffic destined to the migrating VM initially goes through its home DC, then the controller redirects the traffic to its new location using an end-to-end tunnel. In the EVPN method, the controller at the destination site ignites the convergence by sending a MAC advertisement message with an appropriate value in the MAC mobility extended community immediately after the VM is up and running at the destination.

In contrast, in our proposed approach, the SDN controller at the source site distributes a MAC advertisement message with the information of the destination DC carrying an appropriate value in the MAC mobility extended community just before the node enters the stop-and-copy phase. We emulated this approach by letting the end host send a specific signaling packet to the controller.

2) Inter subnet: This scenario investigates how our proposed solution enhances the application performance by removing the hair-pinning problem. In this scenario the migrat-

ing VM and its peers belong to different broadcast domains and the gateway entity is required to route the traffic between them. Our proposed method is compared with a centralized gateway that leads to the hair-pinning problem.

Experiment Setup: Fig. 6 shows the network topology which is emulated using Mininet[3]. The experiments are conducted using four 3.2GHz Core i7 processor Intel systems with 8 cores and 16 GB of RAM under Linux 4.4.0 kernel. SDN controllers are deployed in three different hosts and one computer hosts the Mininet. We implemented extensions to Beryllium version of the ODL controller. All switches in the topology are OpenFlow capable (vSwitch 2.6[4]). DC networks are emulated with a linear topology including a single TOR and a core switch, acting as the edge router, that interconnects DCs. Links inside DCs are configured for 1 Gbps bandwidth and 1 ms delay. The links in the core network have the bandwidth of 1 Gbps and we vary the latency values between 214 ms, 110 ms, and 10 ms respectively. 214 ms delay is selected according to [6] where the authors measured the latency between a site in Palo Alto, California and a remote site which is located at Bangalore, India. 110 ms is considered as a latency between a site in Europe and a site in North America. Finally, 10 ms delay is considered for two sites which both are located in Europe.

We are aware that emulated setup does not allow to understand all the implications of a real-life situation such as the variable VM downtime which depends on the specific processing activity performed by the VM. However, conducting a real long-haul VM migration requires significant infrastructure such as a distributed OpenStack environment, EVPN-VXLAN capable routers, etc., to which we do not have access.

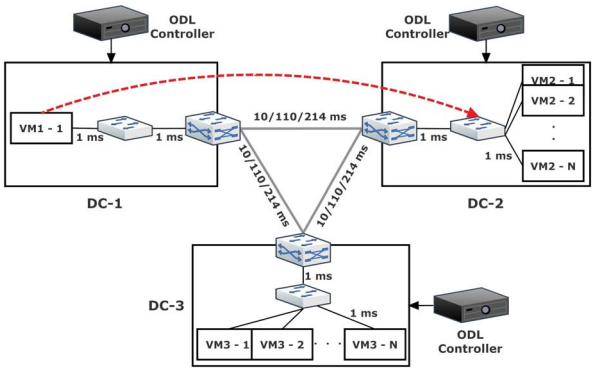


Fig. 6: Evaluation scenario

Migration: The migrating node is initially located in DC-1 and the migration destination is DC-2. In the intra-subnet experiment VM peers are placed in DC-3 while in inter-subnet experiment peers are in DC-2. Peers of migrating VM start to send traffic (Iperf or ping depends on the measurement goal) at t=0. The VM migration procedure is emulated by conducting the following steps: 1) At t=5 s, the migrating VM sends a message to the controller, informing the controller about the forthcoming stop-and-copy phase. 2) Migrating VM sleeps for 1 second. 3) Mininet script detaches the VM from the switch to which it was connected initially and attaches it to the switch in DC-2.

We selected 1 second as the duration of stop-and-copy phase according to [6] which provides a consistent downtime of ≈ 1 second under different network conditions. As the focus of [6] is on optimizing the state migration procedure, we can assume that 1 second downtime only refers to the duration of the stop-and-copy phase. Note that the migration downtime is primarily affected by the VM workload (such as memory dirty rate, etc.) when the migration is triggered and is not necessarily influenced by the size of VM.

Emulation facts and assumptions: The experiments are conducted without background traffic and each emulation lasts until all clients send the pre-defined amount of traffic to their peers. Herein, we assume that EVPN is established between DCs and VXLAN tunnels are created. Hence, the controller responsibility is to route/re-route the traffic to the correct place before and after the migration. In order to measure the downtime (TDown), each peer VM sends ping requests every 1 ms. We captured the ping values in a file for each peer VM along with their time stamp. To evaluate the flow completion time (FCT), peer VMs send 250 MB of data to their corresponding peer in the DC-1 using Iperf.

