RESEARCH ARTICLE OPEN ACCESS

AMQ Protocol Based Performance Analysis of Bare Metal Hypervisors

Dr Deepak Arora[1], Varun Kumar[2], Prabhat Kumar Verma[3]
[1](Head, Department of Computer Science Engineering, Amity University, Lucknow)
[2](M. tech Scholar, Department of Computer Science Engineering, Amity University, Lucknow)
[3](M. tech Scholar, Department of Computer Science Engineering, Amity University, Lucknow)

ABSTRACT
Cloud computing is one of the most exciting technology because of its cost-reducing approach, flexibility, and scalability. Hypervisor is the essential part of cloud technology; it is a component of software that provides a virtualized hardware environment to support running multiple operating systems concurrently using one physical server. In this paper we took KVM, XEN, Hyper-V and ESXi as hypervisors. We have compared the performance of Virtual Machines (VMs) by RabbitMQ message broker server that uses Advanced Message Queuing Protocol(AMQP) for breaking messages. We establish the setup on bare metal hypervisor that is installed directly on the hardware of the system. We took SAN (Shared Storage Network) server for maintaining the storage of all VMs. By the evaluation of these hyperviosrs we got a brief idea about their performance on different parameters. These results will be beneficial to small enterprise, social group or any private IT firm which is choosing to build small cloud infrastructure with optimal benefits. Experiment results of checking the performance of VMs for all the hypervisors shows that there is performance variation on different applications and workloads of the hypervisors. None of the hypervisors outperform another at every aspect of our comparison.

Keywords - Cloud computing, Hypervisor, Performance, RabbitMQ, Virtualization

I. INTRODUCTION

In the present era of cloud computing and its services are governed by small and large public or private enterprises. Its services are being imparted in small enterprise like social groups, schools or any other private firm which needs computing services.With this service, the department's information rates can be remarkably reduced, as well as, it can quickly enhance the competitiveness of its information environment because of the following reasons: centralized monitoring, quick management, dynamic optimization, and efficient backup. To resolve several cloud issues Information Technology took the step ahead to enter in the era of cloud computing, the following definition of cloud computingprovided by NIST"Cloud computing is a model for empowering advantageous, on-demand network access to an imparted pool of configurable computing assets (e.g., systems, servers, stockpiling, requisitions, and administrations) that could be quickly provisioned and released with minimal administration exertion or administration supplier cooperation." [1].In the era of cloud computing, virtualization plays an important role by simplifying management and improving resource efficiency.

Virtualization and cloud computing were evolved to enhance the utility of computing resources whereas streamlining processes and increasing efficiencies to decrease the value of possession.Virtualization could be previous step to offer cloud computing. Most of the people can consider virtualization and cloud computing as similar in fashion but in actual cloud computing and virtualization is quite different to each other. However cloud computing can be represented as a service provider where virtualization is one of the part of physical infrastructure [2].

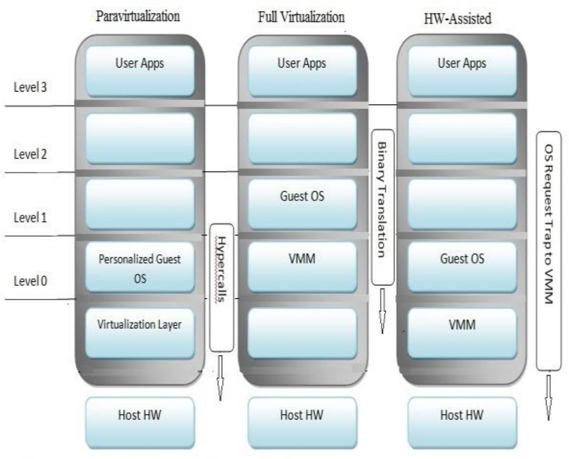

Fig. 1 Different Approaches to Providing the Virtualization Layer

Above figure describes the different approaches used today in hardware virtualization i.e. Full, Para, Hardware Assisted virtualization. Para-virtualization needs to do changes in the guest OS, basically showing the OS how to make request to the hypervisor when it need to get to limited resource of the OS. As shown in above figure level 0, level 1 and level 3 are used. It clears up the level of hardware equipment abstraction that must be given; however form control between the hypervisor and para-virtualized OS is difficult since they are controlled by diverse associations. Full virtualization supports running unmodified guests through binary translation.

