Live Migration of Virtualized Carrier Grade SIP Server

Tadasuke Nozoe, Masahiko Noguchi, Minoru Sakuma and Mikio Isawa

NTT Network Service Systems Laboratories, Nippon Telegraph and Telephone Corporation, Japan
t.nozoe@gmail.com, noguchi.masahiko@lab.ntt.co.jp, sakuma.minoru@lab.ntt.co.jp, isawa.mikio@lab.ntt.co.jp

***Abstract*:** The concept of network virtualization, such as network functions virtualization, has attracted considerable attention from telecom carriers and a live migration technique is a key feature of virtualization technology. However, there are some challenges associated with applying server virtualization technology including live migration to a SIP server. Previous work has not dealt with the performance or behavior of a SIP server during live migration. Neither has it targeted a carrier grade SIP server for live migration. In this paper we present a virtualized carrier grade SIP server running on a virtual machine, which is configured with Carrier Grade Linux, HA middleware and SIP-AS application. We also assess its performance to investigate the impact of throughput degradation and suspension on a SIP layer and HA cluster configuration.

***Keywords*:** IMS, Live Migration, SIP, Virtualization.

1. Introduction

For the past decade, telecom carriers worldwide have been migrating from a PSTN (Public Switched Telephone Network) to an all-IP NGN (Next Generation Network) based on the 3GPP IMS (IP Multimedia Subsystem) architecture to introduce new services more quickly and reduce CAPEX and OPEX [1]. IMS uses the Session Initiation Protocol (SIP) [2] as a signaling protocol, and SIP servers in IMS such as CSCF (Call Session Control Function) and SIP-AS (Application Server) provide multimedia services such as audio, video, text, and data. In addition to session control, CSCF, which is divided into S-CSCF (Serving-CSCF), P-CSCF (Proxy-CSCF) and I-CSCF (Interrogating-CSCF), also performs various functions such as SIP message routing, user registration, policy control, and invoking SIP-AS. On the other hand, SIP-AS only controls call sessions and executes services based on SIP messages transferred from CSCF.

Each SIP server in IMS needs to both reduce CAPEX and meet requirements for carrier grade high availability which is comparable to a PSTN switch. To meet these needs, each server adopts a standardized hardware platform, OS and middleware for telecom systems. More specifically, the ATCA (Advanced Telecom Computing Architecture) hardware platform, CGL (Carrier Grade Linux) and HA (High Availability) middleware have been used in IMS, and which are standardized by the PICMG (PCI Industrial Computer Manufacturers Group), The Linux Foundation and the SAF (Service Availability Forum), respectively. Each SIP server in IMS is configured in a cluster to provide high availability and service continuity by using the AMF (Availability Management Framework), which is one of the APIs specified by the SAF for HA cluster management.

Recently, the concept of network virtualization, such as NFV (Network Functions Virtualization), has attracted considerable attention from telecom carriers because of advances in server virtualization technology and the improved performance of IA servers. Network virtualization can contribute to EoL on ATCA hardware platforms, further reducing CAPEX, OPEX and service introduction times, and speeding recovery from disasters and so on. In fact, there are some challenges in applying server virtualization technology to IMS servers including CSCF and SIP-AS. In a virtualized environment, each IMS server runs on a virtual machine (VM) created and managed by a virtual machine monitor (VMM) installed on a physical machine (PM).

Live migration is a key feature of server virtualization technology. This is technique by which a VM is moved from one PM to another with near zero downtime [3]. Live migration works by iteratively copying a VM memory and can be used for system maintenance interruption, load balancing or saving energy. It is, however, reported that the throughput of running a VM decreases during migration due to the overhead of migration processes on a PM [3][4]. In addition, operation of a VM also needs to be suspended to finish the live migration process, the duration of which is close to but not zero. It is therefore important to investigate the impact of throughput degradation and suspension of operation on the behavior of the SIP layer and a HA cluster configuration in order to apply a live migration technique to a carrier grade SIP server requiring high availability.

Our goal in this paper is to evaluate how a virtualized carrier grade SIP server, which is HA clustered using both CGL and HAM and run on a VM on an IA server, behaves during live migration. A SIP-AS is used for evaluation in this paper because a SIP-AS has fewer functions than CSCF and it is easier to analyze the behavior during live migration.

The rest of this paper is organized as follows. In section 2, we briefly discuss related work on server virtualization and live migration for a SIP server. In section 3, a virtualized carrier grade SIP server and experimental scenarios are presented for estimation. Section 4 describes and discusses experimental results. In Section 5, we conclude the paper.

2. Related Work

There are some studies on applying the server virtualization technique to an IMS entity or a SIP server. Virtualization of IMS is described as one of the use cases for NFV.

