A Framework for Secure Live Migration of Virtual Machines

Anala M R
Department of Computer Science & Engineering, RVCE, Bangalore, India
anala_m_r@yahoo.co.in

Jyoti Shetty
Department of Computer Science & Engineering, RVCE, Bangalore, India
sjyothi.12@gmail.com

Shobha G
Department of Computer Science & Engineering, RVCE, Bangalore, India
shobhatilak@rediffmail.com

***Abstract*- Server virtualization is an emerging technology that provides efficient resource utilization and cost-saving benefits. It consolidates many physical servers into a single physical server saving the hardware resources, physical space, power-consumption, air conditioning capacity and man power to manage the servers. Thus virtualization assists "Green Technology". Live migration is an essential feature of virtualization that allows transition of a running virtual machine from one system to another without halting the virtual machine. Live migration extends the list of benefits server virtualization provides. Almost all virtualization softwares now include support for live migration of virtual machine. Live migration is in its infant stage where security of live migration is yet to be analyzed. The usages of live migration and security exploits over it have both increased over time. The security concern of live migration is a major factor for its adoption by the IT industry. In this paper we discuss the attack model on the virtualization system and design and implement a security framework for secure live migration of virtual machines. The framework is an integrated security solution that addresses role based access policy, network intrusion, firewall protection and encryption for secure live migration process.**

***Keywords*- live migration, live migration security, live migration attack model, role based access control policy, reactive IDS, inter VM attacks.**

I. INTRODUCTION

Server virtualization is an emerging technology that provides efficient resource utilization and cost-saving benefits. It consolidates many physical servers into single physical server saving the hardware resources, physical space, power-consumption, air conditioning capacity and man power to manage the servers. In this way virtualization assists "Green Technology" [1-2]. Live migration of virtual machines (VM) is an essential feature of "virtualization" that allows transition of a running VM from one physical server to another without halting the VM [3]. Thus the services provided by VM continue to run without interruption. This ability to provide the services uninterruptedly is key requirement of many applications [4]. For example the cloud computing providers like Amazon EC2 will have thousands of VMs distributed across a set of servers. Consider that due to resource conflict the VMs running on the same physical machine may fail to serve continuously. To resolve the conflict and avoid failover of the VMs, one or more VM could be live migrated to another physical server. The user of the migrated VM is unaware of this migration as the VM is running, while migration is in progress. The live migration feature facilitates user mobility, dynamic load balancing, high availability, online system maintenance and consolidation of virtual machines.

Although virtualization technology and live migration provide many technical and cost advantages, there are many security implications associated with it [5-9]. Live migration is relatively a new concept where its security is yet to be explored. Live migration is susceptible to many attacks like "man-in-middle" attack, "denial-of-service" attack, "stackover flow" attack. The data during the migration can be sniffed or tampered easily as it is not encrypted [10-11].Thus compromising integrity and confidentiality of migrating VM data. Due to these security concerns sectors such as healthcare, banking, business and national defense hesitate to take advantage of live migration. The current state of challenge is to develop mechanism to provide secure live migration of virtual machines.

This paper defines a security framework which includes multiple defensive mechanisms addressing multiple dimensions of security for secure live migration of virtual machine. It uses Xen [12] as virtualization platform. The framework includes defining role based access control policies to protect against unauthorized usage of migration privileges. The firewall is deployed using iptable rules to protect against malicious incoming and outgoing traffic. The reactive intrusion detection system is implemented using an open source intrusion detection system called Snort to protect the system against intrusions and network attacks. Finally migration is done over a secure encrypted channel to preserve confidentiality of migration data over the network.

The following section 2 will discuss the live migration attack model. The attack model discusses the possible ways the live migration security can be threatened or compromised. The section 3 will discuss the proposed security framework, section 4 discusses the implementation and results and section 5 discusses the conclusion and future work.

II. ATTACK MODEL

The attack model of live migration process discusses how the migration process can be exploited by the attackers or intruders.