Hypervisor is the best component in working with distributed or cloud computing, it is a segment of programming that gives a virtualized equipment environment to help running various working frameworks simultaneously utilizing one physical server.

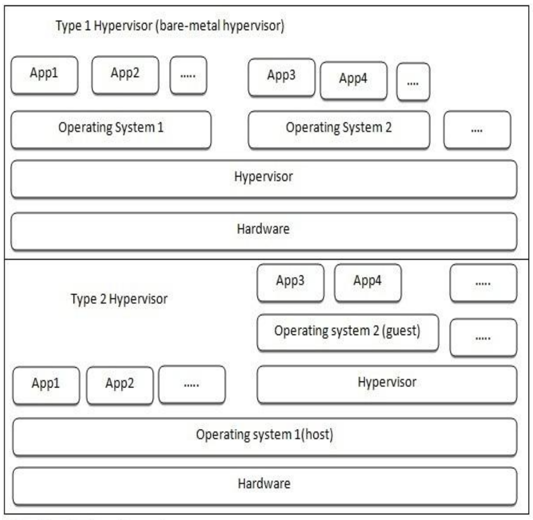

Fig 2 Classification of hypervisors

- Class 1 (or native, bare metal): “Those hypervisors that run directly on the host's hardware to control the hardware and to manage guest operating systems. A guest operating-system thus runs on another level above the hypervisor.”
- Class 2 (or hosted) “Those hypervisors run within a conventional operating-system environment. With the hypervisor layer as a distinct second software level, guest operating-systems run at the third level above the hardware.” [3]

RabbitMQ is an open source message broker that works on AMQP protocol. The RabbitMQ server is composed in Erlang and is based on the OTP framework for grouping and failover of messages. RabbitMQ offers a reliable, accessible, versatile and compact informing framework with portable and reliable throughput and inactivity. RabbitMQ integrates seamlessly into applications written in C++, Java, Python, Erlang and other standard languages. It is an open source and fully based on open standard protocol that allow the user to be free and not dependent on other libraries. It supports SMTP, STOMP and HTTP for lightweight web messaging. RabbitMQ is developed with interoperability capability for business messaging through different adapters that supports SMTP, STOMP and HTTP for lightweight web messaging. It is supported by a thriving community of active contributors[4].

In this research we perform the extensive comparison of the four hypervisors by checking their overview of performance, channels, connection, exchange and queue. We measured their performance with the use of their competing VMs with same parameters. For our comparison we took one main server and installed all the four hypervisors on that server with different disk partitions. On each partition we created VM with same configurations i.e. with 1 virtual CPU and 1GB of RAM to measure the time required for compiling the RabbitMQ Workload. Even with this operation we observe a significant performance difference as shown in figure 3