Yang et al. focused on HSS (Home Subscriber Server), which stores user profile information in the IMS, and proposed a cloud-based HSS architecture to improve the performance of the IMS [5]. They only dealt with HSS and not a SIP server such as CSCF and SIP-AS in IMS. Bellavista et al. developed a Cloud Broker Engine able to dynamically up/down-scale cloud resources across multiple cloud platforms [6]. ASs are implemented in a cloud provider platform and KVM is employed as a VMM. Corte et al. designed a virtualized IMS testbed which contains virtualized P-CSCF, S-CSCF, HSS and UE (User Equipment) based on VMWare [7]. Carella et al. implemented an IMS testbed comprising P-CSCF, I-CSCF, S-CSCF, HSS and DNS on top of an OpenStack cloud and proposed an efficient IMS-as-a-Service architecture [8]. Segec et al. built a laboratory prototype environment [9]. They used Xen as the VMM and open source components such as Kamailio and the Sailfin SIP servlet server. Roly et al. designed a SIP server virtualized by KVM to build a testbed for overload control in SIP networks [10]. Chen et al. evaluated the performance of Asterisk, which is open source software, to implement an IP-PBX system, virtualized on Xen [11]. Lee et al. designed an experimental platform which consists of a media server, a SIP server and a call control server virtualized by Xen to estimate their proposed hypervisor’s scheduler [12]. Voznak et al. used Asterisk virtualized by KVM, Virtual Box and VMware to measure performance such as the delay between sending an INVITE request and receiving a response [13]. None of these studies, however, dealt with live migration.

Fischer et al. proposed a method for wide-area migration of VMs, which is VM migration across different subnets, to improve SIP network resilience [14]. The proposed method uses SIP-based end-to-end notification to redirect SIP requests after performing a wide-area migration, so it is not live migration of VMs. Fakhfakh et al. proposed a live migration solution for overload of virtualized P-CSCF [15]. Under the condition that some P-CSCFs run on a VM created by Xen on the same PM, when one of the P-CSCFs receives a larger number of SIP requests than others, this P-CSCF is live migrated to another PM for load balancing. The session setup delay is measured both before and after the live migration, however, this is not done during live migration. Neither do they refer to the behavior of P-CSCF during live migration. Nkubito et al. investigated resource overheads and performance degradation of a virtualized Astarisk server on Xen VMM, such as migration time, dirty page rate, and CPU usage of PMs, during live migration [16]. They did not, however, investigate the behavior of a SIP server or SIP layer during live migration. Femminella et al. implemented Java-based SIP-AS on ESXi VMM and performed live migration using vMothion, however, they did not provide a detailed description of their results [17].

As mentioned above, there is no previous study on the performance or behavior of a SIP server during live migration. Neither has previous work dealt with a carrier grade SIP server as a target for live migration.

3. Experimental Setup

3.1 Virtualized SIP server

We evaluate the performance and behavior of a virtualized carrier grade SIP server during live migration. An overview of the virtualized carrier grade SIP server for evaluation is depicted in Figure 1.

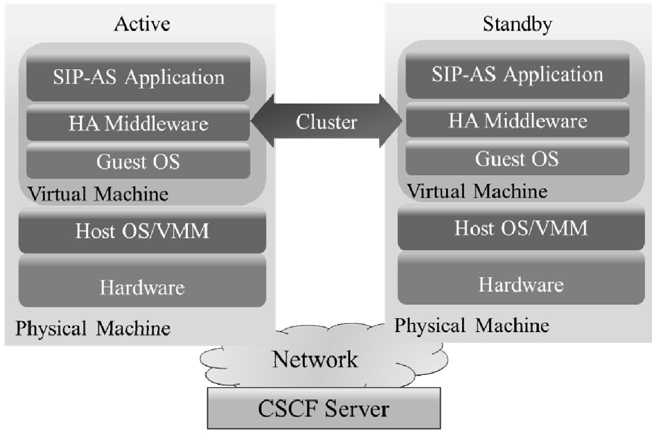


Figure 1. Overview of virtualized carrier grade SIP server

The details of each component of the server are provided below:

3.1.1 Physical machine

Our experiments are conducted on a Dell PowerEdge M610 machine with two Intel Xeon E5502 1.87GHz 2-core sockets, 32GB of RAM, and 146GB of SCSI hard disk. It also has 2 onboard and 4 additional 1Gbps network cards, which configure NIC bonding, and are used for SIP, cluster and management communication, respectively.

3.1.2 Host OS and VMM

The host OS is a Linux 2.6.32-431.el6.x86_64 kernel in which KVM is used as a VMM and the versions of qemu-kvm and libvirt are 0.12.1.2-2.415.el6.x86_64 and 0.10.2-29.el6.x86_64, respectively.

3.1.3 Virtual machine

The virtual machines are configured with 4 vCPUs, 16GB of RAM, 100GB of virtual IDE disk and 6 virtual e1000 network cards which are one-to-one connected to physical network cards. The VM disk images are stored on a NFS shared file system on the physical host.

