Achieving Sub-Second Downtimes in Internet-wide Virtual Machine Live Migrations in LISP Networks

Patrick Raad *‡, Giulio Colombo †, Dung Phung Chi #*,
Stefano Secci *, Antonio Cianfrani †, Pascal Gallard ‡, Guy Pujolle *
* LIP6, UPMC, 4 place Jussieu 75005, Paris, France. Email: firstname.lastname@lip6.fr
† U. Roma I - La Sapienza, P.za Aldo Moro 5, 00185 Rome, Italy. Email: lastname@diet.uniroma1.it
# Vietnam National University (VNU), Computer Science dept., Hanoi, Vietnam. Email: dungpc@vnu.edu.vn
‡ Non Stop Systems (NSS), 27 rue de la Maison Rouge, 77185 Lognes, France. Email: {praad, pgallard}@nss.fr

***Abstract*—Nowadays, the rapid growth of Cloud computing services is stressing the network communication infrastructure in terms of resiliency and programmability. This evolution reveals missing blocks of the current Internet Protocol architecture, in particular in terms of virtual machine mobility management for addressing and locator-identifier mapping. In this paper, we propose some changes to the Locator/Identifier Separation Protocol (LISP) to cope this gap. We define novel control-plane functions and evaluate them exhaustively in the worldwide public LISP testbed, involving four LISP sites distant from a few hundred kilometers to many thousands kilometers. Our results show that we can guarantee service downtime upon virtual machine migration lower than the second across Asian and European LISP sites, and down to 300 ms within Europe. We discuss how much our approach outperforms standard LISP and triangular routing approaches in terms of service downtime as a function of datacenter-datacenter and client-datacenter distances.**

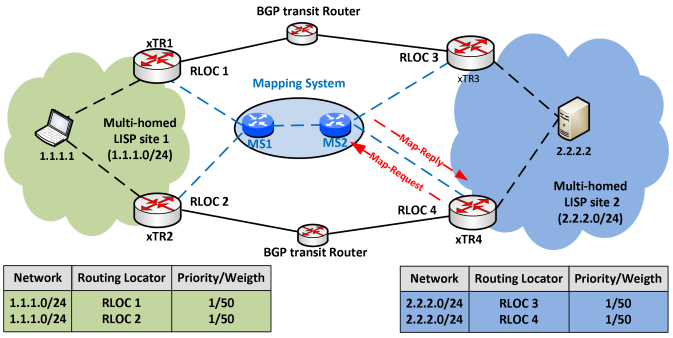


Fig. 1. LISP communications example

I. Introduction

Virtualization has been revolutionizing datacenter networking since a few years. Once solely based on physical server and mainframe interconnections, Cloud datacenters evolve with an increasing deployment of virtualization servers hosting, sending and receiving virtual machines (VMs), to and from local and distant locations. This evolution is arising many networking issues in terms of address continuity and traffic routing. Should VMs maintain (or use) the same (or multiple) Ethernet and/or IP addresses upon migration, if, when, and how, have been and are still open research questions in Cloud Networking. Similar challenges appear with the emergence of advanced services such as Infrastructure as a Service (IaaS), often requiring multiple VMs physically located at different sites to communicate to each other and keep communicating with each other, as well as with Internet users, while moving across datacenters.

In virtualization servers, the hypervisor is a software-level abstraction module essential to manage several VMs at the same time on a physical machine. VM migration is a service included in most hypervisors to move VMs from one physical machine to another, commonly within a data center. Migrations are executed for several reasons, ranging from fault management, energy consumption minimization, and quality-of-service improvement. In legacy Cloud networks, VM location was bound to a single facility, due to storage area network and addressing constraints. Eventually, thanks to high-speed low-latency networks, storage networks can span metropolitan and wide area networks, and VM locations can consequently span the whole Internet for public Clouds.

Multiple solutions are being experimented to make VMs location independent. The objective is to allow transparent VM migrations by developing advanced functionalities at the hypervisor level [1]. This leads to the definition of addressing, bridging and routing functions at the hypervisor software level, creating a new area in networking called Software Defined Networking (SDN). Despite no formal definition of SDN exists, it basically predicates the separation of data-plane and control-plane functions to simplify network management and configuration, letting some of these functionalities being personalized by ad-hoc software elements.

In terms of addressing, the main problem resides in the possibility of scaling from private Clouds to public Clouds, i.e., seamlessly migrating a virtual server with a public IP across the Internet. Multiple solutions exist to handle addressing issues, ranging from simple ones with centralized or manual address mapping using MAC-in-MAC or IP-in-IP encapsulation, or both, to more advanced ones with a distributed control-plane supporting VM mobility and location management. Several commercial (non-standard) solutions extend (virtual) local area networks across wide area networks, such as [2], [3], and [4] differently handling layer-2 and layer-3 inter-working.

Among the standards to handle VM mobility and addressing issues, we can mention recent efforts to define a distributed control-plane in TRILL (Transparent Interconnection of a Lot of Links) architecture [5] to manage a directory that pilots layer-2 encapsulation. However, maintaining layer-2 long-distance connectivity is often economically prohibitive, a too

high barrier for small emerging Cloud service providers, and not scalable enough when the customers are mostly Internet users (i.e., not privately interconnected customers). At the IP layer, the addressing continuity can be guaranteed using ad-hoc VM turntables as suggested in [6], or Mobile IP as proposed in [7]. However, at an Internet scale, these approaches may not offer acceptable end-to-end delay and downtime performance, because of triangular routing and many-way signaling. Even if optimizations exist in mobile IP to avoid triangular routing, the concept of home and foreign agents is not necessary, and moreover solutions not modifying the end-host would be more scalable.

