HAL
archives-ouvertes.fr

Control Protocol and Self-adaptive Mechanism for Live Virtual Machine Migration over XIA

Dalu Zhang, Xiang Jin, Dejiang Zhou, Jianpeng Wang, Jiaqi Zhu

▶ **To cite this version:**

Dalu Zhang, Xiang Jin, Dejiang Zhou, Jianpeng Wang, Jiaqi Zhu. Control Protocol and Self-adaptive Mechanism for Live Virtual Machine Migration over XIA. 11th IFIP International Conference on Network and Parallel Computing (NPC), Sep 2014, Ilan, Taiwan. pp.357-368, 10.1007/978-3-662-44917-2_30 . hal-01403105

HAL Id: hal-01403105
https://hal.inria.fr/hal-01403105

Submitted on 25 Nov 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution| 4.0 International License

Control Protocol and Self-adaptive Mechanism for Live Virtual Machine Migration over XIA*

Dalu Zhang **, Xiang Jin, Dejiang Zhou, Jianpeng Wang, and Jiaqi Zhu

Department of Computer Science and Technology, Tongji University, Shanghai, China
daluz@acm.org,
{jinxiang8910, dejiang_zhou, wangjianpeng4321, garyzjq}@163.com

Abstract. FIA (Future Internet Architecture) is supported by US NSF for future Internet designing. XIA is one of the projects which comply with clean slate concept thoroughly. Meanwhile, virtual machine migration technique is crucial in cloud computing. As a network application, VM migration should also be supported in XIA. This paper is an experimental study aims at verifying the feasibility of VM migration over XIA. We primarily present intra-AD (Administrative Domain) and inter-AD VM migration with KVM instances. The procedure is achieved by a migration control protocol which is suitable for the characters of XIA architecture. Moreover, an elementary self-adaptive mechanism is introduced to maintain VM connectivity and connection states. It is also beneficial for VM migration in TCP/IP network. Evaluation results show that our solution well supports live VM migration in XIA and all the communications leading to VM can be kept uninterrupted after migration.

1 Introduction

For decades, Internet has become one of the most useful tools in our daily life. It achieved great flourish because of large quantity of applications and various kinds of media. However, TCP/IP network at present is suffering from serious issues such as difficulties on scalability, mobility and security problems. This leads to the emergence and development of future network.

XIA is one of the FIA projects supported by US NSF in 2010. It is also one of the FIA-NP (Next Phase) projects announced in May 2014. XIA [1] aims at getting rid of TCP/IP concepts. Network, host, service and content can be abstracted as *principals*. New principal types can be defined for special use, even if they have not been natively supported [2]. Network address is replaced by DAG (Directed Acyclic Graph), which is flexible for addressing. In addition, *fallback* allows communicating entities to choose an alternative action if intent node is unreachable. Addresses are managed by name service, which provides a mapping converting human-readable names to DAGs.

* Supported by the NSF of China (No. 61073154)
** Corresponding author

Virtualization is necessarily a key technology in cloud computing, which allows to run multiple operating systems on a single platform, utilizing host's expensive resources independently, such as CPU cycles and memory space. Data centers can achieve load balancing, host maintenance, energy management or disaster recovery [3] by VM migration.

Extensive research has been carried out on VMs with shared-storage [4], but shared-storage VMs cannot be applied in all scenarios [5]. For example, a user may not necessarily have access to a particular data center permanently. If a shared virtual disk is allocated through network, abnormal latency would occur. Thus, we prefer full VM migration, during which virtual disk is transferred as well as memory and CPU information. We choose KVM since block (storage) migration is intrinsically supported. KVM is a full virtualization solution frequently used in research area. KVM module is integrated in the kernel of common Linux distributions. However, direct kernel modules access is not permitted for users. This issue can be solved by QEMU which is also a piece of open-source virtualization software and is adopted as a management tool in user space.

In this paper, we first design VM migration platform in two different situations, in a single AD and between ADs. Since VM migration in XIA network is quite different to that in TCP/IP network, we adopt VM migration control protocol to manage migration procedure. Moreover, a current-network based self-adaptive mechanism is introduced to keep up all the connections leading to VM. We can achieve VM migration over XIA with the control protocol and self-adaptive mechanism. The experimental results demonstrate that VMs can get migrated with downtime no longer than 2s for full migration over XIA network.