978-1-4673-6217-7/13/$31.00 ©2013 IEEE

It is assumed that live migration is done within Local Area Network (LAN), the host VM and the superuser/root user is trusted, whereas the guest VMs and the network are untrusted. The figure 1 shows the attack model for live migration of VM. The numbers 1, 2, 3, 4, inside the circle indicates the attack point, explained as follows.

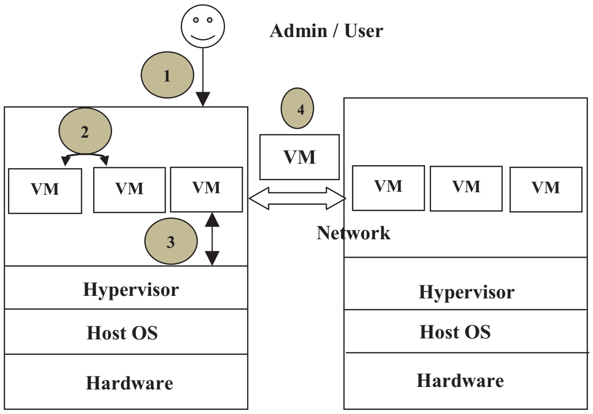


Fig. 1. Live Migration Attack Model

1. **Management console:** The system administrator controls the operation of server through the management console. Normally system admin is authorized personnel with rights to perform all operations like creating VM, deleting VM, migrating out a VM, migrating in a VM, controlling VM, configuring the VM setting etc. Thus this interface can be platform for attacks on virtual machine. The attacks can be deliberate attempts to gain access to interface or simple security breach as result of configuration errors by system administrator. Hence system administrator role is very important in preserving the security of the system. In an enterprise environment administrators with different roles like system administrator, security administrator, and network administrator will be defined. That is more persons will need to have the superuser/administrator privilege. This increases the security risk of compromising the whole system via the administrative interface.

   An attacker can exploit live migration functionality via lenient administrative interface in following ways

   - **Denial-of-service attack**: A malicious user can overload host VM by creating large number of VMs(fake VMs with no purpose) such that host sever is over loaded and will not support any further migration of VMs over it.
   - **Useless migration of guest VM**: Dynamic load balancing is a feature that does live migration of a VM from heavily loaded server to another less loaded server. A malicious user can exploit this feature by overloading a server by creating large number of VMs(fake VMs with no purpose) such that load balancing feature migrates one or more VMs to another server.
   - **VM hopping**: A malicious user can simply cause a VM to live migrate from one server to another affecting the normal operation of VM.
   - **Collocation attack**: A malicious user can migrate a malicious VM and place it on same host server as the target VM. The malicious VM then creates a covert-channel that leaks information of target VM [13].
   - **False resource advertisement:** A malicious user can make false resource availability announcement influencing VMs to migrate to the server, overloading the server.

   It is important that system administrators and users be educated to coordinate and maintain the security standards. The document [14] discusses the security base line standards to be followed by system admin managing a VM system.

2. **Inter-VM communication Attack:** Although each VM is isolated, but it can communicate with other VMs and host on same physical machine. This gives raise to potential for a malicious VM attacking other VMs running on same physical machine.
3. **Host OS and guest VM communication attack:** As already said a VM can communicate with host system and vice versa. The host OS has complete control over the all the guest VMs running over it. A compromised host can compromise the guest VM running on it. Similarly a malicious guest VM can compromise the host OS.
4. **Attack on transmission channel:** The VM migration protocol does not encrypt the migration data by default. Thus the migration data appears as clear text over the network. Thus the transmission channel is susceptible to man-in-middle attack. The attack can be passive attack or active attack. The active attack includes manipulating authentication services like sshd, bin login, Pam, manipulating kernel memory etc. passive attacks include eavesdropping of messages for sensitive data, passwords and keys, capturing authenticated packets and replying them later etc. The figure 2 shows man-in-middle attack on live migration.