3.1.4 Guest OS

We use CGL based on a Linux 2.6.18-53 kernel as a guest OS on the VM. The CGL has carrier grade functions and mechanisms, such as live patching that enables Linux to patch a process, without having to shut down the process, boot cycle detection and a boot image fallback mechanism whenever a serious error occurs, and a fast online process dump mechanism.

3.1.5 HA middleware

The carrier grade SIP server is clustered in an active/standby configuration with the SAF-based HA middleware consisting of multiple area servers such as AMF (Availability Management Framework), CLM (Cluster Membership Service), CKPT (Checkpoint Service), and MSG (Message Service). The in-memory database is also implemented as a SAF component [18].

AMF provides service availability by coordinating other area servers and components such as the in-memory database or the SIP-AS application within the cluster. AMF provides component registration, lifecycle management, error reporting and health monitoring, and also assigns active or standby workloads to the components. CLM manages a cluster membership and decides whether a configured server is transitioned to be a member VM of the cluster. CKPT provides a facility for processes to record checkpoint data incrementally to protect an application against failures. When failover or switchover occurs, the checkpoint data can be retrieved, and the application can resume from the state recorded before the failure. MSG is a buffered message-passing system based on the concept of a message queue for processes on the same or different VMs. The in-memory database uses MSG for inter-server communication.

3.1.6 SIP-AS application

The SIP-AS application is a B2BUA SIP server and provides a number translation service in this experiment. The SIP-AS application translates a non-geographic number into a hidden geographic number by means of user profile information in the in-memory database after receiving an initial INVITE request from the CSCF server. After that, it sends an initial INVITE with the hidden geographic number to the CSCF server. In this experiment, all SIP messages from the initial INVITE request to the 200 OK response to the BYE request are routed via the SIP-AS.

In addition to session controlling and number translation, the SIP-AS application has the functions of traffic information gathering, log output, overload control and so on. The overload control is a critical function for a PSTN/ISDN switching system [19] and a NGN SIP server [20]. The overload control of the SIP-AS application monitors the CPU utilization of the VM and regulates receiving request messages autonomously to prevent serious throughput degradation. It consists of two regulation phases according to the CPU utilization of the VM. When the CPU utilization of the VM exceeds 70% for 20 consecutive seconds, phase 1 regulation is started. The SIP-AS application processes only 20 initial INVITE requests per second under the phase 1 regulation. If the SIP-AS application receives more than 20 initial INVITE messages under the phase 1 regulation, it sends a 503 response to the initial-INVITE request to the CSCF server. In the case of 80 % CPU utilization of the VM for 20 consecutive seconds, the SIP-AS application starts phase 2 regulation under which it refuses to process all initial INVITE messages and sends a 503 response. When the CPU utilization of the VM is less than 65% and 75% for 40 consecutive seconds, phase 1 and phase 2 regulations are stopped. The requests, other than the initial INVITE, Re-INVITE, UPDATE, ACK, BYE etc., are not regulated in this evaluation.

3.1.7 CSCF server

The CSCF server performs the role of a P-CSCF and S-CSCF. It forwards both request and response messages from UEs to the SIP-AS. It also routes messages from the SIP-AS to the appropriate UE. It is connected to the SIP-AS over TCP and all SIP messages are forwarded over TCP.

3.2 Experimental scenarios

Live migration as performed in this experiment is depicted in Figure 2. Either the active VM or the standby VM, which are clustered, is migrated from physical machine 1 to physical machine 2 for evaluation. Physical machines 1 and 2 have the same specs as described in this section. The live migration process is initiated 120 seconds after starting measurement.

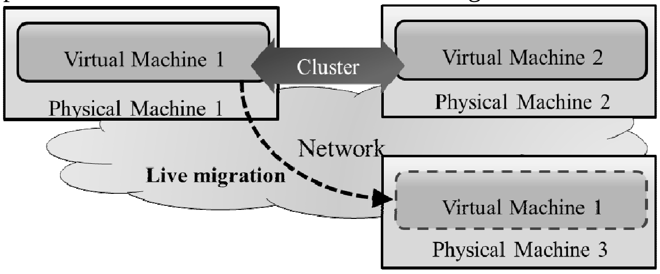


Figure 2. Diagrammatic representation of live migration

The SIP sequence between the SIP-AS and the CSCF server is shown in Figure 3. A PRACK request as defined in [21] and an UPDATE request as defined in [22] are used in the sequence. The UPDATE request is used for session refresh. The called UE sends a 200 OK response to the initial INVITE to the calling UE 10 seconds after it receives a 200 OK response to the PRACK request. It also sends a BYE request to the calling UE 180 seconds after receiving the ACK request.

We evaluate the status of cluster configuration, the behavior of the SIP-AS application, CPU utilization of both the VM and PM, and incomplete calls before, after and during live migration in this experiment. The incomplete call is one whose execution sequence from session establishment to termination has been not completed normally. There are, for example, cases when an error response, such as 4XX or 5XX, is received in response to an initial INVITE request or when a dialog is terminated by a BYE request due to no response being received to an UPDATE request.