More recently, the Location/Identifier Separation Protocol (LISP) [8], mainly proposed to solve Internet routing scalability and traffic engineering issues, is now considered as a VM mobility control-plane solution and has already attracted the attention for some commercial solutions [9]. In order to efficiently handle locator-identifier mappings, LISP offers a distributed control-plane, decoupled from the data-plane. An advantage of LISP is that it can avoid triangular routing, with encapsulations performed at the first IP-LISP node, and not at the hypervisor level. Nevertheless, based on current standards and literature, there are missing functionalities to guarantee low VM migration downtimes with LISP. Moreover, there is no reproductable experimental study describing the achievable performance of Internet-scale VM mobility under LISP.

The objective of this paper is to define additional LISP functionalities and to show the achievable performance in large-scale live VM migration, providing all the elements to reproduce the results. Our solution is based on the definition of LISP control-plane messages to fast update ID-locator mappings, hence overcoming the long-latency of basic LISP mechanisms. We validate and evaluate our solution using the worldwide public www.lisp4.net testbed, piloting the four LISP sites of LIP6, UNIROMA, VNU and INRIA Sophia Antipolis, in three countries worldwide. The paper is organized as follows. Section II briefly presents the background. Section III describes our protocol extension proposition. Section IV reports experimental results. Section V concludes the paper and discusses future works.

II. Background

In this section we introduce the reader to networking issues related to live VM migration, presenting existing solutions at the state of the art, and we give a synthetic overview of the LISP protocol.

A. VM migration and IP mobility

Live VM migration is a feature introduced in recent hypervisors; it allows moving a running VM between two (physical) container hosts without disconnecting application (TCP and UDP) connections of VM's clients. For most of the hypervisors, live migration is limited to situations in which source and destination hosts look like connected to the same local area network. The main reason is that the machine being live migrated needs to keep the same routing view of the network (gateway, IP subnet) before and after the migration. Alternatively, in some legacy solutions, upon migration the VM changes its IP address, e.g., via the DHCP, to avoid the additional complexity needed to ensure that the origin IP address is not already used in the destination network, and to transfer the routing table.

In order to perform Internet-wide migrations with IP continuity, authors in [7] and [10] propose an IP mobility solution. The logic is implemented in the hypervisor, interacting with the VM before and after its migration to update IP addresses in the VM routing table. While [7] succeeds in bringing lower service downtime compared to [10], the hypervisor has to alter VM configuration to support the IP mobility feature, which leads to scalability concerns. Moreover, as the authors state, their solution performance is expected to worsen in large-scale global live migrations, because of the online signaling nature of the proposition and many-way signaling latencies.

Authors in [11] propose to adapt the Mobile IP (MIP) protocol [12] and signaling to pilot Internet-scale VM migrations, implementing the protocol in the hypervisor. They call their solution HyperMIP. HyperMIP is invoked whenever a VM is created, destroyed or migrated; as in MIP, it involves Home Agents (HA) to keep the connection alive. Whenever a VM changes a location, a tunnel is established between the HA and the source hypervisor to keep the client connected to the VM. The destination hypervisor then destroys the tunnel when the VM registers its new IP address to the HA. However, HyperMIP still introduces an important signaling overhead due to HA tunnel establishment.

Alternatively, to minimize signaling latencies, authors in [6] propose to use an external agent to orchestrate the migration from the beginning to the end, by proactively establishing circuits between the involved containers (source and destination hypervisors) offline, so as to rapidly switch the traffic upon migration. Upon migration, the agent redirects the traffic between the VM and the client by dynamically re-configuring IP tunnels. They achieve a near second network downtime while migrating machines across wide area networks, with a maximum network downtime around 3.8 seconds. Despite being a secure approach, with respect to [7], [10] and [11] their solution involves lower-layer control-plane technologies, hence can be much more costly.

B. Triangular routing solutions vs LISP rerouting

The above described propositions are complimentary with our proposition to use LISP to redirect the traffic coming from Internet clients, yet alone (i.e., without LISP) those techniques can allow redirect the traffic, offering however a higher path latency. From the IP routing perspective of an Internet client accessing a server running in the VM, the above described approaches can be considered as triangular routing (or indirect forwarding) solutions. The reason is that the traffic has to reach the VM source network and/or container before being encapsulated and sent to the new VM location.

C. Locator/Identifier Separation Protocol (LISP) overview

LISP implements an additional routing level on the top of the Border Gateway Protocol (BGP), separating the IP location from the identification using Routing Locators (RLOCs) and Endpoint Identifiers (EIDs). An EID is an IP address that identifies a terminal, whereas an RLOC address is attributed to a border tunnel router. LISP uses a map-and-encap scheme

at the data-plane level, mapping the EID address to an RLOC and encapsulating the packet into another IP packet before forwarding through the Internet transit. At the control-plane level, multiple RLOCs with different weights and priorities can be associated with an EID: for unipath communications, the least-priority RLOC corresponds to the one to be selected for encapsulation; when a subset or all of the RLOCs have the same priority value, load-balancing is performed on the equal-priority RLOC. RLOC priority and weight are assigned by the destination EID space owner using its LISP routers.