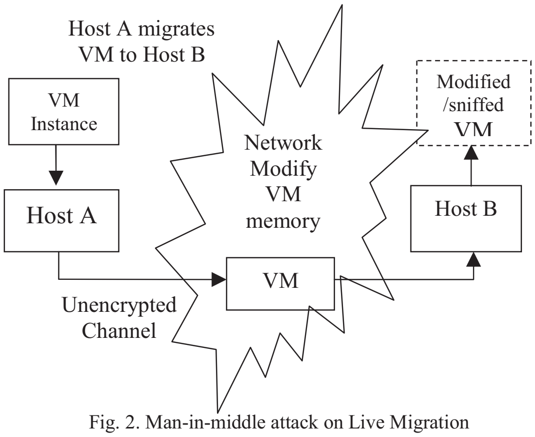


Fig. 2. Man-in-middle attack on Live Migration

Thus a secure live migration requires security to be applied before migration, during migration process and after migration is done. Typically a secure live migration must ensure that

- The source and destination machines should be trusted ones.
- An authorized access to management interface.
- Protection against network attacks, intrusions, viruses etc.
- Protection against vulnerabilities in the migration software.
- The confidentiality and integrity of migration data should be preserved during migration over the network.

In essence the live migration process requires typical defense-in-depth approach to security.

III. PROPOSED SECURITY FRAMEWORK

The figure 3 shows the security architecture for live migration of a virtual machine. It has got following modules:

1. **Common Security modules:** These security modules are applicable to the host VM as well as all the Guest VMs running over the host system.
   - i. **Attestation or platform integrity verification:** This module ensures that the migrating source or destination is trusted. That is the application will behave as expected. Hardware approaches using TPM are used for platform integrity verification [15] [16].
   - ii. **Access control policies:** This module allows the administrator to configure roles based access control (RBAC) policies that manage migration privileges. The access control policies define who (user) can migrate a VM, who can create a VM, who can delete a VM.
   - iii. **Digital signature/ MAC or checksum:** This module uses digital signature or MAC or checksum to ensure that migration data is not modified during transmission over the insecure network. Thus it protects integrity of migration data.
   - iv. **Encryption/Decryption:** This module is responsible for encrypting the migration data and metadata at the source and decrypting the same at the destination. Thus it preserves confidentiality of migration data over insecure network.
   - v. **Host System Firewall:** This module controls the communication of the host machine with the outside world. It allows the administrator to define firewall rules that controls the open ports for communication, the protocols for communication, and the list of allowed and rejected hosts/VMs. This defines a security layer for entire host system.
   - vi. **Intrusion Detection System:** An IDS detects and reports malicious intrusion attempts. The report may be in the form of alerts like log message, an email to system administrator, pop up console message etc. The IDS alerts need to be monitored to initiate a suitable action against the malicious behavior.
2. **Individual/Per VM security module:** Each VM running on the host system may have its own specific security requirement. Thus this module is defined separately for each VM running over the host system that addresses each VMs specific security requirements.
   - i. **Per-VM Firewall:** This module controls the inter-VM communication and the Host-VM communication. A Per-VM firewall is defined for each guest VM running inside the host machine. The firewall rules controls the open ports for communication, the protocols for communication, and the list of allowed and rejected hosts/VMs. It provides protection against network attacks.
   - ii. **Other components:** Each VM may have its own security components like anti-virus, spyware.