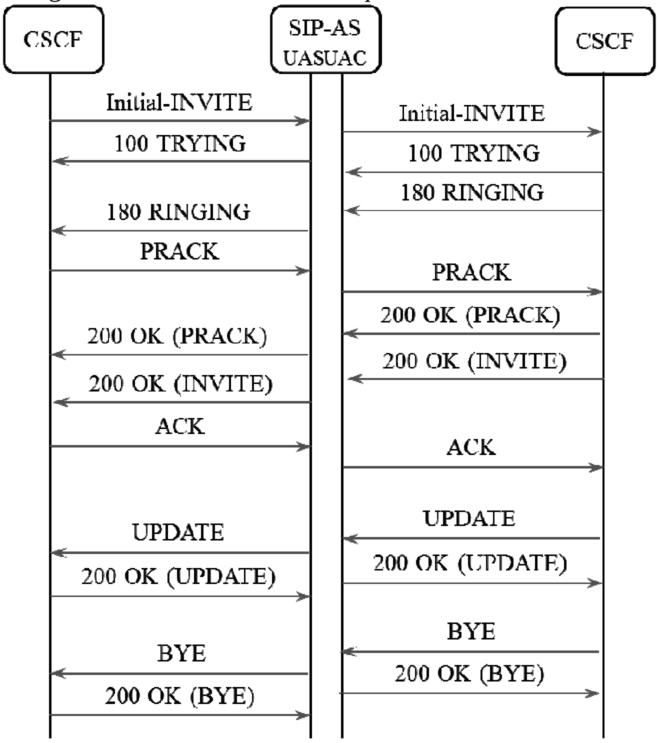


Figure 3. SIP sequence

4. Experimental Results

We performed measurements to investigate the performance and behavior of the SIP-AS on the VM during live migration. Figure 4 shows the average CPU utilization of both the active and standby VM before and during live migration under different calls per second (cps). The curve of average CPU utilization of the active VM during live migration follows a linear behavior until 15 cps and remains about 60% after 20 cps, which is less than the regulation threshold described in section 3.

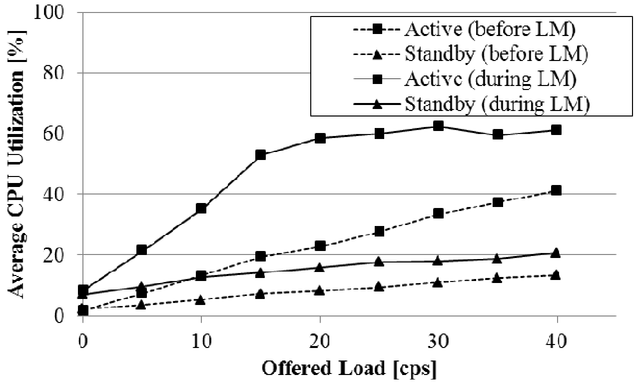


Figure 4. Average CPU utilization of VM during live migration

The active line before live migration, on the other hand, shows a linear increase with an offered load until 40 cps. The average CPU utilization of the standby VM before and during VM migration increases with offered load until 40 cps as well as before active VM live migration. The CPU utilization of the standby VM before and during live migration increases at the same rate, on the other hand, the increasing rate of active VM CPU utilization during live migration is higher than before live migration. The difference in CPU utilization between before and during live migration indicates that overhead caused the live migration process to begin on the PM. In the case of standby VM live migration, the overhead of live migration is constant. The overhead of active VM live migration, on the other hand, increases with offered load until 15 cps.

We observed about 50 incomplete calls, 408 Request Timeout responses to the initial INVITE request from SIP-AS to calling UEs during active VM live migration after 20 cps and standby VM live migration after 35 cps. The active and standby configuration of VMs was not affected by live migration regardless of offered load. A 408 response is sent to calling UEs when Timer A, one of the SIP timers defined in [2], expires. The SIP-AS cannot process receiving SIP messages quickly due to the overhead of live migration. After receiving initial INVITE requests from calling UEs via the CSCF server, the SIP-AS sends the initial INVITE requests to called UEs via the CSCF server and called UEs send 180 responses. While the SIP-AS receives them, it cannot however process them properly because it cannot use its CPU resources fully due to the live migration overhead.

The migration time is shown in Figure 5. The live migration time of the active VM and offered load are proportional until 25 cps and the live migration time remains almost flat after 35 cps. The standby VM's migration time also increases with offered load because the memory of the standby VM is synchronized with one of the active VMs by the CKPT function.

Figure 6 and Figure 7 show the CPU utilization of the VM and PM with no offered load. In Figure 6, VM1 is active and VM2 is on standby. VM1 is migrated from PM1 to PM3 and VM2 is on PM2. The live migration process began after an elapse of 120 seconds and finished 60.0 seconds later. The CPU utilization of PM1 increases by about 50% and one of the VM1s also increases CPU utilization by about 6% during live migration regardless of there being no offered load, which means that this is the live migration overhead. In Figure 7, VM1, being the standby VM, is migrated from PM1 to PM2. The live migration time was 58.9 seconds. As with active VM live migration, the PM1 and VM1 CPU utilization increases during live migration.