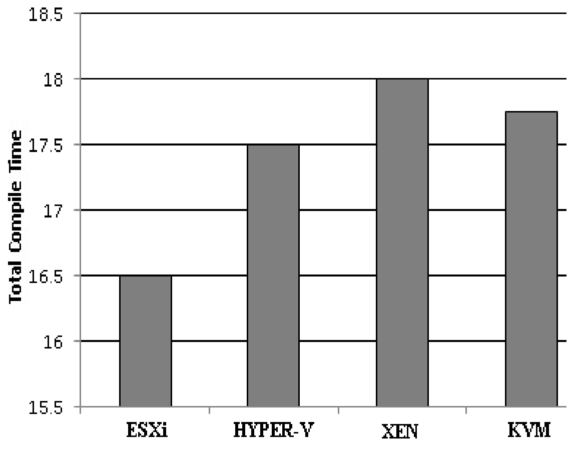

Fig 3 Rabbit MQ Workload

Our results suggest that there are performance variation depending on the type of resource stressed by RabbitMQ application. In the above bar graph it is showing the time of each hypervisor while executing the RabbitMQ workload (total message 40,000 and each message have 8 byte). By plotting the bar graph we can see that VMWare ESXi having slight better performance while comparing with other hypervisors.

II. BACKGROUND & RELATED WORK

The foundation of RabbitMQ was laid down in the year 2000 when Alexis Richardson co-founds Metalogic to develop a solution for caching java objects in financial sector. With development of Metalogic venture Alexis meets Matthias Radestock (now RabbitMQ CTO), who was at that duration working for Lshift. After that in 2005 Alexis co-founds CohesiveFT for developing application stack and tools which is now become as cloud computing. In the year 2006 Alexis and other folk at CohesiveFTrealised that message queuing needed to be a key component of the stack they were developing. Discussions between Alexis and Matthias led them to AMQP.

AMQP was originally developed for vendor-neutral protocol to manage the flow of messages for an enterprise business sys tems. It was developed by JPMorgan and iMatix from 2004 to mid-2006. In the same year 2006 Rabbit Technologies Ltd. and the first version of RabbitMQ was born and its source code is released under the Mozilla Public License. In the figure 4 it shows the hierarchy for the development phase of RabbitMQ from 1980 to 2007. The below figure shows the foundation of Teknekorn than after in the year 1990-95 MQ series development by IBM. Later after 2005 MQ series launched by Microsoft and RabbitMQ first version launched.

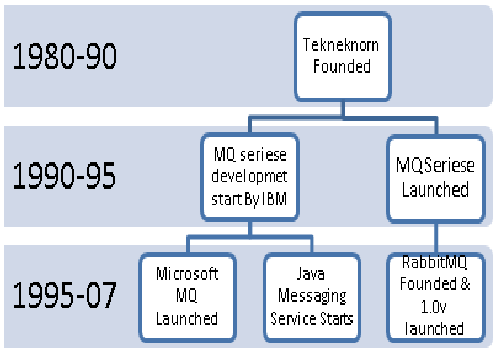
1980-90
Tekneknorn Founded
1990-95
MQ seriese developmet start By IBM
MQ Seriese Launched
1995-07
Microsoft MQ Launched
Java Messaging Service Starts
RabbitMQ Founded & 1.0v launched

Fig 4 Development phase of RabbitMQ

Today, RabbitMQ is not only a single messaging service available in the IT industry. Apart from RabbitMQ different alternatives like ActiveMQ, ZeroMQ and Apache Qpid all providing different approaches to messaging queue. RabbitMQ and Qpid they both only implement the AMQP open standard. One of the important feature of RabbitMQ is clustering of messages because of Erlang. Therefore it is very reliable and crash resistant than its competitors. RabbitMQ is easy to use and install whether to configure single server or cluster of servers it takes minimum time to work with full efficiency [5].