A LISP site is managed by at least one tunneling LISP router (xTR), which has a double functionality: IP packet encapsulation (packet received by a terminal; ingress functionality, or ITR) and packet decapsulation (packet received by the network; egress functionality, or ETR). The IP-in-IP encapsulation includes a LISP header transporting control functionalities and a UDP header allowing differentiation between data and control plane messages. For a better understanding, consider the example in Figure 1: the traffic sent to the 2.2.2.2 host is encapsulated by the source's ITR toward one of the two destination's RLOCs. The one with the best (lowest) priority metric is selected, which at reception acts as ETR and decapsulates the packet, before sending it to the destination. On the way back to 1.1.1.1, RLOC4 queries a mapping system and gets two RLOCs with equal priorities, hence performs load-balancing as suggested by the weight metric (RLOC1 is selected in the example's packet).

In order to guarantee EID reachability, LISP uses a mapping system that includes a Map Resolver (MR) and a Map Server (MS). As depicted in Figure 1, a Map Resolver holds a mapping database, accepts MAP-REQUESTs from xTRs and handles EID-to-RLOC lookups; a particular MAP-REQUEST message, called SOLICIT-MAP-REQUEST (SMR) can have a flag set (S bit) to solicit a MAP-REQUEST to self by the receiver (passing via the MR). A Map Server receives MAP-REGISTERs from ITRs and registers EID-to-RLOC in the mapping database [13].

For managing EID-to-RLOC mapping, two different architectures are proposed: LISP-ALT (Alternative Topology) [14] and DDT (Delegated Database Tree) [15], the first relying on BGP signaling primitives, the second being inspired by DNS. Due to lack of flexibility, LISP+ALT is now replaced by DDT on the LISP beta network (www.lisp4.net). It is worth noting that, in a given configuration, if two xTRs, exchanging traffic, use the same MS/MR for registering and resolving, when an xTR sends a MAP-REQUEST for an EID that belongs to the other xTR, the MS/MR does not need to use DDT and hence DDT does not add an additional mapping latency to the xTR-MS/MR path latency.

D. Existing LISP-based Mobility Management Solutions

In a LISP network context, the VM can keep the same IP, provided that the locator change is notified to the mapping system. Two mechanisms at the state of the art can perform this operation.

One is a host-based LISP implementation called LISPmob [16]: the host is itself a tiny xTR implementing basic data-plane control-plane functions, using the network-assigned IP as RLOC and registering mapping updates for its EID with the mapping servers. Essentially conceived for mobile equipment, LISPmob could also be installed in the VM; there would be, however, a problem with most current hypervisors that impose the VM external address to be in the same subnet before and upon migration, which practically limits the LISPmob usability only to situations where source and destination networks are either LISP sites themselves, or layer-2 over WAN solutions. In the first case, a double encapsulation is needed, which could increase mapping latency, overhead and MTU issues. There may also be scalability issues with a high VM number.

Another existing method to handle VM mobility in a LISP context is actually implemented in some Cisco products, only partially documented in [9]. The xTR automatically changes the mapping upon reception of outgoing data-plane traffic from an EID that has been registered as mobile node. The solution has an attracting light impact on LISP operations, yet it seems to be weak against EID spoofing, and it seems not to have authentication mechanisms. Moreover, in order to guarantee fast mapping convergence, additional logic may need to be implemented in the VM or in the hypervisor to allow sending outgoing artificial data traffic even if no real outgoing traffic exist.

III. Proposed LISP-Based VM Migration Solution

We propose a novel solution to support Internet-scale VM live migration exploiting the LISP protocol[1]. A live migration should be able to move a VM keeping its unique EID, from its actual DC to a new DC maintaining all VM connections active.

As a preliminary step, the source and destination DCs have to share the same internal subnet, i.e., the VM unique EID should be routable beyond its RLOC, wherever it is. LISP supports a large number of locators, and it does not set constraints on RLOC addressing – i.e., while EIDs should belong to the same IP subnet, the RLOCs can take an IP address belonging not simply to different subnets, but also to different Autonomous System networks. The current VM location can be selected leveraging on RLOC metrics. We introduce two main enhancements:

- a new LISP control-plane message to speed up RLOC priority update;
- a migration process allowing hypervisor-xTR coordination for mapping system update.

A. Change Priority Message Format

We have implemented a new type of LISP control-plane message called CHANGE PRIORITY (CP) (Figure 2). We use a new control-plane type field value equal to 5[2], and use two bits to define message sub-types to be managed by both xTR and VM containers' hypervisors:

- **H (Hypervisor) bit**: this bit is set to 1 when the message is sent by the destination hypervisor (the

[1] the proposed solution has been implemented in the OpenLISP control-plane [17].

[2] A preliminary version of this new control-plane message has been presented at the first LISP Network Operator Group (LNOG) - http://www.lisp4.net/lnog.