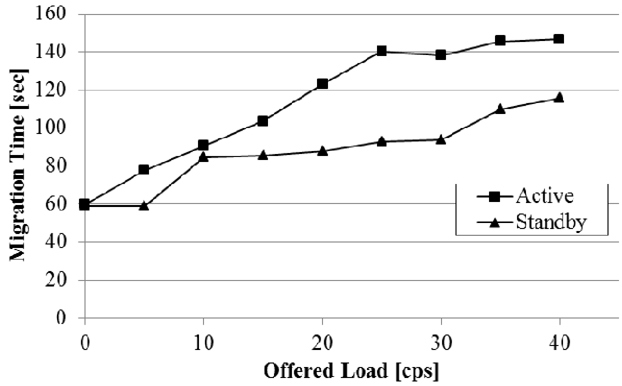
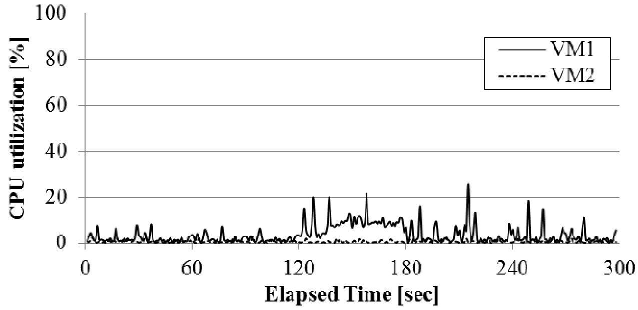
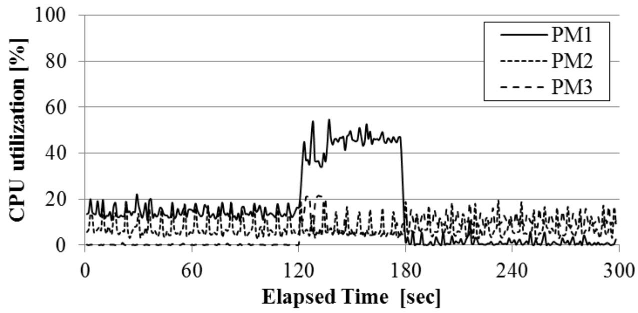


Figure 5. Migration time

The CPU utilization of the VM and PM with 15 cps is shown in Figure 8 and Figure 9. VM1 is active in Figure 8 and on standby in Figure 9. As with no offered load, the CPU utilization of VM1 increases during live migration and becomes normal again after finishing migration in both Figure 8 and Figure 9.