A. Intra Subnet

Table I shows the average downtime for peer nodes in DC-3 for the three methods. All methods have almost the same performance when the latency between remote sites is small (10 ms) as presented in Table I. However, when the link latency between two remote sites is high (214 ms) the average downtime using MAC withdraw approach is approximately 400 ms and 700 ms lower in comparison to EVPN and mobile IP methods, respectively. The reason is that the controller in DC-1 informs the remote sites in advance about the migration. Therefore, controller in the remote site (DC-3) starts to re-route the traffic by modifying the corresponding flow rules, while the VM is in stop-and-copy phase. As a result, when the VM is up and running at the new location, the rules are already installed and packets are routed towards the new location, waiting for the VM to respond. The results also show that EVPN method, which postpones the network convergence to after the migration, performs better than the mobile IP approach since the traffic that is re-directed using tunnels experiences a larger round trip time (RTT) after the migration.

TABLE I: Average TCP performance for 20 peer VMs

	Mobile IP			EVPN			Withdraw		
Latency (ms)	10	110	214	10	110	214	10	110	214
T_{Down} (sec.)	1.0	1.3	1.9	1.0	1.2	1.6	1.0	1.1	1.2
Tput (Mbps)	24.2	8.4	2.5	39.1	12.2	3.8	47.6	16.5	6.0

Fig. 7 depicts the RTT over time for a single peer VM during the migration phase when the latency between remote sites is 214 ms. The latency is similar for EVPN and withdraw methods after the VM is migrated. The only difference is in the

[3]http://mininet.org/.

[4]http://openvswitch.org/.

time that the first ping is replied by the VM after the migration which shows the network convergence effect. In contrast, the RTT for the mobile IP method is significantly higher after the migration due to triangular routing. Furthermore, the mobile IP method experiences a larger downtime as depicted in Fig. 7.

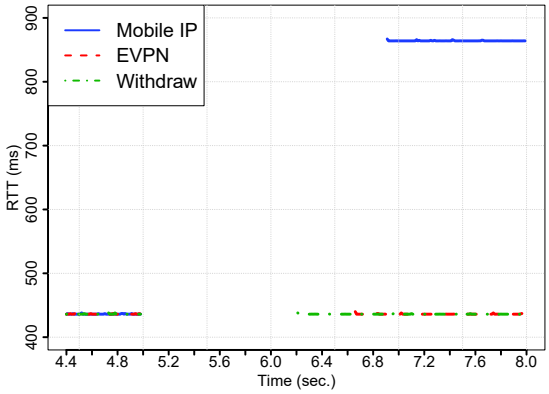


Fig. 7: RTT over time (ms)

The results of flow completion time for peer nodes (which are located at DC-3) are shown in Fig. 8. The controller of DC-3 modifies the flow rules on PE-3 upon receiving the MAC advertisement message in EVPN and withdraw scenario. In contrast, in mobile IP scenario DC-3 traffic continues passing through the source DC-1. Then, the SDN controller of DC-1 re-routes the traffic to the new VM location (DC-2). Akin to the previous experiments, Fig. 8 confirms that all three methods show the same performance when the delay between remote sites is small (Fig. 8a). In contrast, when the latency between remote site increases, the withdraw method significantly outperforms the EVPN and mobile IP methods. The flow completion time using withdraw method is approximately 35 seconds smaller than the EVPN method in 95% percentile while both methods outperform the mobile IP solution when the delay between remote sites is 214 ms.

B. Inter Subnet

This scenario shows the importance of a distributed anycast gateway for migration in inter-subnet traffic. For this evaluation, the migrating node peers are located in DC-2. Moreover, the migrating node and its peers are assumed to be in different subnets. As a result, the gateway is required to route the traffic between them. In the hair-pinning effect, the controller in DC-1 conducts the routing regardless of the location of the nodes. In contrast, in default gateway scenario the controller in DC-2 takes the routing responsibilities since it is the closest gateway to nodes. As we mentioned earlier, the gateway information spreads over the network when EVPN connection is established between remote sites. The experiment is conducted for 10, 20, and 40 peers in DC-2 while the latency among the remote sites changes between 10 ms, 110 ms, and 214 ms.

The average downtime and throughput values for 40 peer nodes are shown in Table II. According to Table II, the distributed gateway method outperforms the centralized method due to following reasons:

1) Sending flows back and forth to DC-1 results in contention between the flows over limited resources in the WAN and introduces congestion on DC egress links.
2) Conducting the gateway functionality at the controller when the VM migrates to the new DC avoids the traffic from traversing a high latency path between DCs. Moreover, unlike the hair-pin effect, traffic between two nodes inside a DC can benefit from the higher aggregated bandwidth inside a DC.