Virtualization is necessary to maintain unified management for various cloud computing platforms, even in future data centers. It is also beneficial to keep user diversity and application isolation. Therefore, virtual machine will be in existence for a long period of time in the future. As a typical application in future networks, it should also be well supported in XIA. Base on the research about VM migration over traditional TCP/IP network, we try to study VM migration techniques in future Internet. On the one hand, it can test whether VM migration is supported in XIA. On the other hand, in comparison with the technologies used for VM migration in TCP/IP network, it is beneficial to perfect the design of future networks.

The rest of this paper is organized as follows: In Sect. 2 we discuss related work. In Sect. 3, we introduce the VM migration system design and migration modules processing migration. After that, we demonstrate the control protocol and self-adaptive mechanism to achieve VM migration in Sect. 4. Sect. 5 further discusses the experimental results. Some special issues and future works are discussed in Sect. 6 and Sect. 7 concludes this paper.

2 Related Work

Most of recent researches on VM migration are dedicated in studying the mechanisms of VM migration and factors that trigger it. In general, VM migration

method can be mainly classified as pre-copy [6], post-copy [7]. There are also some optimizations based on pre-copy algorithms, such as transferring bitmaps[8] or log file [9] of dirty pages, or delivering "hot pages" in final round [10] to minimize the number of pages being transferred.

In TCP/IP network, an important issue is to get the migrated VM noticed by all the network elements after migration. It can be achieved by generating an unsolicited ARP reply on destination host, advertising location change of VM. But the fact is that this method may not be effective in all scenarios. If source host and destination host are in different subnets, some hosts or routers would not receive the ARP messages broadcasted by the migrated VM because of network isolation.

In order to solve the problem, lots of researches has been carried out, which can be classified into two categories. One is based on the concept of mobile IP. Article [11] presented to build a tunnel between original address and the new address so as to keep all the communications that have been set up before. In addition, dynamic DNS is utilized to record address update, so clients can connect to VM by obtaining the new address after VM migration. Network agents [12] are presented to be set in both source and destination subnet with ARP agents maintained on. The ARP agent in source subnet will broadcast ARP messages to advertise the information of VM's new location. But the limitation is that clients should locate in source subnet, otherwise, they cannot receive the ARP messages. Mobile IPv6 is introduced in [13], one of the benefits is that hosts supporting Mobile IPv6 can bypass the tunnel and connect to the VM through route optimization mode.

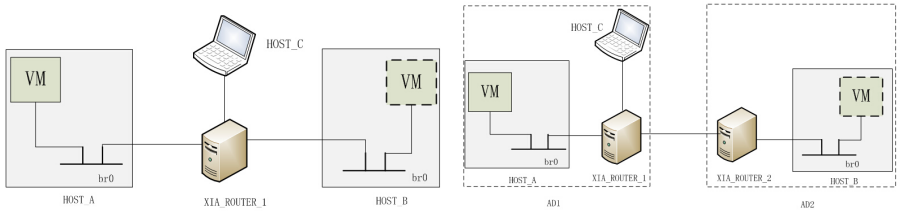
The other method depends on overlay network. In [14], source and destination network of VM migration procedure are repartitioned into the same VPN. ARP message can be forwarded at VPN level to update Ethernet switch mappings at both sites. This will help to redirect network traffic to VM's new location. In ViNe [15], hosts can be addressed by virtual network addresses first. Overlay network methods require to build virtual network before migration occurs, while in fact VM migration is triggered by some particular factors, virtual networks should be reorganized for migration each time since the source and destination network may not be constant.

We seldom find research about VM migration technologies over future Internet architectures. Therefore, it is contributory for us to conduct virtual machine migration research on XIA. It is contributive for the design of future networks.

3 System Design

3.1 Testbed Design

Our research goal can be summarized as four rules which we name as *"four any"*, that is, VMs can be deployed on any physical host and be migrated to any one, name service can run on any host in the network and any of the applications should not be interrupted during VM migration.



(a) Intra-AD (b) Inter-AD

Fig. 1: VM migration testbed design in XIA network, (a) is the testbed in single AD and (b) is a typical testbed for VM migration between different ADs.