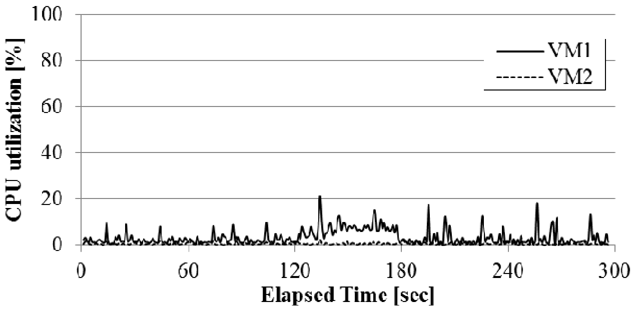


(a) Virtual machines

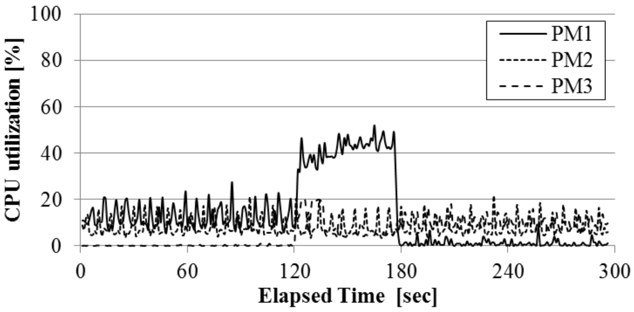


(b) Physical machines

Figure 6. CPU utilization during active VM live migration with no offered load

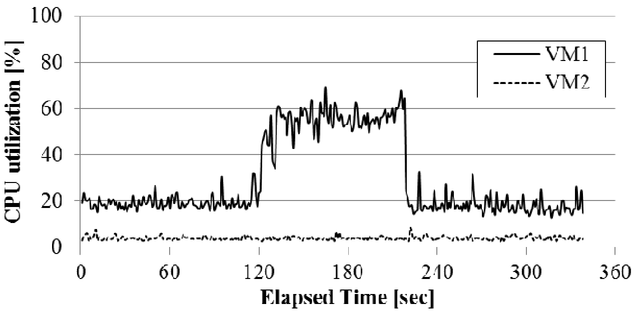


(a) Virtual machines

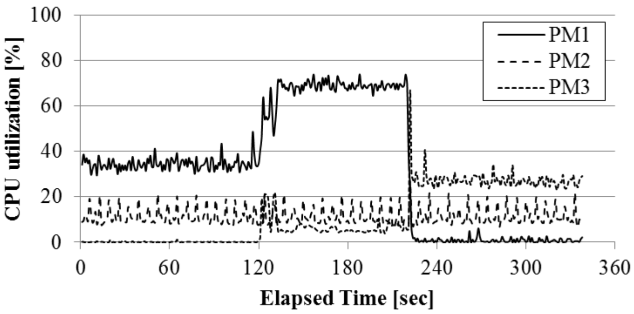


(b) Physical machines

Figure 7. CPU utilization during standby VM live migration with no offered load

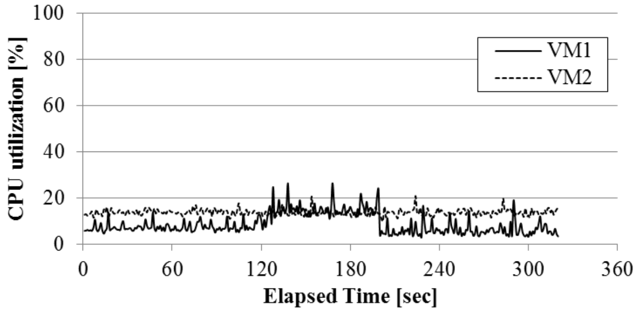


(a) Virtual machines

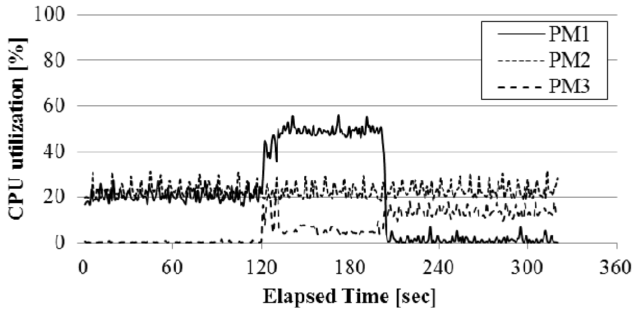


(b) Physical machines

Figure 8. CPU utilization during active VM live migration with 15 cps

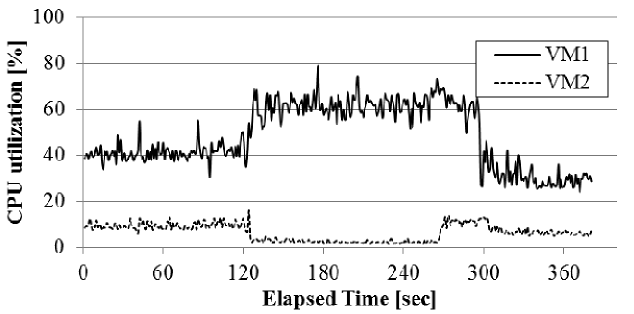


(a) Virtual machines

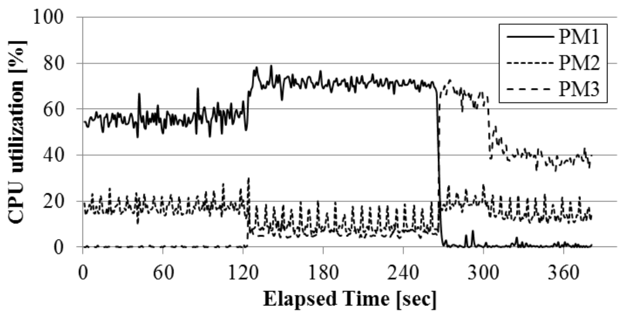


(b) Physical machines

Figure 9. CPU utilization of standby VM live migration with 15 cps



(a) Virtual machines

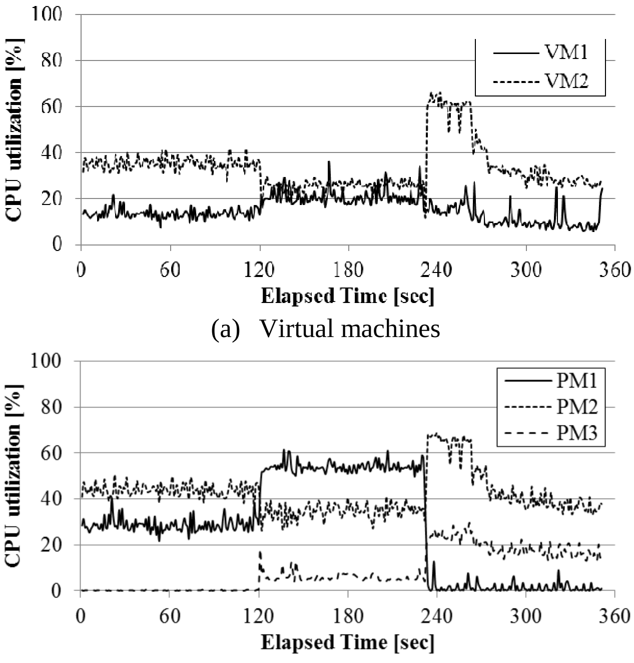


(b) Physical machines

Figure 10. CPU utilization during active VM live migration with 40 cps

The CPU utilization of PMs shows similar behavior with no offered load. The migration times in Figure 8 and Figure 9 are 103.7 and 85.3 seconds respectively.