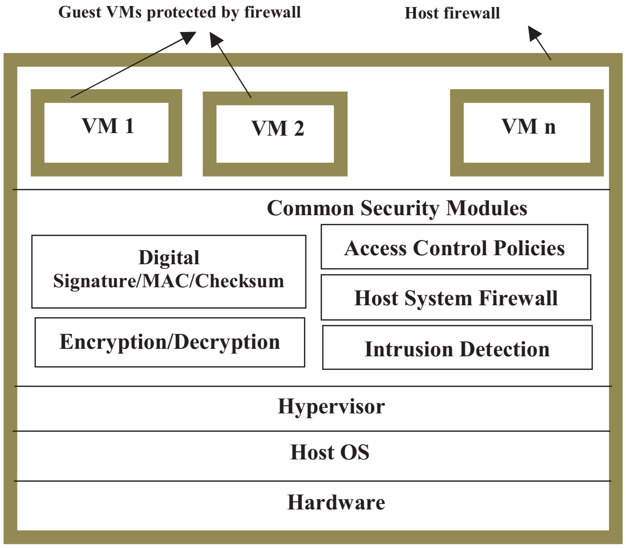


Fig. 3. SLVM Architecture

IV. IMPLEMENTATION AND RESULTS

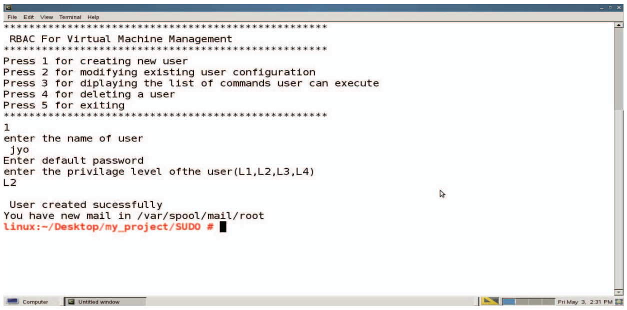
A framework named (secure live virtual machine migration) SLVM is implemented that addresses the security issues of live migration in virtualized environment. It addresses live migration between the hosts in same local area network (LAN), and stores VM images on shared storage such as network file service (NFS). It is built on Xen hypervisor. Xen is an open source para-virtual hypervisor popular among research community. The implemented framework is evaluated by simulating attacks on the virtual machine before migration, during migration and after migration.

Role based access control (RBAC): Every enterprise will have a hierarchy of administrators defined that clearly separates their responsibilities. For example: server administrators, network administrators, security administrators, system analysts etc. There should be clear separation of duties to protect against the misuse of administrative control. The role based access control policy determines what operations a particular administrator can perform. The approach used here defines four privilege levels

(L1, L2, L3 and L4) of access control policies for VM management. The privilege level restricts administrator to access only subset of VM management commands required for his role. The RBAC is implemented using SUDO tool[17]. The RBAC component presents a command line interface to the superuser to perform operations such as create a user with defined privilege level, modify existing user privilege level, delete user and query set of commands a user is authorized to execute. The four privilege levels defined are:

- L1 is the highest level of privilege. It covers entire set of VM management commands. This level should be assigned only to high level authority person like system administrator.
- L2 is next lower level privilege where only subset of all VM management commands is allowed. The commands in this privilege level are like starting VM, stopping VM, pause a VM etc. This level can be applied roles such as server administrator, network administrators etc.
- L3 is next lower level with fewer privileges than level L2. Commands such as listing the executing domains, getting information of domain, status information of VM etc are allowed in Level L2. This level is applicable to roles such as security administrators, system analysts etc.
- L4 defines normal user who does not have access to any VM management command.

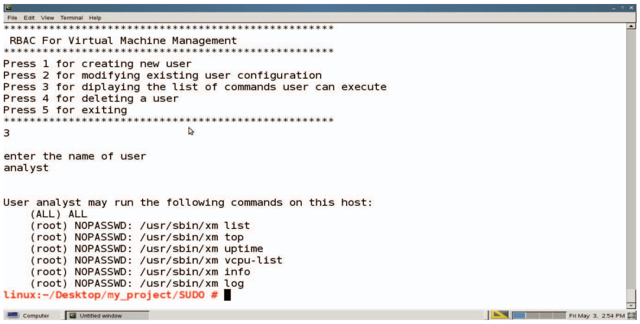
Thus this module allows super user to create users with appropriate privilege level and prevent accidental or intentional misuse of administrative interface.



```
****************************************************
 RBAC For Virtual Machine Management
****************************************************
Press 1 for creating new user
Press 2 for modifying existing user configuration
Press 3 for diplaying the list of commands user can execute
Press 4 for deleting a user
Press 5 for exiting
****************************************************
1
enter the name of user
 jyo
Enter default password
enter the privilage level ofthe user(L1,L2,L3,L4)
L2

 User created sucessfully
You have new mail in /var/spool/mail/root
linux:~/Desktop/my_project/SUDO #
```