The XIA prototype is developed base on software router click[16]. Common routers in TCP/IP network can't be recognized in XIA because of the differences in protocol formats. As a solution, XIA routers are realized by physical machines with two or more NICs. Name service is necessary for host address query in XIA. Any host can run as name server and provide global name resolution service. When the addresses of hosts or services are changed, they should be registered to name server.

AD is introduced in XIA to partition the network. It is convenient for network management. A VM in different AD will obtain different addresses as the AD number is changed. Therefore, inter-AD VM migration is more complex than that in a single AD. We propose two VM migration testbeds, in single AD and between ADs, concerning the issue of whether DAG has to be changed. Fig. 1a shows the testbed of VM migration in single AD. HOST_A, HOST_B and HOST_C represent the source host, destination host and client host respectively. HOST_VM depicted in dashed box denotes the virtual machine to be migrated. Name service runs on HOST_A because it can be put on any hosts. ROUTER_1 indicates a XIA router, routing and forwarding packets. In Fig. 1b, the topological structure is partitioned into two independent ADs and each XIA router manages its AD respectively. Name service still runs on HOST_A. All the hosts or services can register their DAG-style addresses to name service.

3.2 Migration Control Modules

We attempt to conduct VM migration in XIA network with now available virtualization product KVM and take the advantages of CHUNK provided in XIA for data delivery. The whole structure of VM migration consists of three modules and their relationship are shown in Fig. 2.

Migration Data Sending and Receiving There are four migration modes in KVM. Among them, *tcp* mode is primarily used for VM migration in TCP/IP network by default. In *exec* mode, migration data are read and sent to standard I/O by the sender, while on the receiver side KVM hypervisor obtains data from

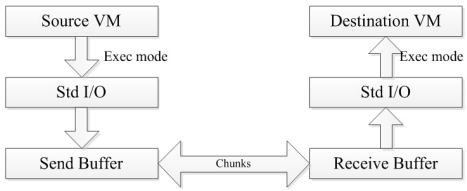


Fig. 2: Data flow between different modules for exec mode VM migration

standard I/O and then reload VM, no matter how data are transferred. *Tcp* mode cannot be adopted here since TCP connection is not supported in XIA. We choose to get KVM migration run in *exec* mode.

Migration Data Transfer Three data transmission methods are provided in XIA, namely STREAM, DGRAM and CHUNK. STREAM is connection oriented and provides reliable transmission, just as TCP in TCP/IP protocol stack. Correspondingly, DGRAM is connectionless like UDP. Concept of CHUNK is widely used in content-centric future Internet architectures, especially XIA and NDN. All the data should be divided into chunks when transmitting in CHUNK mode. CID of a chunk is obtained by hash of the whole content block, so it can get self-verified. Network traffic is well controlled because each transaction is originated by the receiver and the sender just need to put the data that are required into content cache.

CHUNK mode is quite reliable because of its error control and intrinsic traffic control mechanisms though it is connectionless. We employ it as a transmission method for VM migration in XIA. When migration data are sent to standard I/O, migration sending process acquires and delivers them to the destination host. Meanwhile, receiving process accepts the chunks and writes them into standard I/O. The details of chunk mode data transmission procedure and the control protocol for its management will be introduced in Sect. 4.1.

Migration Test and Verify In order to test whether VM migration procedure is live or not, we propose to run some applications in VM. Since traditional applications cannot work efficiently in XIA network environment, a calculation application (expressed as Cal in the context) is introduced, which is developed with APIs provided in XIA. A calculation server daemon runs in VM, calculating and verifying *Goldbach conjecture* (every even number can be expressed as a sum of two prime numbers). A client process runs on client host (HOST_C in Fig. 1), acquiring answers from server and printing them onto screen.

We propose to use *Xping* provided in XIA prototype as a way for downtime evaluation. For example, if time interval of Xping packets is set as Δt and n packets are dropped during downtime, we can get informed that downtime measured is $(n \pm 1) * \Delta t$, that means the downtime is $n * \Delta t$ with deviation of Δt. This is quite accurate if value Δt is small enough.