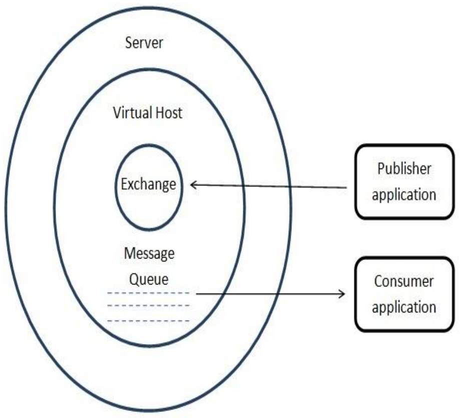

Fig 5 Architecture of RabbitMQ

In the above figure it shows the middleware server which performs the two main function first one is to accept the messages from producer and then second one is to routes them to various consumers according to their arbitrary criteria, and it buffers them in memory or on disk when consumers are not able to accept them fast enough. “In a pre-AMQP server these tasks are done by monolithic engines that implement specific types of routing and buffering”.The AMQ Protocol works on the method of breaking messages into smaller pieces and then combined them in a robust way. The task is divided in two steps: first is to accept the messages from all producers and then step two is routes all the cluster of the messages to their particular queues. The interface used between exchange and message is called “binding”. The main feature of AMQ protocol is that it can create arbitrary exchange and message queue types along with it provides runtime-programmable semantics.

We can get a suitable idea of performance comparison for selected hypervisors i.e. XEN, KVM, Hyper V, ESXi. The comparison has been performed on the basis of their hardware setting, Bytemark, Ramspeed, filebench, Netperf and some other benchmarks parameter. As a results they indicated that there is no perfect hypervisor and that different workloads may be best suited for different hypervisors. [6]

Another performance comparison of the paravirtualized version of Xen can be found in [7] and [8]. The comparison has been performed on the basis of microbenchmarks and macrobenchmarks but these papers do not discuss in depth the issues which are related to Xen product scalability. In 2007 VMware, Inc. and XenSource, Inc. published two technical reports [9][10] comparing performance obtained by VMware ESX Server, Xen, and Xen Enterprise under both microbenchmarks load and more complete load Virtualization has given a path to use the resources among the VMs by partitioning the software/hardware , time-sharing and dynamic resource sharing, it adds a new layer between the OS and hardware. Virtualization provides full support in the area of infrastructure so that VMs can be created according to the users need. We can refer this layer as hypervisor or virtual machine monitor. A hypervisor or virtual machine monitor (VMM) is a piece of computer software, firmware or hardware that creates and runs virtual machines. A system on which a hypervisor runs more than a single VM is called as host machine. All the VMs are called as guest machine. In their 1974 article "Formal Requirements for Virtualizable Third Generation Architectures" Gerald J. Popek and Robert P. Goldberg classified two types of hypervisor.\

III. METHODOLOGY

The methodology for our performance comparison is to check the performance of the hypervisor by applying a specific amount of workload on them and then comparing all VMM according to the performance of VMs on different criterion such as Time, Processor, Memory and Disk I/O.

3.1 Testing Environment

Base System/ H/w Setup	
Vendor/Product	Dell Inspiron N4050
Processor	Intel(R) Core(TM) i5
Processor Speed	2450 (MHz)
Processor Cores	4 cores, 4 threads/core
L3 Cache	3072 KB
Memory	4096 MB + 4096 MB (DDR3-1333 MHz RAM)
OS	Win 7 ; Best performance (not to best appearance)
HDD	1000 GB
Network	1 NIC
Other Software	VMware Workstation 10.0

Shared Storage Configuration

SAN Setup	
SAN OS	OpenFiler 2.6.32
RAM	1024 MB
Block storage	8 block storage of 50GB
Connection	SATA 3.0Gbp/s
Rotational Speed	5400RPM

VM Configuration Details	
Virtual CPU'S	2
Virtual CPU Speed	2450 MHz
Memory	5.7 GB
Virtual Network card	2
Virtual Network card Description	VMware Ethernet Controller
Storage Description	SAN Attached
Virtual Disk Size	50 GB
Virtual machine OS	Windows 2008 R2
Virtualization Technology	Intel VT-x/EPT support

3.2Performance Monitoring

In Windows we used RabbitMQ plug-in for monitoring all the activity. We plot graphs based on various parameters. RabbitMQ dashboard plug-in is a RabbitMQ Server inbuilt utility diagnostic tool. We can use Plug-in Monitor to view performance data. We send n number of messages from one application to another to compare the parameters such as Time, Memory, Disk Storage and Network.