Fig. 4. Role Based Access Control Policy



```
****************************************************
 RBAC For Virtual Machine Management
****************************************************
Press 1 for creating new user
Press 2 for modifying existing user configuration
Press 3 for diplaying the list of commands user can execute
Press 4 for deleting a user
Press 5 for exiting
****************************************************
3

enter the name of user
analyst


User analyst may run the following commands on this host:
    (ALL) ALL
    (root) NOPASSWD: /usr/sbin/xm list
    (root) NOPASSWD: /usr/sbin/xm top
    (root) NOPASSWD: /usr/sbin/xm uptime
    (root) NOPASSWD: /usr/sbin/xm vcpu-list
    (root) NOPASSWD: /usr/sbin/xm info
    (root) NOPASSWD: /usr/sbin/xm log
linux:~/Desktop/my_project/SUDO #
```

Fig. 5. RBAC for user ‘analyst’

The figure 4 shows the implemented interface for defining RBAC. It provides options to create user with a privilege level, modify the privilege level assigned to the user, querying capabilities of a user and deleting the user. Only superuser has access to this interface. The figure 5 shows list of commands user ‘analyst’ can access without giving root password.

Firewall: The inter-VM communication happens through a soft switch. This inter-VM traffic never reaches the network interface. Thus the traditional network security tool such as firewall, IDS cannot see the inter-VM traffic and protect against inter-VM attack. Also there is possibility of VMs with different trust levels (web server, a mail server, database server) likely to be hosted on same physical server. An attacker can compromise a weakly secured VM and use it to infect other VMs or take control of the host (VM escape) as there is no control over the inter-VM traffic. The solution included in SLVM framework is to add a firewall rule that rejects/drops incoming and outgoing traffic over the virtual interface by default.

```
Iptables  –A FORWARD –m physdev –physdev-is-bridged –j  DROP
```

The migration to/from untrusted node/subnet is addressed by adding a firewall rule to reject/drop traffic on the port 8002 to the identified untrusted source/destination host/subnet.

```
Iptables  –A  OUTPUT  -d destination ipaddress –m state  --state NEW –m tcp –p –dport 8002 –j DROP
```

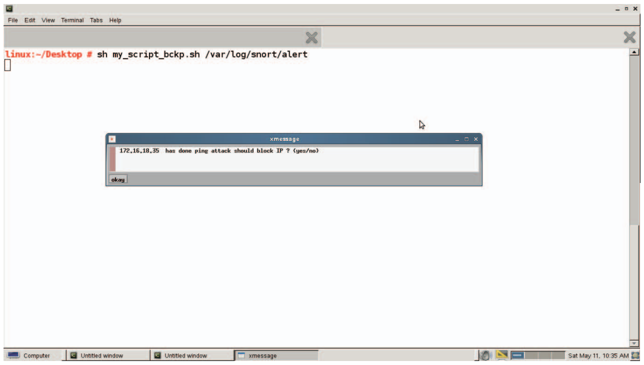
The above iptable rule drops the live migration packets to defined *destination ipaddress,* preventing migration request to untrusted destination.

Also firewall for a host and guest systems must be separately defined. It is bad idea to protect the guest VMs hosted on a server using the iptables of the host VM for two reasons. First each VM running inside host system has different security requirement. It complicates the iptables of the host server each time a VM is started/stopped/added/deleted/migrated. Second it overloads the host server. Hence separate firewalls for the host VM and guest VMs (Per-VM firewalls) are defined.

Reactive intrusion detection system: A Reactive intrusion detection system monitors the network interface, detects the intrusions or attacks, logs alert messages and initiates action against the intrusion or attack. The IDS [19] used here is called as Snort. Snort is a light weight, passive, open source intrusion detection system. It is called as passive because it just logs alert messages against unusual events or packets entering the system. Snort is signature based network intrusion detection system. It requires an attack signature be defined. The Snort sniffs the network packets and logs alert messages for packets matching the defined attack signature. By default it logs alert messages and does not take any protection action against the event.