TABLE II: Average TCP performance for 40 peer VMs

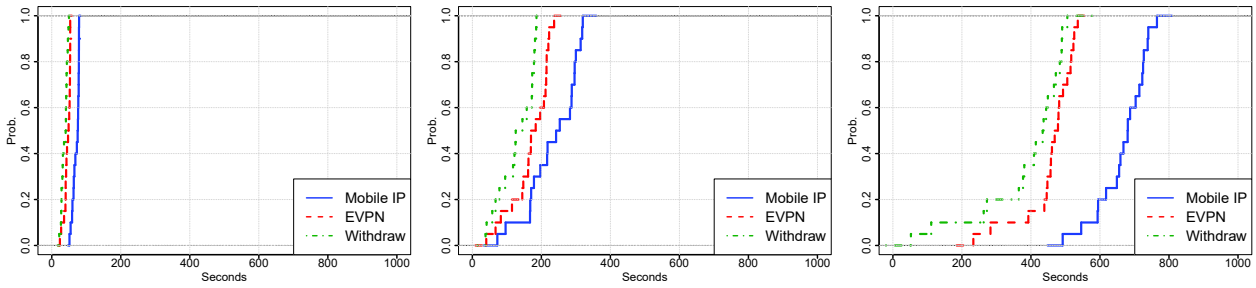
	Hair-pin			Distributed Gateway		
Latency (ms)	10	110	214	10	110	214
T_{Down} (sec.)	1.4	1.4	1.5	1.0	1.0	1.0
Tput (Mbps)	20.9	4.4	2.1	25.18	22.5	21.1

The flow completion time for hair-pinning and default gateway scenarios is illustrated in Fig. 9 when both the number of peers and the latency in the network are increased. Contention between flows over the DC-2 egress link decreases the application throughput, as shown in Fig. 9. The higher latency dramatically intensifies the problem and degrades the application performance significantly. Illustrated in Fig. 9, the distributed gateway solution outperforms the hair-pin solution in all scenarios. However, when the migrating VM has a large number of ongoing connections and the migration happens between two sites with high latency, the hair-pin problem severely degrades the performance. In contrast, the distributed gateway solution increases the application performance by addressing the sub-optimal routing problem.

Finally, Fig. 10 shows the RTT values during the experiment for both methods when the latency between remote sites is 214 ms. Although the migrating VM and its peers are located in the same DC after the migration, due to hair-pinning effect the RTT remains unaffected. However, when the controller in DC-2 performs the gateway functionality, the RTT decreases significantly and subsequently reduces the downtime.

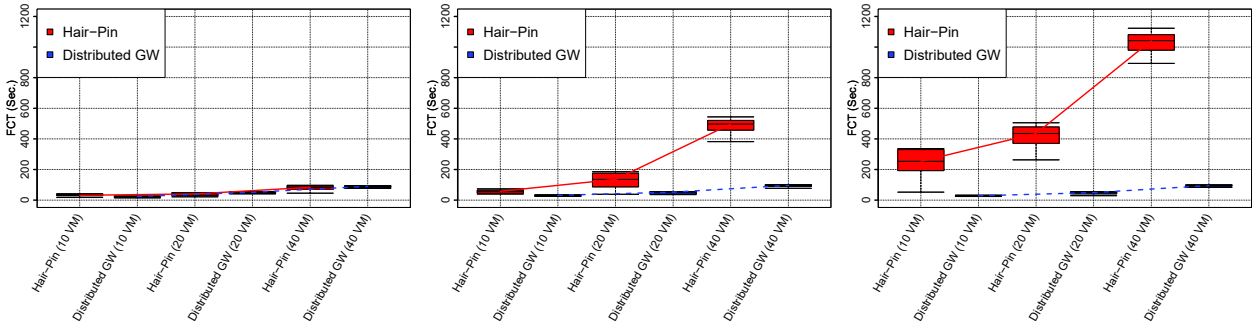
VI. Conclusions

In this paper, we have tackled the problem of improving the live VM migration process over the WAN. A key challenge is the higher latency and lower available bandwidth across the WAN compared to the live VM migration within a data center. In contrast to existing approaches, we coordinate the WAN live VM migration among different SDN controllers using extensions to MP-BGP. Consequently, our SDN-based approach proactively updates the flow rules of existing flows and re-establishes connectivity quickly which reduces the network downtime. Once the VM is up again at the new location, the packets are already redirected and the controller can re-optimize the paths later on. In addition, the SDN controller acts as a distributed anycast gateway for inter-subnet traffic which avoids the hair-pin effect. We implemented our extensions into the Opendaylight SDN controller. Our evaluation shows that our proposal effectively reduces the downtime leading to a more seamless live VM migration and increases the application performance.