3.3 Case Study

For providing the heavy workload on hypervisors we created multiple applications of RabbitMQ in .NET framework on VMs. Some applications named as producer or message generator and some are message subscriber and consumer.

The case study of these exchanges are A(i), A(ii), B(i), B(ii) and C(i)

In first case A(i) the producer produces the message and forward to the fan-out exchange. Here the exchange, sends the same message to queue in fan-out order. Auto Creation and Auto Deletion is the speciality of this queue. Now queue sends the one byte message to consumer C1 one lakh time.

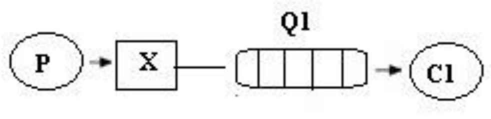

Fig 6 Sending one byte of message

In second case A(ii) we are following the same procedure for sending 1 kilobyte of messages from message generator to message subscriber.

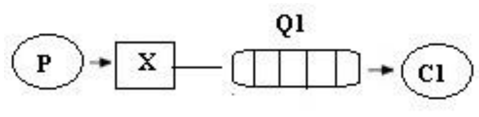

Fig 7 Sending one Kilobyte of message

In third case there is more than one consumer available so there are multiple queues and each queue is sending the message to respective consumer. So the total produced messages 1,00,000 but total delivered messages are 30,00,000. (In case of three consumers).

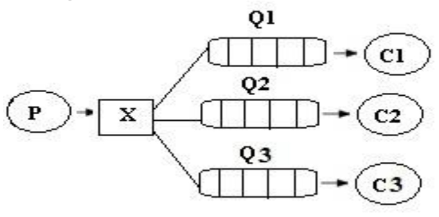

Fig 8 Sending one Byte of messages to many consumer

In fourth case B(ii) we are following the same procedure as in case B(i) but the only change is that the size of the messages will be of one kilobyte.

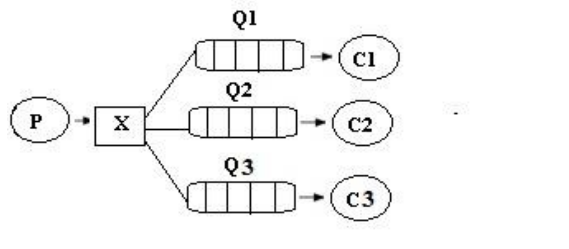

Fig 9 Sending one kilobyte messages to many consumer

We performed the above test scenario multiple times and we took the average values for all the result. Then took the log values and plotted the bar graph.

In fifth case we have generated the graph for all the four hypervisors which shows the time duration of message queuing. The graph shows the number of messages queued while sending the messages to the consumer. When the queue is short it represents that less messages are waiting and it resides completely in the memory. While when queues are large they get paged to disc.

IV. EXPERIMENTAL RESULTS

4.1 Case A(i)

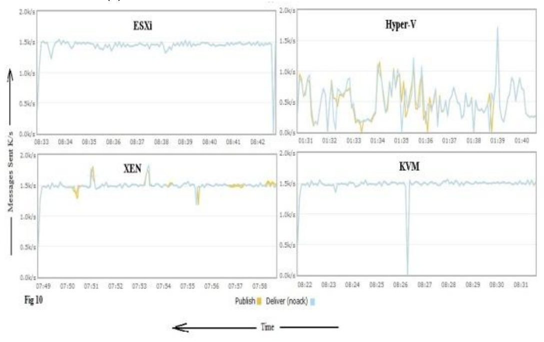

Fig 10

In this case we get the results for sending one byte of message to the single consumer through fanout exchange. The yellow line represent the publish rate while the blue one represents the deliver rate to the consumer. We get to see nearly fair performance of the messsages delivery process in the case of three hypervisors except one i.e Hyper-V. In the case of Hyper-V send rate and receive rate fluctute quite a lot. The send rate drops to zero many times therefore the latency increases steadily.