The purpose of reactive intrusion detection system component is to detect and protect system from intruder activities at the network level. It is integration of IDS and firewall. The reactive intrusion detection system has two sub components the Snort IDS component and a reactive script. The IDS subcomponent monitors the network interface for the defined attack signature and logs alert messages when a packet matches the rule. The reactive script initiates action when attack count reaches the defined threshold level. The threshold value is to be set by administrator based on the type of application and network. The reactive script reads log file and counts number of attack attempts from each ipaddress. If the attack count from specific ipaddress exceeds threshold value then system administrator is informed and asked if identified attack source is to be blocked. The administrator then decides if the host is actual malicious attacker or a valid activity is falsely being categorized as attack (false positive alerts). Depending on administrator response either the packets from identified IP are dropped or no action is taken. This is to reduce the effect of false positive alerts.

The snort is configured to run as back ground process upon system startup. To simulate the attack the ping messages are flooded towards the host and the reactive script is scheduled to execute at repeated time interval using cron. The reactive script identifies an attack attempt and prompts the administrator against the identified attack source. Then depending on the administrator response the traffic from identified ipaddress is blocked and an email is sent to administrator. The figure 6 shows the reactive IDS output.

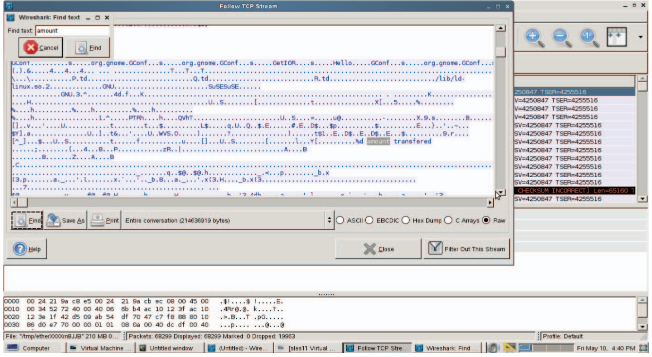


```
linux:~/Desktop # sh my_script_bckp.sh /var/log/snort/alert
```

```
172.16.18.35  has done ping attack should block IP ? (yes/no)
```

Fig. 6. Reactive IDS output

Secure encrypted channel: By default the Xen migration does not encrypt the migration data. Hence the migration data appears as clear text over the network, susceptible to active and passive attacks. To start with a script infinitely that prints "amount transferred" on the console is executed on guest VM on source host. This Guest VM is then live migrated using default live migration. A sniffing tool such as wireshark is used to capture and analyze the live migration packets. Figure 7 shows the clear text present in the captured packets of default live migration. The SLVM approach to secure transfer of migration data is to establish an encrypted channel between the migrating source and destination and perform live migration over this encrypted tunnel. The SSH tunneling establishes an encrypted tunnel between the source and destination using SSH protocol. To setup SSH tunnel the SSH client is configured to forward a specified local port to a port on the destination machine. Once the SSH tunnel is established the user can connect to local port to access the service. In this implementation a SSH tunnel is established between the nfs client and nfs server, where nfs is used as shared storage for saving and restoring of migrating VM image. The figure 8 shows wireshark sniffing results for live migration using SLVM – secure encrypted channel. It shows data is encrypted and secure against sniffing attack.



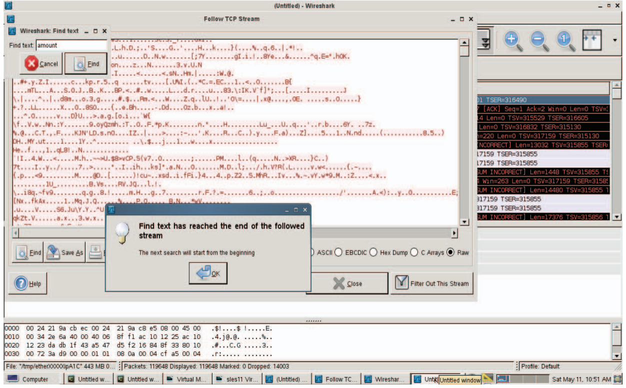
Entire conversation (214636919 bytes)

```
0000  00 24 21 9a c8 e5 00 24  21 9a cb ec 08 00 45 00   .$!....$ !.....E.
0010  00 34 52 72 40 00 40 06  6b b4 ac 10 12 3f ac 10   .4Rr@.@. k....?..
0020  12 3e 1f 42 d5 09 ab 54  df 70 47 c7 f8 88 80 10   .>.B...T .pG.....
0030  86 d0 e7 70 00 00 01 01  08 0a 00 40 dc df 00 40   ...p.... ...@...@
```