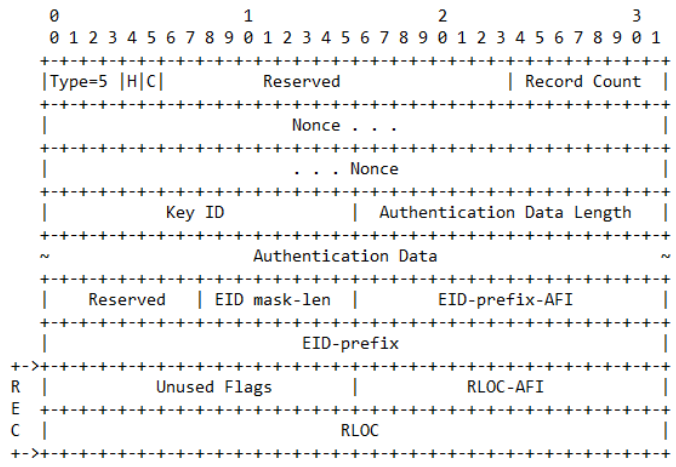


Fig. 2. CHANGE-PRIORITY message architecture

hypervisor that receives the VM), indicating to the xTR that it has just received a new EID. With the H bit set, the record count should be set to 0 and the REC field is empty;

- **C (Update Cache) bit**: this bit is set to 1 when an xTR wants to update the mapping cache of another xTR. With the C bit set, the record count is set to the number of locators and the REC field contains the RLOC information to rapidly update the receiver mapping cache.

The other fields have the same definition as the MAP-REGISTER message fields [8], i.e., with EID and RLOC fields, a nonce field used to guarantee session verification, and HMAC authentication fields useful to secure the communication (with the important feature that the authentication key used for CP messages can be different than the key used by MAP-REGISTER, provided that the xTR is able to handle different keys as provided in [17]).

B. VM migration process

The LISP mapping system has to be updated whenever the VM changes its location. Before the migration process starts, the xTRs register the VM's EID as a single /32 prefix or as a part of larger EID (sub-)prefix. The involved devices communicate with each other to *atomically* update the priority attribute of the EID-to-RLOC mapping database entries. The following steps describe the LISP-based VM migration process we propose and demonstrate.

1) The migration is initialized by the hypervisor hosting the VM; once the migration process ends, the destination hypervisor (the container that receives the VM) sends a CP message to its xTR (also called *destination xTR*) with the H bit set to 1, and the VM's EID in the EID-prefix field.
2) Upon reception, the *destination xTR* authenticates the message, performs an EID-to-RLOC lookup and sets the highest priority to its own locators in the mapping database with a MAP-REGISTER message. Then, it sends a CP message, with H and C bits set to 0, to update the mapping database of the xTR that was managing the EID before the migration (also called *source xTR*).
3) Before the VM changes its location, the *source xTR* keeps a trace file of all the RLOCs that have recently requested it (we call them *client xTRs*), i.e., that have the VM RLOCs in their mapping cache.
4) When the *source xTR* receives the CP message from the *destination xTR*, it authenticates it and updates the priorities for the matching EID-prefix entry in its database.
5) In order to redirect the client traffic, there are two different client-redirection possibilities, whether the *client xTR* is a standard router not supporting CP signaling (e.g., a Cisco router implementing the standard LISP control-plane [8]), or an advanced router including the CP logic (e.g, using the OpenLISP control plane [18]).
   - For the first case, the *source xTR* sends a SMR to standard *client xTRs*, which triggers mapping update as of [8] (MAP-REQUEST to the MR and/or to the RLOCs, depending on the optional usage of mapping proxy, followed by a MAP-REPLY to the xTR).
   - For the second case, in order to more rapidly redirect the traffic to the VM's new location (*destination xTR*), the *source xTR* sends a CP message with C bit set to 1 directly to all the OpenLISP *client xTRs*, which will therefore process it immediately (avoiding at least one client xTR-MR round-trip-time).
6) Upon EID mapping update, the *client xTRs* update their mapping cache and start redirecting the traffic to the VM's new routing locator(s).

Algorithm 1 CP processing

Ensure: authenticity of CP message
  extract EID from EID-prefix field
  if H bit is set to 1 **then**
    set own locators' priority to 1
    send CP to xTR group with H bit and C bit set to 0
    register mapping to Map Server
  end if
  if H bit and C bit are both set to 0 **then**
    set own locators' priority to 255
    set locators' priority in RLOC field to 1
    send CP with C bit set to 1 to all locators that have requested the VM's EID
    stop registering for EID
  end if
  if C bit is set to 1 **then**
    update mapping cache according to the received message
  end if

C. Implementation aspects

The proposed solution has been implemented using open-source software (see [17]), and its implementation involves both the hypervisor and the xTR sides.