(a) 10 ms (b) 110 ms (c) 214 ms

Fig. 8: Flow completion time



(a) 10 ms (b) 110 ms (c) 214 ms

Fig. 9: Flow completion time

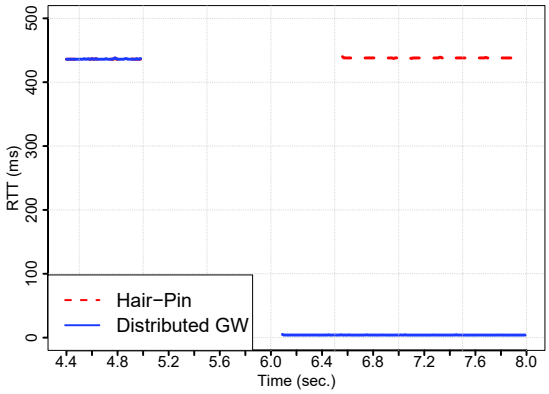


Fig. 10: RTT over time (ms)

ACKNOWLEDGMENT

Parts of this work has been supported by the Knowledge Foundation Sweden through the profile HITS.

REFERENCES

[1] "Mobile edge computing - introductory technical white paper," Tech. Rep., Sep. 2014.
[2] T. Taleb *et al.*, "Follow-me cloud: When cloud services follow mobile users," *TCC*, vol. PP, no. 99, pp. 1–14, 2017.
[3] W. Zhang *et al.*, "Adaptive live vm migration over a wan: Modeling and implementation," in *CLOUD*. IEEE, June 2014, pp. 368–375.
[4] T. Wood *et al.*, "Cloudnet: Dynamic pooling of cloud resources by live wan migration of virtual machines," *Transactions on Networking*, vol. 23, no. 5, pp. 1568–1583, Oct. 2015.
[5] R. Bradford *et al.*, "Live wide-area migration of virtual machines including local persistent state," in *VEE*. New York, NY, USA: ACM, June 2007, pp. 169–179.
[6] A. J. Mashtizadeh *et al.*, "Xvmotion: Unified virtual machine migration over long distance," in *ATC*. Berkeley, CA, USA: USENIX Association, 2014, pp. 97–108.
[7] C. E. Perkins, "IP mobility support for IPv4," RFC 3344, Aug. 2002.
[8] K. Kompella *et al.*, "Virtual Private Lan Service (VPLS) using BGP for auto-discovery and signaling," RFC 4761 (Proposed Standard), Internet Engineering Task Force, Jan. 2007.
[9] E. Silvera *et al.*, "IP mobility to support live migration of virtual machines across subnets," in *SYSTOR*. New York, NY, USA: ACM, 2009, pp. 13:1–13:10.
[10] P. Raad *et al.*, "Achieving sub-second downtimes in large-scale virtual machine migrations with LISP," *Transactions on Network and Service Management*, vol. 11, no. 2, pp. 133–143, June 2014.
[11] C. H. Benet *et al.*, "Minimizing live vm migration downtime using openflow based resiliency mechanisms," in *Cloudnet*. IEEE, Oct. 2016, pp. 27–32.
[12] S. Q. Zhang *et al.*, "Fast network flow resumption for live virtual machine migration on SDN," in *ICNP*. IEEE, Nov. 2015, pp. 446–452.
[13] A. Sajassi *et al.*, "BGP MPLS-based ethernet VPN," RFC 7432 (Proposed Standard), Internet Engineering Task Force, Feb. 2015.
[14] Opendaylight. [Online]. Available: https://www.opendaylight.org/
[15] E. Harney *et al.*, "The efficacy of live virtual machine migrations over the internet," in *VTDC*. IEEE, 2007, pp. 8:1–8:7.
[16] D. B. Johnson *et al.*, "Mobility support in IPv6," RFC 6275, Jul. 2011.
[17] V. Mann *et al.*, "Crossroads: Seamless vm mobility across data centers through software defined networking," in *NOMS*. IEEE, Apr. 2012, pp. 88–96.
[18] F. Hao *et al.*, "Enhancing dynamic cloud-based services using network virtualization," in *VISA*. ACM, Aug. 2009, pp. 37–44.
[19] B. Boughzala *et al.*, "Openflow supporting inter-domain virtual machine migration," in *WOCN*. IEEE, May 2011, pp. 1–7.
[20] J. Rabadan *et al.*, "Interconnect Solution for EVPN Overlay networks," Internet Engineering Task Force, Internet-Draft, Jul. 2017, work in Progress.
[21] K. A. Noghani *et al.*, "Automating ethernet VPN deployment in SDN-based data centers," in *SDS*. IEEE, May 2017, pp. 61–66.