File: "/tmp/etherXXXXm8JJB" 210 MB 0... Packets: 68299 Displayed: 68299 Marked: 0 Dropped: 19963

Fig. 7. Wireshark sniffing results for default live migration

The migration downtime is used as metric to evaluate the performance of implemented secure live migration framework. Migration downtime is the time for which the VM is unavailable during the migration process. It depends on the load on VM and migration network. The migration process restores entire state of VM as it is on the destination including the network connectivity, however it may experience a short downtime during the process. Knowing this fact the downtime can be measured using ping packets. That is the VM is ping before migration is initiated and continued during the migration from source to destination until the migration is complete. After completion the ping statistics shows number of packets lost during the migration. These lost packets give the time the VM was not reachable during the transfer, which is the VM migration down time.



Find text has reached the end of the followed stream

The next search will start from the beginning

```
0000  00 24 21 9a cb ec 00 24  21 9a c8 e5 08 00 45 00   .$!....$ !.....E.
0010  00 34 2e 6a 40 00 40 06  8f f1 ac 10 12 25 ac 10   .4.j@.@. .....%..
0020  12 23 da db 1f 43 a5 47  d5 f2 16 84 8f 33 80 10   .#...C.G .....3..
0030  00 72 3a d9 00 00 01 01  08 0a 00 04 cf a5 00 04   .r:..... ........
```

File: "/tmp/etherXXXXipA1C" 443 MB 0... Packets: 119648 Displayed: 119648 Marked: 0 Dropped: 14003

Fig. 8. Wireshark sniffing results for secure migration using SLVM

The Table 1 shows the measured down time for secure live migration framework for different VM RAM sizes over shared storage using NFS.

Table 1 Migration down time analysis

	VM RAM size		
	256 MB	512 MB	1024 MB
Shared Storage	10.98 sec	21.76 sec	39.8 sec

Fig 9 below shows the graph for the measured downtime in the Table 1. The graph shows that the downtime for secure live migration increases proportionally to VM RAM size.

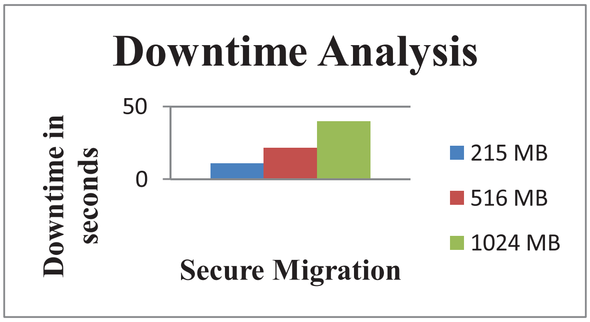


Fig. 9. Migration down time analysis

Here the average down time for VM of size 516 MB is approximately 21.76 sec i.e less than half minute. For sensitive applications where security has high priority like healthcare, banking applications, national defense, webservices, mail servers etc the implemented framework can be used at compromise of down time of few minute.

V. Conclusion and Future Work

The goal of implemented layered security frame work is to secure the live migration operation. It protects against unauthorized access to management operations, migration to/from unauthorized systems, breach of confidentiality and integrity of migration data and network attacks. The framework extends traditional security approaches to virtualization. Further layered approach reduces the complexity by segregating the distinct security components into different layers. The security approach is at application layer hence has more downtime. The future work would be to consider implementing the encryption at the hypervisor level to improve the downtime.

References

[1] Lu Liu, Masfary, O.; Jianxin Li. “Evaluation of server virtualization technologies for green IT”. In proceedings of “Service Oriented System Engineering (SOSE), 2011 IEEE 6th International Symposium”, 2011,Irvine, CA, pp 79-84.

[2] San Murugesan. “Harnessing Green IT: Principles and Practices”. Published by IEEE Computer Society, In proceeding of” IT Professional”, 2008, volume 10, pp 24-33.

[3] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield. “Live migration of virtual machines”. In Proceedings of NSDI’05, Berkeley, CA, USA. USENIX Association, 2005, pp 273–286.