Figure 10 and Figure 11 show CPU utilization with 40 cps, under which incomplete calls are observed during live migration. In Figure 10, the active VM is migrated and its live migration time is 146.7 seconds. Unlike the case of no offered load and 15 cps, the increase in CPU utilization of VM1 and PM3 continues for 30 seconds after finishing the live migration process. In addition, the CPU utilization of VM2, which is the standby VM, decreases during live migration and it becomes normal again immediately after migration. This behavior cannot be observed in Figure 8 and Figure 9. This result means that the throughput of the active VM decreases during live migration and the amount of synchronizing memory needed for the standby VM decreases. The throughput degradation of the active VM during live migration causes retransmission of SIP messages from UEs. This results in the active VM processing a large number of SIP messages, which includes retransmitted messages from UEs, after live migration. This processing increases the CPU utilization after live migration. In Figure 11, the CPU utilization of VM1, the standby VM, increases and the CPU utilization of VM2, the active VM, decreases during VM1



(a) Virtual machines

(b) Physical machines

Figure 11. CPU utilization during standby VM live migration with 40 cps

migration. The CPU utilization of VM2, in addition, rises sharply after finishing the live migration process and continues for about 30 seconds. The performance of the standby VM being migrated degrades due to the overhead of live migration and the active VM cannot fully process memory synchronization with the standby VM. The active VM tries to synchronize memory with the standby VM in the event of receiving or sending SIP messages, however, it takes time to complete the process because of performance degradation of the standby VM. As a result, the throughput of the active VM decreases during live migration, which causes retransmission of SIP messages from UEs as was seen in the case of active VM migration in Figure 10. The active VM has to process a large number of SIP messages after live migration due to such retransmission.

5. Conclusion

In this paper we have presented a virtualized carrier grade SIP server which is configured in a HA cluster based on SAF to evaluate its performance during live migration. We have shown that CPU utilization of the virtualized SIP server increases during live migration. In this experiment, we set the first threshold of regulation at 70% for overload control and the CPU utilization of an active VM during live migration did not exceed the first threshold for 20 consecutive seconds. Nonetheless, we observed incomplete calls under conditions of more than 20 cps, which means that the existing overload control based on CPU utilization described in section 3 is not effective in the case where a live migration technique is used. This experiment also showed that there are incomplete calls when the standby VM is migrated in the case where 35 cps is exceeded. The active VM cannot fully process memory synchronization with the standby VM due to performance degradation of the standby VM.

HA cluster configuration was not affected by performance degradation and suspension of the VM in both active and standby live migration. This result means that an effective approach is to use SAF-based HA middleware to ensure the high availability of a virtualized carrier grade SIP server. Continuous real-time live migration is another technique used to ensure high availability in a virtualized environment. For example, in [23], Xen with Remus is used for a high availability SIP server which consists of a primary and backup PM. If the primary PM fails, the SIP server on the PM is transparently migrated to the backup PM. This approach, however, can only cope with hardware failure and cannot address software failure because software failure is also copied to another host by the live migration process. Accordingly, it is useful that HA middleware runs on a VM to ensure high availability.

Compared with other protocols, a SIP protocol has a retransmission mechanism. It can ensure SIP-based service continuity under the condition of throughput degradation during live migration and suspension of the VM. However, this results in a large amount of SIP traffic being sent to the SIP server. As shown in Figure 10 and Figure 11, these retransmission messages increase the CPU utilization of the active VM after finishing live migration. Thus, an operator needs to be aware of the status of the SIP server not only during migration but also after migration, especially under high offered load.

We used CGL based on a Linux 2.6.18-53 kernel as a guest OS in this experiment, which was not a virtio-compatible guest kernel. In our future work, we would like to develop a virtio-compatible CGL and measure performance and behavior of the carrier grade SIP-AS running on it during live migration. We also would like to investigate other IMS entities. A media server handles media streams in addition to SIP messages. So we will measure delay, jitter, packet loss and MOS as performance metrics such as in [24] during live migration. We plan to combine live migration technology with overload control [20], capacity dimensioning [25] and so on.

References

[1] T. Murakami, “The NGN - a carrier-grade IP convergence network,” 12th IEEE/IFIP Network Operations and Management Symposium Workshops, Osaka, pp.67-73, 2010.

[2] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, E. Schooler, “SIP: session initiation protocol,” IETF RFC 3261, 2002.