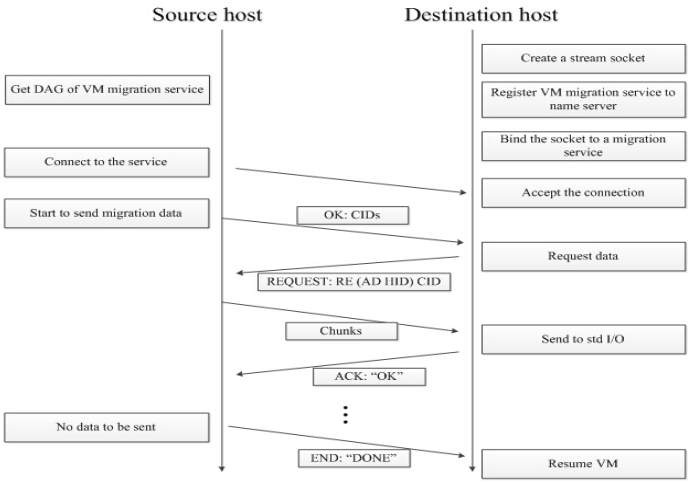


Fig. 3: Detailed working process of VM migration control protocol

4 Control Protocol and Self-adaptive Mechanism

We present a migration control protocol to manage the VM migration procedure in XIA. Since keeping the migrated VM accessible is significant, we introduce the self-adaptive mechanism for network work recovery after VM migration.

4.1 VM Migration Control Protocol

Fig. 3 shows the process of data transmission. Control messages are delivered by STREAM because it is simple and reliable. The destination site starts a stream socket first and binds it to a migration service. The socket is in listening state, waiting for connection of migration data sender host. After connection is set up, the source host (sender) will notify the destination host (receiver) of CIDs of the chunks that needed to be delivered. The destination host will construct messages to request for these chunks. The receiver then acknowledges for this round of transmission if the data are check to be correct and the sender continues its data transmission procedure till nothing to be delivered. If the data sender is sure of the end of migration, a "DONE" message will be sent out to announce the termination of VM migration.

4.2 Self-adaptive Mechanism

In implementation of VM migration in WAN of TCP/IP network, VM will get unreachable after being migrated to destination subnet. There are mainly three challenges. Firstly, the IP address obtained before migration belongs to the source subnet and it can't be recognized in new subnet, even the destination host which the VM lies on is not aware of the existence of VM. Secondly, if a

Fig. 4: An simple example of XIA address expressed in the DAG form (*src* indicates a virtual source of DAG. *AD* is a 160-bits number identifying an AD. Similarly, *HID* and *SID* are 160-bits identifiers presenting a host or a service.)

new address is acquired, e.g. by DHCP, it is also difficult to get the new address information propagated to the whole Internet. Thirdly, all the communications related to the migrated VM should be recovered and the communications set up afterward must be routed to the right location.

Therefore, we present self-adaptive mechanism after VM is resumed on destination host. In order to solve the above-mentioned problems, self-adaptive procedure mainly focuses on three aspects, namely VM mobility perception, new location notification and traffic redirection.

Migration Perception One basic precondition for VM migration accomplishment and traffic recovery is that migration of VM should be detected as soon as possible. A general solution is to intercept and capture signals from the hypervisor when particular event occurs. In XIA network, this goal can be achieved conveniently. XIA gateway router inside an AD broadcast beacons periodically which contain identifiers of the AD and router. Any host that receive the broadcast packets can easily determine which AD they belong to at present.

New Location Notification Addresses in XIA are expressed by DAG and a simple form is depicted in Fig. 4. This structure is constructed with all kinds of IDs that are necessary, so it is convenient for re-construction. AD should be changed when a VM is migrated to a new AD, and a new DAG form address should be re-registered to name service as soon as VM migration is detected. Additionally, since a router is to manage the AD it locates in, the VM should also make itself noticed by the gateway router in the destination AD and the router will append its routing table with an entry directing to the VM.

Traffic Redirection Network traffic related to VM should be resumed after it is migrated to destination host. Most of the researches that studying live VM migration in WAN adopt agents on source host and tunnels between agent and VM. Neither is needed when it comes to XIA because of its particular characteristics. Information contained in either source or destination address field of XIP layer header is not yet IP, but DAG instead. *Lastnode* field stores an identifier that indicates the node which is last processed by router. When a router receives a packet, it first checks *Lastnode* field and then processes the nodes afterward. Thus, a packet is routed and forwarded based on the information of a particular node in DAG, not DAG as a whole.