4.2. Case A(ii)

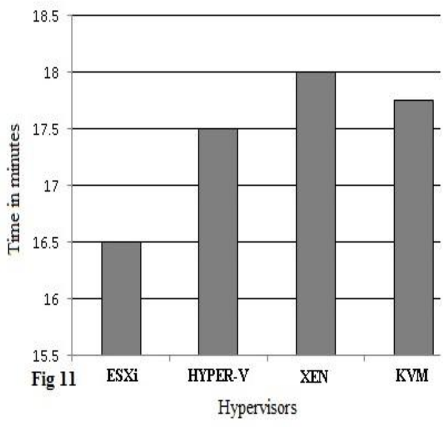

Fig 11

In this case A(ii) we get the results for sending one Kilobyte of message to the single consumer through fanout exchange. We have plotted the bar graph of time consumption of each hypervisor.We get to see nearly fair performance of the messsages delivery process in the case of three hypervisors except one i.e ESXi. In the case ESXi it takse only 16.5 min to send one Kilobyte of message to the single consumer which is less as compare to the time of other hypervisors.

4.3. Case B(i)

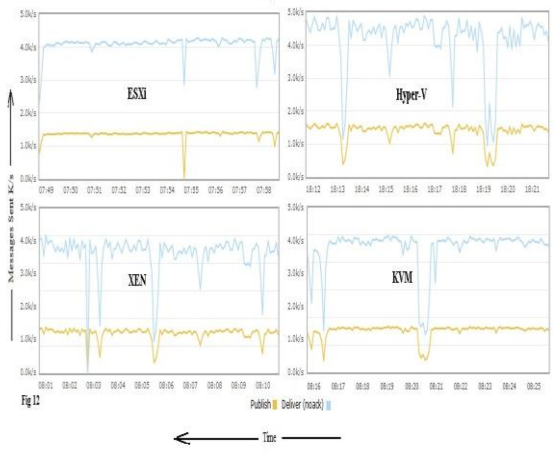

Fig 12

In this case B(i) we get the results for sending one byte of message to three consumer through fanoutexchange.Sothe total produced messages 1,00,000 but total delivered messages is 3,00,000. The yellow line represent the publish rate while the blue one represents the deliver rate to the consumer and with the help of graph we can see that blue line is three level above the yellow one showing that three times of message produced. We get to see nearly fair performance of the messsages delivery process in the case of three hypervisors except one i.e ESXi. In the case of ESXi send rate and receive rate do not fluctute a lot. The send rate drops to zero less times therefore the latency is less as compare to other hypervisors..

4.4. Case B(ii)

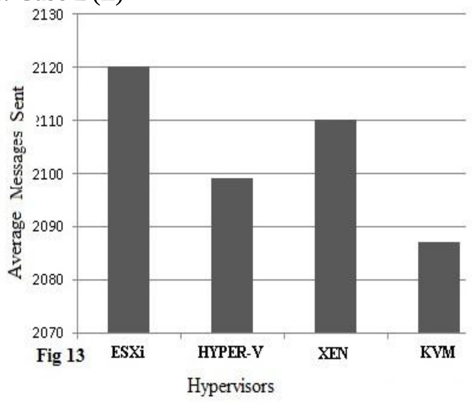

Fig 13

In this case B(ii) we get the results for sending one Kiobyte of message to three consumer through fanoutexchange.Sothe total produced messages 1,00,000 but total delivered messages is 3,00,000.We have plotted the bar graph of time consumption of each hypervisor. We get to see nearly fair performance of the messsages delivery process in the case of three hypervisors except one i.e ESXi. In the case ESXi it sends average of 2120 message to the single consumer which is more as compare to the average message of other hypervisors.In the case of ESXi send rate and receive rate fluctute quite a lot. The send rate drops to zero many times therefore the latency increases steadily thus increasing the overall time duration.