[3] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, A. Warfield, “Live migration of virtual machines,” 2nd Conference on Symposium on Networked Systems Design and Implementation, Boston, Vol.2, pp.273-286, 2005.

[4] Y. Kuno, K. Nii, S. Yamaguchi, “A study on performance of processes in migrating virtual machines,” 10th International Symposium on Autonomous Decentralized Systems, Tokyo and Hiroshima, pp.567-572, 2011.

[5] T. Yang, X. Wen, Y. Sun, Z. Zhao, Y. Wang, “A new architecture of HSS based on cloud computing,” 13th IEEE International Conference on Communication Technology, Jinan, pp.526-530, 2011.

[6] P. Bellavista, G. Carella, L. Foschini, T. Magedanz, F.Schreiner K. Campowsky, “QoS-aware elastic cloud brokering for IMS infrastructures,” 7th IEEE Symposium on Computers and Communications, Cappadocia, pp.157-160, 2012.

[7] G. D. Corte, M. D. Mauro, A. Robustelli and M. Longo, “An IMS-based virtualized architecture: performance analysis,” 11th WSEAS International Conference on Mathematical

Methods and Computational Techniques in Electrical Engineering, Timisoara, pp.311-316, 2009

[8] G. Carella, M. Corici, P. Crosta, P. Comi, T. M. Bohnert, A. A. Corici, D. Vingarzan, T. Magedanz, "Cloudified IP multimedia subsystem (IMS) for network function virtualization (NFV)-based architectures," 19th IEEE Symposium on Computers and Communication, Madeira, pp.1-6, 2014.

[9] P. Segec, T. Kovacikova, "A survey of open source products for building a SIP communication platform," Advances in Multimedia, Vol. 2011, Article ID 372591, 21 pages, 2011.

[10] A. Roly, L. Schumacher, "SIP overload control testbed: Design, building and validation tests," 2nd IEEE Consumer Communications and Networking Conference Research Student Workshop, Las Vegas, pp.228-230, 2011.

[11] Z. Chen, D. Kaeli, K. Murphy, "Performance evaluation of virtual appliances," 1st International Workshop on Virtualization Performance: Analysis, Characterization, and Tools, Austin, 2008.

[12] M. Lee, A. S. Krishnakumar, P. Krishnan, N. Singh, "Supporting soft real-time tasks in the xen hypervisor," 6th ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments, Pittsburgh, Vol. 45, No.7, pp. 97-108, 2010.

[13] M. Voznak, J. Slachta, J. Rozhon, "Performance analysis of virtualized real-time applications," International Journal of Mathematical Models and Methods in Applied Sciences, Vol.2, pp.305-313, 2012.

[14] A. Fischer, A. Fessi, G. Carle, "Wide-area virtual machine migration as resilience mechanism," 30th IEEE Symposium on on Reliable Distributed Systems Workshops, Madrid, pp. 72-77, 2011.

[15] M. Fakhfakh, O. Cherkaoui, I. L. Bedhiaf, "High availability in IMS virtualized network," 1st International Conference on Communications and Networking, Hammamet, pp.1-6, 2009.

[16] P. Nkubito, A. Murgu, "Live migration of VoIP services in virtualized aervers," Southern Africa Telecommunication Networks and Applications Conference, George, 2012.

[17] M. Femminella, E. Maccherani, "Performance management of Java-based SIP application servers," IFIP/IEEE International Symposium on Integrated Network Management, pp.493-500, Dublin, 2011.

[18] T. Yoshida, M. Hisada, S. Tomita, "Implementation of highly available memory database as SAF component," 4th International Service Availability Symposium, Durham, Lecture Notes in Computer Science, Vol.4526, pp.43-51, 2007.

[19] K. Mase, H. Yamamoto, "Advanced traffic control methods for network management," IEEE Communication Magazine, Vol.28, No.10, pp.82-88, 1990.

[20] H. Yamada, T. Kawata, "Notification about congestion information through SIP session for call congestion control of VoIP application," IEEE Military Communications Conference, Orlando, pp.1-6, 2007.

[21] J. Rosenberg, H. Schulzrinne, "Reliability of provisional eesponses in the session initiation protocol (SIP)," IETF RFC 3262, 2002.

[22] J. Rosenberg, "The session initiation protocol (SIP) UPDATE method," IETF RFC 3311, 2002.

[23] D. Patnaik, A. Bijlani, V. K. Singh, "Towards high-availability for IP telephony using virtual machines," 4th IEEE International Conference on Internet Multimedia Services Architecture and Application, Bangalore, pp.1-6, 2010.

[24] X. Che and L. J. Cobley, "VoIP performance over different interior gateway protocols," International Journal of Communication Networks and Information Security, Vol.1, No.1, pp.34-41, 2009.

[25] A. Saeed, A. A. Khan, S. I. Shah, S. Yasin, "Capacity dimensioning of HSDPA urban network," International Journal of Communication Networks and Information Security, Vol.6, No.2, pp.148-161, 2014.