According to the analysis, we can take some modifications to the routing tables of routers on the migration path. In this way, packets forwarding to VM are still able to be delivered correctly according to "HID" routing entries, even though "AD" information is changed. We just have to change the *next-hop* field of VM's HID entry to make the path directing to the new location. If a router in source AD receives a packet with VM's obsolete DAG as destination address, it will direct the packet to next hop router which is nearer to destination host. For example, if ROUTER_1 in Fig. 1 receives a packet with VM's original DAG, this packet should be routed to ROUTER_2 according to the routing table that has been modified and finally it will arrive at VM because ROUTER_2 knows the location of VM exactly.

The rules of routing table in XIA also bring some troubles. When a VM is running on source host, a HID entry is added into it with the host's HID as destination and next hop address. Similarly, there is also an entry pointing to VM's HID in source host's routing table. This will lead to trouble as both of these entries can't get modified automatically and packets will be routed incorrectly when VM is migrated. Therefore, these two entries should be deleted in order to keep connectivity between source host and VM.

5 Implementation and Evaluation

We carry out VM migration over XIA network with the testbeds depicted in Section 3. With the evaluation results, we can easily get to know the deficiencies of our design and get improvement in next phase. We do not focus on iteration phases during migration or factors that trigger migration. Therefore, we don't need to take any modifications to QEMU-KVM. This is beneficial to make the current virtualization products flourish in the future when future networks such as XIA takes the place of TCP/IP. VM is configured with 4GB virtual disk and 640MB physical memory for full migration. The size is necessary for installation of XIA software. Hosts are configured with Intel core I3 processor and 8GB RAM. Computers with multiple network interface cards are used as XIA routers. All the physical hosts and VM are running Ubuntu 12.04 with kernel 3.5.0. Source code of XIA prototype v1.1 is obtained from Github.

5.1 Comparison of Migration Modes

First of all, it is necessary to compare migration performance in *tcp* and *exec* modes so as to get the differences between them. We implement *exec* mode data transmission by using SOCK_STREAM sockets in TCP/IP network, comparing to the default *tcp* method. Two kinds of workload along with calculation application we developed are introduced, which are widely used in network research areas.

Dbench: an open source benchmark tool to generate I/O workloads, simulating a variety of real file servers. We choose it as an I/O intensive application.

Netperf: a benchmark that can be used to measure the performance of many different types of networks. Here we run it as a workload inside VM and it does not communicate with clients because of limitations of protocol stack.

Fig. 5 shows the total migration time and downtime of VM migration in two modes with different VM workload. We can get concluded from comparison that total migration time and downtime will increase obviously if we use *exec* mode for migration with same workload, especially downtime. As a self-defined transmission method, the throughput of *exec* mode transmission is slightly lower than that of *tcp*. Thus, migration method selection affects the performance and we will complete *exec* mode VM migration over XIA for a comparison to that over TCP/IP network.

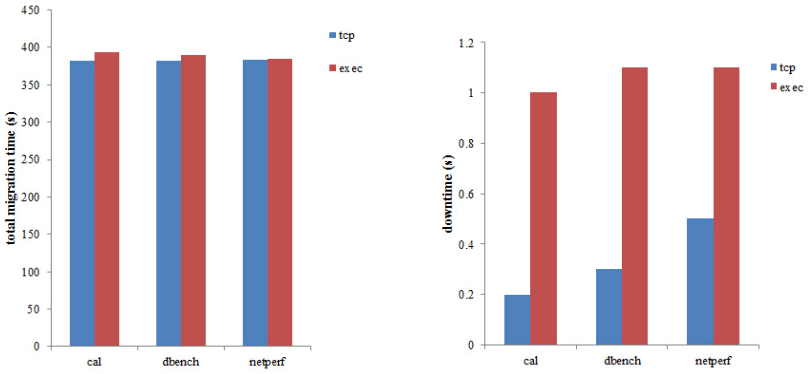


Fig. 5: Total migration time and downtime of two migration mode (*tcp* and *exec*)

5.2 Connectivity Test

We test VM migration in exec mode over both TCP/IP and XIA networks. Processing programs is required for data sending and receiving with sockets provided in TCP/IP and XIA networks respectively. A calculation service always runs in the VM for connectivity test. We just take one application as an example because VM will be unreachable in WAN as a matter of experience, no matter which kind of workload it takes along.