4.5 Case C(i)

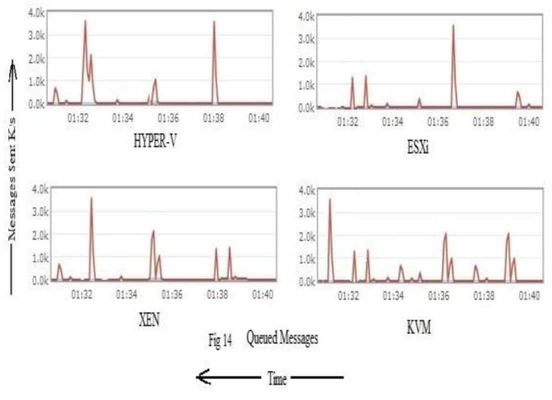

Fig 14

The above figure shows the number of messages queued while sending the messages to the consumer. When the queue is short it represents that less messages are waiting and it resides completely in the memory. While when queues are large they get paged to disc. In our case of comparison we get the conclusion that ESXi hypervisor is very efficient because in the complete process of sending the messages to consumers only 5.5 K messages are queued which is less when compare with other hypervisors.

V. CONCLUSION

In this research work we have done the comparison of hypervisors by using the RabbitMQ AMQ protocol which broadcast the messages to the users. With the help of these messages we get the performance graph of four hypervisors: Hyper-V, KVM, ESXi, and Xen in different cases. In case A(i), A(ii), B(i), C(i) ESXi is quite better while in case B(ii) Hyper-V performs slight better. Our performance results show that VMM ESXi outperform other hypervisors in all cases except for one. So, from our results we can conclude e VMware ESXi is better as compare to other hypervisors. We believe that the outcome of our study exhibit the profits of building exceptionally heterogeneous data centers and cloud infrastructure that provides a variety of virtualization and hardware platforms.

REFERENCES

[1] P. Mell and T. Grance, "*The NIST Definition of Cloud Computing*," National Institute of Standards and Technology, USA2009.

[2] Ryan Sobel. *Virtualization vs. cloud computing: There is a difference*. www.hightech-highway.com /virtualize/ virtualization-vs-cloud-computing-there-is-a-difference/, 2012.

[3] Stuart Devenish, Ingo Dimmer, Rafael Folco, Mark Roy, StephaneSaleur, Oliver Stadler, and Naoya Takizawa, *Ibmpowervm virtualization introduction and configuration*, Redbooks, 1999.

[4] RabbitMQ: About RabbitMQ,www.RabbitMQ.com

[5] Alvaro Videla, Jason J.W.Williams: RabbitMQ in Action, New York (2012)

[6] Jinho Hwang, SaiZeng and Timothy Wood, "*A Component-Based Performance Comparison of Four Hypervisors*" in Proceedings of the 9th *ACM/IFIP/USENIX International Conference on Middleware'08, pp. 366–387, Springer-Verlag* New York, Inc.

[7] Barham, P., Dragovic, B., Fraser, K., Hand, S., Harris, T., Ho, A., Neugebauer, R., Pratt, I., War_eld, A.: Xen and the art of virtualization. In: SOSP '03: *Proc of the 19th ACM symposium on Operating systems principles. pp. 164{177.* ACM, New York, NY, USA (2003)2012.

[8] Clark, B., Deshane, T., Dow, E., Evanchik, S., Finlayson, M., Herne, J., Matthews, J.N.: Xen and the art of repeated research. In: ATEC '04: *Proc of the annual conference on USENIX Annual Technical Conference. pp. 47{47. USENIX Association, Berkeley,* CA, USA (2004)

[9] VMware: A performance comparison of hypervisors. Tech. rep., VMware, Inc. (2007)

[10] XenSource: A performance comparison of commerical hypervisors. Tech. rep., Xen- Source, Inc. (2007)