1) On the hypervisor: we integrated a new function that interacts with libvirt (a management kit handling multiple

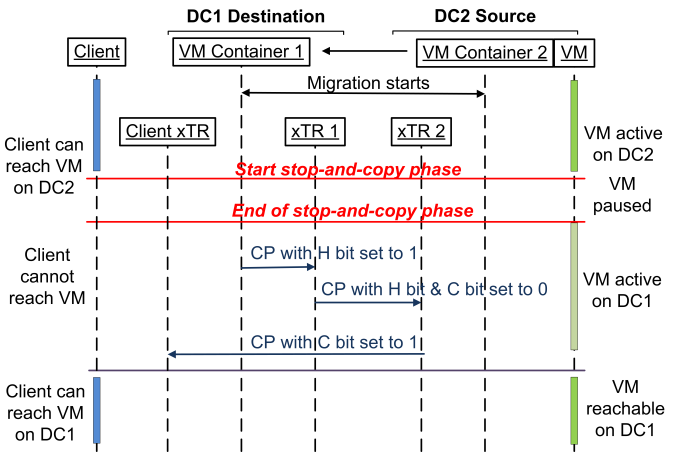


Fig. 3. Example of CP signaling exchange during a VM migration

VMs in the KVM hypervisor) [19] to trigger CP message generation. When a live migration starts, the hypervisor creates a "paused" instance of the VM on the destination host. Meanwhile, libvirt monitors the migration phase from the start to the end. If the migration is successfully completed, libvirt checks if the VM is running on the target host and, if yes, it sends a CP message to its xTR on the UDP LISP control port 4342. The VM EID is included in the EID-prefix field.

2) On the xTR: we implemented the Algorithm 1 function in the OpenLISP control-plane [18], providing control-plane features to the OpenLISP data-plane [20]; the OpenLISP data-plane router runs in the kernel of a FreeBSD machine, while the control-plane runs in the user space. The control-plane has a new feature to capture control-plane message type 5 and the logic to handle CP signaling.

3) A signaling example: upon client's request, or a consolidation engine, a VM needs to be migrated to another public DC. As in the Figure 3 example, VM Container 2 starts migrating VM from DC2 to DC1 while Client is still connected. When the migration reaches the stop-and-copy phase, the VM stops and begins transferring its last memory pages. Meanwhile, Client loses the connection but keeps directing the traffic to DC2.

The hypervisor on VM Container 1 detects that VM is now successfully running, indicating the end of the migration. Then VM Container 1 announces that VM has changed its location by sending a CP message with the H bit set to xTR 1. Upon reception, xTR 1 sends a CP with H bit and C bit set to 0 to notify xTR 2 about the new location of VM: xTR 1 updates the priorities for VM's EID entry in its database.

When xTR 2 receives the CP message, it matches the EID-prefix to the entries within its mapping database, and modifies the priorities accordingly, then it stops registering VM's EID. As mentioned in Section III-B, xTR 2 keeps a trace file of all the locators that recently requested the VM's EID. In this use case, Client is the only client that is communicating with VM. As a result, xTR 2 sends a CP with C-bit set to Client xTR.

Client xTR receives the CP message, maps VM's EID, and updates its cache, then starts redirecting Client's traffic to VM's new location (DC1).

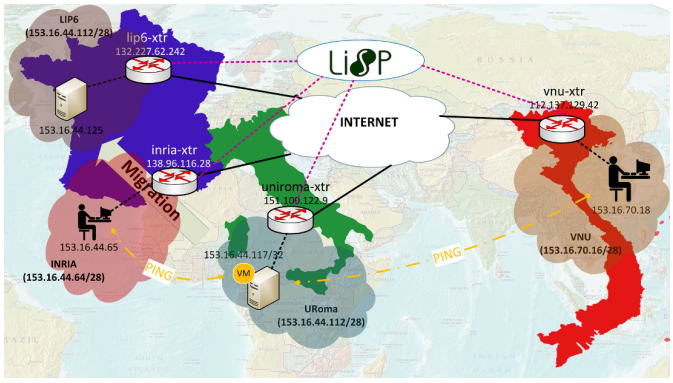


Fig. 4. LISP testbed topology

IV. Testbed Evaluation

We performed live migrations of a FreeBSD 9.0 VM, with one core and 512 MB RAM (corresponding to a typical service VM like a lightweight web server), from UROMA1 (Rome) to LIP6 (Paris), under KVM [21]. Figure 4 gives a representation of the testbed topology. As distributed storage solution, we deployed a Network File System shared storage between source and destination host containers. Hence, only RAM and CPU states are to be transferred during the live migration. The VM containers are Ubuntu 12.04 servers, dual core, with 2048 RAM and using KVM and Libvirt 0.9.8.

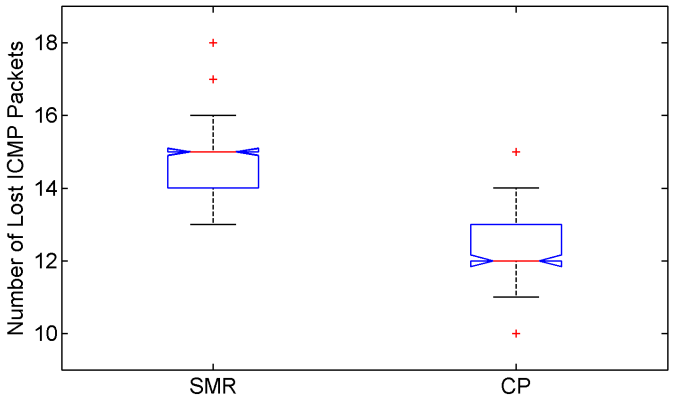
We measured many parameters during migrations, by 20 ms spaced pings from different clients: distant ones at VNU (Hanoi, Vietnam) LISP site, and a close one at the INRIA (Sophia Antipolis, France) LISP site. We should note that:

- the clocks on all LISP sites were synchronized to the same Network Time Protocol (NTP) stratum [22], so that a same VM migration can be monitored concurrently at the different client sites;
- all LISP sites' xTRs register to a same Map Server/Map Resolver located in Denmark (www.lisp4.net), hence avoiding the DDT latency in the mapping convergence (as already mentioned in Section II-C).