[4] Tsugawa, M., Figueiredo, R. ; Fortes, J. ; Hirofuchi, T. ; Nakada, H. ; Takano, R. “On the use of virtualization technologies to support uninterrupted IT services: A case study with lessons learned from the Great East Japan Earthquake”. In proceedings of “2012 IEEE International Conference on Communications (ICC)”, Ottawa, ON, 2012, pp 6324-6328.

[5] YamunaDevi, L. Aruna, P. ; Sudha, D.D. ; Priya, N.” Security in Virtual Machine Live Migration for KVM”. In proceedings of “2011 International Conference on Process Automation, Control and Computing (PACC)”, Coimbatore, 2011, pp 1-6.

[6] T. Garfinkel, M. Rosenblum. “When virtual is harder than real: Security challenges in virtual machine based computing environments”, In Proceedings of the 10th Conference on Hot Topics in Operating Systems, ACM, Berkeley, USA, 2005, vol.10, pp.20.

[7] Steven J. Vaughan-Nichols.” Virtualization Sparks Security Concerns”, Published by the IEEE Computer Society, August 2008, DOI: 10.1109/MC.2008.276, volume 41, pp: 13-15.

[8] Jyotiprakash Sahoo, Subasish Mohapatra, Radha Lath,” Virtualization: A Survey On Concepts, Taxonomy And Associated Security Issues”, In proceedings of Second International Conference on Computer and Network Technology, 2010,DOI: 10.1109/ICCNT.2010.49, pp:222-226.

[9] Andre van Cleeff, Wolter Pieters, Roel Wieringa, “Security Implications of Virtualization: A Literature Study”, International Conference on Computational Science and Engineering, 2009, Baton Rouge, Louisiana, USA, DOI 10.1109/CSE.2009.267, pp: 353-358.

[10] Jon Oberheide, Evan Cooke, Farnam Jahanian. “Empirical Exploitation of live migration of virtual machines”. Black Hat DC Briefings, Westin Washington DC city center, February 2008.

[11] Melvin Ver. “Dynamic Load Balancing Based On Live Migration Of Virtual Machines: Security Threats and Effects”, January 2011,Thesis report Rochester Institute of Technology, B. Thomas Golisano College of Computing and Information Sciences (GCCIS), Rochester, NY, U.S.A.

[12] Xen users‘manual Xen v3.3. http://bits.xensource.com/Xen/docs/user.pdf (accessed on February 6, 2013).

[13] Thomas Ristenpart, Eran Tromer, Hovav Shacham, and Stefan Savage. “Hey, You, get off my cloud: exploring information leakage in third-party compute clouds”. In Proceedings of 16th ACM conference on computer and communication security, 2009, pp 199-212.

[14] Karen Scarfone, Murugiah Souppaya, Paul Hoffman “ Guide to Security for Full Virtualization Technologies” January 2011, Recommendations of the national institute of standards and technology, NIST special publication 800-125.

[15] Wei Wang†, Xiaoxin Wu, Ben Lin, Kai Miao, Xiaoyan Dang,”Secured and reliable VM Live Migration in Personal Cloud”, In Proceedings of international conference computer engineering and technology (ICCET), Chengdu, China 2010. pp 705-709.

[16] Boris Dandev, Ramya Jayram Masti, Ghassan Karame, Srdjan Capkun “Enabling Secure VM-vTPM Migration in private clouds” In Proceedings of the Annual computer Security Applications conference (ACSAC), Oriando, Florida, 2011, pp 187-196.

[17] www.sudo.ws (accessed on February 6, 2013).

[18] Scarfone K. and Hoffman P. “Guidelines on Firewalls and Firewall Policy”, Computer Security Division Information Technology. September 2009, Laboratory National Institute of Standards and Technology Gaithersburg, MD 20899-8930.

[19] Rafeeq UR Reheman. “Intrusion detection with Snort Advanced IDS techniques using Snort, Apache, MySQL, PHP, and ACID”, Bruce Perens’ Open Source Series, First Edition, 2003. ISBN 0-13-140733-3.