First of all, the service runs in VM never get interrupted during the migration procedure. Xping drops several packets during downtime but recovers soon after VM's resuming on destination host. Both connection-oriented and connectionless services will not be interrupted during VM migration, even without agents or tunnels used for network recovery in WAN of TCP/IP network.

Total migration time and downtime of above experiments are demonstrated in Table 1. We can conclude that full VM migration in LAN of TCP/IP network takes the least total migration time and downtime. When it comes to WAN,

Table 1: Performance of VM migration in different networks

	Intra-AD		**Inter-AD**	
	total time	downtime	total time	downtime
TCP/IP	6.5 min	1 s	-	-
XIA	14 min	1.2 s	15 min	1.2 s

Table 2: Migration performance in XIA network with different workload

	Intra-AD		**Inter-AD**	
	total time	downtime	total time	downtime
Calculation	14 min	1.2 s	15 min	1.2 s
Dbench	12 min	1.0 s	13 min	1.5 s
Netperf	12 min	1.7 s	14 min	1.7 s

the VM and its services are all inaccessible after it has been migrated to the destination subnet. The migrated VM has kept the original IP address and this can't be recognized in a different subnet.

In XIA network, VM migration can be achieved successfully though the performance isn't so good. Downtime is about 0.2s longer than that in TCP/IP while total migration time is about twice longer. The long time is probably caused by chunk cache mechanism in XIA routers. Chunks that are passing through a router would be cached for future use. The router will search its cache when a chunk request comes and it will deliver this chunk to client if found, or it will continues to forward this request. Thus the time cost for chunk search will sharply increase chunk transmission time. It can also reduce network throughput to some extent. Lots of effort has to be made for performance optimization in XIA. We have made some modifications to the cache algorithms, that is, to release the first chunk in the cache when extra space is required, which reduces total migration time and downtime sharply.

5.3 Workload Test

We evaluate VM migration performance in XIA with different workloads, representing typical server applications in today's data centers. Experimental results are shown in Table 2.

Calculation is CPU-intensive and network-intensive because it calculates results and delivers the data rapidly. Both *netperf* and *dbench* can be regarded as I/O intensive. We can see in Fig. 6 that total migration time varies little between the migrations in intra-AD and inter-AD. In contrast, migration of VM with *dbench* has the largest downtime variation. Among the applications listed, VM running *netperf* and *dbench* suffers longer downtime while the calculation application has longer total time. We can draw a conclusion that network intensive services will suffer longer total migration time, while downtime is longer for I/O intensive services.

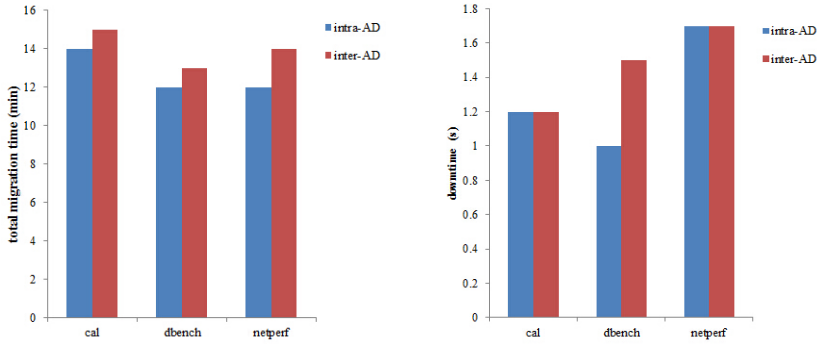


Fig. 6: Total migration time and downtime of VM migration in XIA network with different workload test

6 Future Works

This paper takes the first step towards the research of VM migration over XIA and our future studies will base on the testbeds designed. There are still many open issues that need to be further explored.

First of all, though we have achieved VM migration over XIA, the performance isn't so ideal. Performance optimization is an urgent matter. We hope to reduce total migration time and downtime in XIA to be as short as in TCP/IP. Another research point is migration strategies such as load balancing. Besides, we can study VM migration in more future Internet architectures in order to find one that is most suitable for future demand.