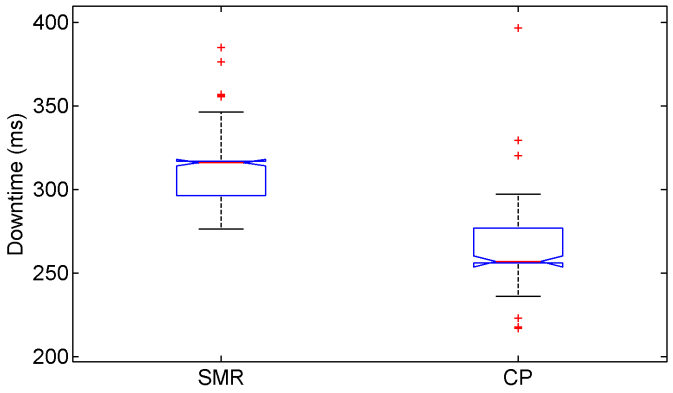
We performed hundreds of migrations from the UROMA1 to the LIP6 sites, over a period of 3 months at different times of the day. We used the two possible inter-xTR mapping update modes with the proposed control-plane enhancement: SMRs to simulate standard client xTRs, and CP to encompass the case with enhanced xTRs at client LISP sites. Given the Internet wideness of the testbed, both the bottleneck bandwidth and RTTs were floating, depending by the time and day, hence we did a statistical evaluation as described hereafter. The average measured RTTs between each site during the migration are reported in Table I; having both close and far clients' sites allow us to precisely assess the migration performance.

In order to experimentally assess the relationship between different time components and network situations, we measured these different parameters:

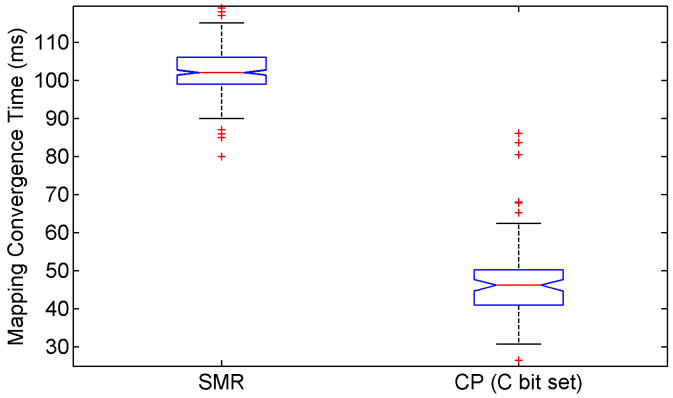
- number of lost packets for each client (i.e., the number of ICMP messages that are lost on each client during migration);



(a) Number of lost packets

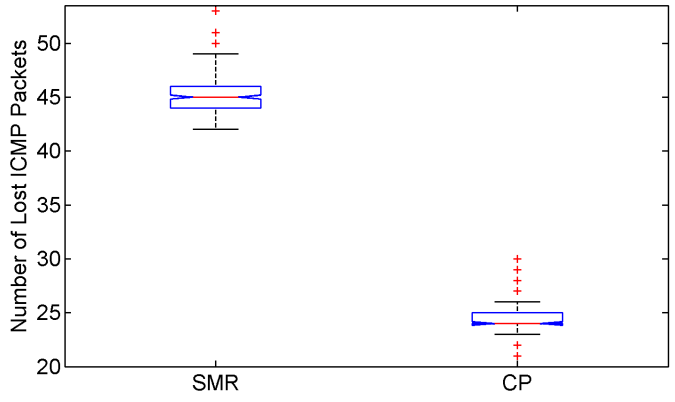


(b) Downtime

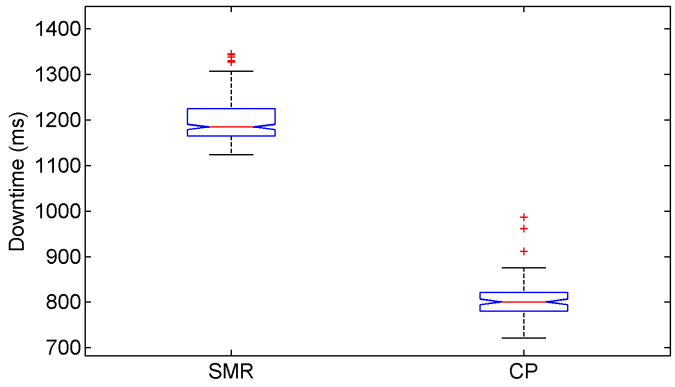


(c) Mapping convergence time

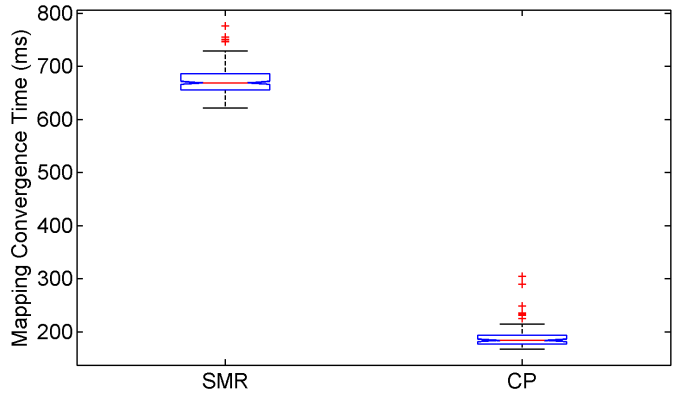
Fig. 5. INRIA client result parameters (boxplots statistics)