7 Conclusion

This paper designs experimental testbeds in future Internet prototypes comparing against VM migration in LAN and WAN of TCP/IP network. We propose KVM virtual machines to be migrated with independent sending and receiving programs. Data are transmitted in chunks and this procedure is managed by a migration control protocol. We then introduce a self-adaptive mechanism in order to solve the application interruption problem after VM is migrated, especially among ADs. All the traffics directed to VM can be recovered even if they are still using original VM addresses. Evaluation results show that VMs can be migrated in XIA networks successfully with downtime in the acceptable range. Performance improvement can leave for future research.

Acknowledgments The authors are sincerely grateful for the technical support from Prof. Peter Steenkiste and Mr. Dan Barrett in Carnegie Mellon University and financial support from Shanghai INGEEK Information Technology Co. Ltd.

References

1. Han, D., Anand, A., Dogar, F., et al.: XIA: Efficient Support for Evolvable Internetworking. In: 9th NSDI, USENIX Association, Berkeley (2012)
2. Anand, A., Dogar, F., Han, D., et al.: XIA: An architecture for an evolvable and trustworthy Internet. In: Proceedings of the 10th ACM Workshop on Hot Topics in Networks, Article No. 2. ACM, New York (2011)
3. Kang, T.S., Tsugawa. M., Fortes, J., et al.: Reducing the Migration Times of Multiple VMs on WANs Using a Feedback Controller. In: IPDPSW, IEEE, Piscataway (2013) 1480-1489
4. Al-Kiswany, S., Subhraveti, D., Sarkar, P., et al.: VMFlock: virtual machine co-migration for the cloud. In: 20th international symposium on High performance distributed computing, ACM, New York (2011) 159-170
5. Comer, D.: A future Internet architecture that supports Cloud Computing. In: 6th International Conference on Future Internet Technologies, ACM, New York (2011) 79-83
6. Ibrahim, K.Z., Hofmeyr, S., Iancu, C. et al.: Optimized pre-copy live migration for memory intensive applications. In: International Conference for High Performance Computing, Networking, Storage and Analysis, ACM, New York (2011) 1-11.
7. Michael, R.H., Umesh, D., Kartik, G.: Post-copy live migration of virtual machines. In: ACM SIGOPS Operating Systems Review, vol. 43, iss. 3. ACM, New York (2009) 14-26
8. Luo, Y.W., Zhang, B.B., Wang, X.L., et al: Live and incremental whole-system migration of virtual machines using block-bitmap. In: IEEE International Conference on Cluster Computing, IEEE, Piscataway (2008) 99-106
9. Liu, H.K., Jin, H., Liao, X.F., et al: Live migration of virtual machine based on full system trace and replay. In: High performance distributed computing, ACM, New York (2009) 101-110
10. Fei, M., Feng, L., Zhen, L.: Live virtual machine migration based on improved pre-copy approach. In: Software Engineering and Service Sciences (ICSESS), IEEE, Piscataway (2011) 230-233
11. Bradford, R., Kotsovinos, E., Feldmann, A., et al: Live Wide-Area Migration of Virtual Machines Including Local Persistent State. In: 3rd international conference on Virtual execution environments, ACM, New York (2007) 169-179
12. Silvera, E., Sharaby, G., Lorenz, D., et al.: IP Mobility to Support Live Migration of Virtual Machines across Subnets. In: SYSTOR'09, Article No. 13. (2009)
13. Harney, E., Goasguen, S., Martin, J., et al: The Efficacy of Live Virtual Machine Migrations over the Internet. In: 2nd international workshop on Virtualization technology in distributed computing, ACM, New York (2007) 8-14
14. Wood, T., Ramakrishnan, K.K., Shenoy, P., et al.: CloudNet: Dynamic Pooling of Cloud Resources by Live WAN Migration of Virtual Machines. In: 7th VEE, ACM, New York (2011) 121-132
15. Tsugawa, M., Riteau, P., Matsunaga, A.: User-level Virtual Networking Mechanisms to Support Virtual Machine Migration over Multiple Clouds. In: GLOBECOM Workshops, IEEE, Piscataway (2010) 568-572
16. Kohler E, Morris R, Chen B, et al: The Click modular router. In: ACM Transactions on Computer Systems, vol. 18, iss. 3, ACM, New York (2000) 263-297