(a) Number of lost packets



(b) Downtime



(c) Mapping convergence time

Fig. 6. VNU client result parameters (boxplots statistics)

- mapping convergence time for each client: the time between the transmission of CP by the hypervisor and the mapping cache update on each client.
- downtime perceived by each client: the time during which the client could not communicate with the VM.

As depicted in Figure 7 and as of previous arguments, it is worth underlining that one should expect that: $downtime \geq$ downtime introduced by the hypervisor (stop-and-copy[3] duration) + the mapping convergence time. The stop-and-copy duration depends on the volume of the last memory pages to be transferred that, with standard tools, we do not control. The mapping convergence time reflects our protocol overhead, which is differently affected by the RTT between LISP sites (Table I) depending on the client xTR support of CP signaling.

In order to characterize absolute service downtimes suffered by clients, Figures 5,6 report the boxplots (minimum, 1^{st} quartile, median with the 95% confidence interval, 3^{rd} quartile, maximum, outliers) of the obtained number of lost packets, downtime, and mapping convergence time. We measured the results with the two possible modes for inter-xTR mapping update, using SMR signaling and using CP signaling.

[3]During the stop-and-copy phase the VM stops on the source container in order to transfer the rest of the memory pages, those that have been recently used by the VM (also called dirty pages) [23].

4) Using SMR *signaling:* as explained in Section III-B, as of LISP standard control-plane, the SMR message is sent by an xTR to another to solicit mapping update for a given EID. Upon reception of a SMR, the target xTR sends a MAP-REQUEST to its map-resolver that forwards it as an encapsulated message down to the source xTR (if MAP-REPLY proxy is not enabled as by default In Cisco routers and as is in our simulations), followed by a MAP-REPLY. The overall SMR signaling time should therefore be lower bounded by one and a half the RTT between the two xTRs, which impacts the mapping convergence time and hence the service downtime. As of our experimentations, we obtained a median downtime of about 320 ms for the INRIA client (Figure 5(b)), 1.2s for VNU (Figure 6(b)). This large gap between the downtimes of close and distant clients can be explained not only by the distance that separates each client from the VM, impacting the propagation delay (see Table I), but also by the fact that the Map Resolver is closer to INRIA than to VNU, as mentioned in Section IV. We find this gap also in the number of lost ICMP packets, two to three times higher for distant clients than for close ones (Figure 5(a), Figure 6(a)).

5) Using CP *signaling:* as explained in Section III-B, using CP signaling the mapping convergence time can be decreased of at least one RTT between xTRs, with an authenticated one-way message that directly updates xTR cache upon reception. For the INRIA client, we obtain a median downtime of 260 ms gaining a few dozens of ms, whereas we could gain 200 ms for VNU . Moreover, we notice that the number of lost ICMP packets for distant clients has exponentially decreased. This important decrease is due to the fact that xTRs have no longer to pass via the Map Resolver to update their mapping cache. Finally, Figures 5(c),6(c) show that the mapping convergence time component of the downtime decreases with CP signaling for all cases. While it is roughly between one-third and one-half the downtime with SMR signaling, it falls to between one-sixth and one-third with CP signaling, and this ratio is higher for distant clients. This implies that the hypervisor downtime (stop-and-copy phase) is less sensible to the RTT than the mapping convergence is (likely, the last page transfer profits from an already established TCP connection with an already performed three-way handshake).

LISP Sites	Average RTT
LIP6-UROMA1	30.47 ms
LIP6-VNU	299.86 ms
LIP6-INRIA	16.47 ms
UROMA1-VNU	321.27 ms
UROMA1-INRIA	27.27 ms

TABLE I. AVERAGE MEASURED RTT DURING MIGRATIONS

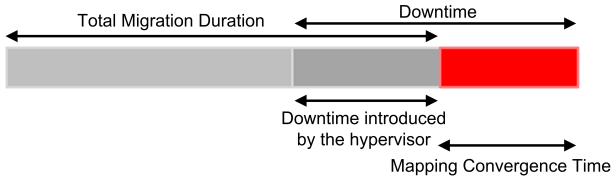


Fig. 7. Migration duration and downtime composition

A. Comparison with alternative solutions

In conclusion, our approach offers quite interesting performance, and we show the advantage of extending the signaling logic to clients' xTRs. Our solution offers a median mapping convergence overhead that varies from 50 ms for nearby clients (within a few hundreds km) to 200 ms for distant client (at many thousands km), depending on the signaling scope. With respect to described alternative methods at the state of the art (see Section II), fully handled at the hypervisor level, we can assess that:

- with HyperMIP [11], authors experienced a 100 to 400 ms of overhead, which is almost two times more than our approach, the main reasons being the usage of Mobile IP and triangular routing;
- similarly in [10] Mobile IPv6 signaling is used to detect VM location change, reaching a minimum overhead around 2500 ms, linearly increasing with the network delay, hence largely higher than our approach;
- authors in [6] went a step further implementing pro-active circuit provisioning, reaching an application downtime varying between 800 and 1600 ms, which is more than 4 times higher than with our approach.

V. CONCLUSION

In this paper, we have proposed a novel LISP-based solution for VM live migrations across distant datacenters over the Internet. We tested it via the global LISP testbed. We can summarize our major contributions as follows:

- we have defined and implemented a new type of LISP control-plane message to update VM location upon migration, with the interaction between hypervisors and LISP routers[4];
- we performed hundreds of Internet-wide migrations between LISP sites (LIP6 - Paris, UROMA1 - Rome) via the LISP testbed, including the case of clients close to source and destination containers (INRIA - Sophia Antipolis), and the case of distant clients (VNU - Hanoi);
- by exhaustive statistical analysis on measured relevant parameters and analytical discussions, we have characterized the relationship between the service downtime, the mapping convergence time and the RTT.
- we showed that with our approach we can easily reach sub-second downtimes upon Internet-wide migration, even for very distant clients.

We plan to extend the migration measurement campaign including other distant LISP sites.

ACKNOWLEDGMENT

The authors would like to thank Damien Saucez from INRIA Sophia Antipolis for running LISP site clients probing.
.

[4]The LISP control-plane code with related functionalities is publicly available in [17]. Part of the CP signaling logic was implemented into libvirt.

This work was partially supported by the French "Investissement d'Avenir" NU@GE project (http://www.nuage-france.fr).

REFERENCES

[1] M. Nelson et al., "Fast transparent migration for virtual machines," in *Proceedings of the annual conference on USENIX Annual Technical Conference*, pp. 25–25, 2005.

[2] S. Setty, "vMotion Architecture, Performance, and Best Practices in VMware vSphere 5," tech. rep., VMware, Inc., 2011.

[3] Cisco and VMware, "Virtual Machine Mobility with VMware VMotion and Cisco Data Center Interconnect Technologies," Tech. Rep. C11-557822-00, August 2009.

[4] Cisco, "Cisco Overlay Transport Virtualization Technology Introduction and Deployment Considerations," tech. rep., Cisco Systems, Inc., January 2012.

[5] L. Dunbar et al., "TRILL Edge Directory Assistance Framework." draft-ietf-trill-directory-framework-00, February 2012.

[6] F. Travostino et al., "Seamless live migration of virtual machines over the MAN/WAN," *Future Generation Computer Systems*, vol. 22, pp. 901–907, Oct. 2006.

[7] H. Watanabe et al., "A Performance Improvement Method for the Global Live Migration of Virtual Machine with IP Mobility," in *Proc. ICMU 2010*, 2010.

[8] D. Lewis et al., "Locator/ID Separation Protocol (LISP)." draft-ietf-lisp-22, February 2012.

[9] Cisco, "Locator ID Separation Protocol (LISP) VM Mobility Solution," tech. rep., Cisco Systems, Inc., 2011.

[10] E. Harney et al., "The efficacy of live virtual machine migrations over the internet," in *Proceedings of the 2nd international workshop on Virtualization technology in distributed computing*, p. 8, ACM, 2007.

[11] Q. Li et al., "Hypermip: hypervisor controlled mobile ip for virtual machine live migration across networks," in *High Assurance Systems Engineering Symposium, 2008. HASE 2008. 11th IEEE*, pp. 80–88, Ieee, 2008.

[12] C. Perkins, "IP mobility support for IPv4." IETF RFC 3344, 2002.

[13] V. Fuller and D. Farinacci, "LISP Map Server Interface." draft-ietf-lisp-ms-16, March 2012.

[14] D. Lewis et al., "LISP Alternative Topology (LISP+ALT)." draft-ietf-lisp-alt-10.

[15] D. Lewis and V. Fuller, "LISP Delegated Database Tree." draft-fuller-lisp-ddt-01, March 2012.

[16] C. White et al., "LISP Mobile Node." draft-meyer-lisp-mn-08, Oct. 2012.

[17] http://www.lisp.ipv6.lip6.fr, "OpenLISP control plane."

[18] D. Phung Chi et al., "An open control-plane implementation for LISP networks," in *Proc. IC-NIDC 2012*, 2012.

[19] "libvirt: The virtualization API." http://libvirt.org/.

[20] L. Iannone et al., "OpenLISP: An Open Source Implementation of the Locator/ID Separation Protocol," *ACM SIGCOMM, Demo paper*, 2009.

[21] A. Kivity et al., "kvm: the Linux virtual machine monitor," in *Proceedings of the Linux Symposium*, vol. 1, pp. 225–230, 2007.

[22] D. Mills, "Internet time synchronization: the network time protocol," *Communications, IEEE Transactions on*, vol. 39, no. 10, pp. 1482–1493, 1991.

[23] C. Clark et al., "Live migration of virtual machines," in *Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation - Volume 2*, NSDI'05, (Berkeley, CA, USA), p. 273286, USENIX Association, 2005.