**Enterprise adoption for the current cloud computing systems: an empirical survey**

**1 Introduction**

“Cloud Computing” is based on a collection of many old and few new concepts in several research fields like Service-Oriented Architectures (SOA), distributed and grid computing as well as virtualization, it has created much interest in the last few years. This was a result of its huge potential for substantiating other technological advances while presenting a superior utilitarian advantage over the currently under-utilized resources deployed at data centers. In this sense, cloud computing can be considered a new computing paradigm that allows users to temporary utilize computing infrastructure over the network, supplied as a service by the cloud-provider at possibly one or more levels of abstraction. Consequently, several business models rapidly evolved to harness this technology by providing software applications, programming platforms, data-storage, computing infrastructure and hardware as services.

Cloud computing, defined by Gartner as a style of computing where massively scalable IT-enabled capabilities are delivered “as a service” to external customers using Internet technologies, in particular should have "transformational impact" on the enterprise, according to the report. This means the technology will change the way the IT industry "looks at user and vendor relationships”.

As service provisions (a critical aspect of cloud computing) grow, vendors must become or partner with service providers to deliver their technologies indirectly to users, according to the report. User organizations will watch portfolios of owned technologies decline as service portfolios grow. The key activity will be to determine which cloud services will be viable, and when. Companies that are leading the way in the emerging cloud-computing market are Google, Amazon.com, Microsoft and Salesforce.com, according to Gartner. The impact of green IT will be high but less relevant to organizations than the impact of cloud computing, according to the report. However, organizations may use green IT practices to lessen the impact and acceleration of global climate change, and so could have a more long-term effect on that phenomenon.

From being free to an on-demand solution, cloud computing has the innate quality of adapting itself to the requirements of the user. Following the 2007-2009 recession, when cost cutting became a prominent part of business best practices, cloud computing offered a new economic model for enterprises to exploit. According to IDC's analysis, the forecast for 2013 amounts to $44.2 billion. The reason for such a new economic order is the flexibility and cost efficiency offered by cloud computing. Massive investments are being made to concentrate the hardware and the architecture is being revamped to offer global capabilities.

All in all, cloud computing promises compelling benefits to enterprises - reduction in capital and operational cost, faster processing of large data sets, increased business agility by enabling quick responses to changing conditions. But how does an enterprise evaluate, identify and prioritize the applications that are best suited for cloud.

For enterprise adoption of the current cloud computing platforms, there are varied businesses, technology, and risk considerations which can have profound effect on the overall success of cloud initiatives in an enterprise, meaning there is no "one-size-fit-all" answer for whether an application is suitable for cloud migration. Some of the questions businesses need to ask themselves before undertaking cloud initiatives are:

* What factors should be considered for cloud enablement of enterprise applications? How to judge different competing priorities?
* How to identify applications and services that are best suited for cloud migration based on business priority and technical fitment?
* How to prioritize applications and services for phase wise cloud enablement? How to avoid "gut feeling" and bring objectivity into the evaluation?
* What are the different risks involved?

There is a need, therefore, for an application portfolio assessment approach to evaluate and identify enterprise applications suited for cloud environment. Recently there are some articles about how some quantitative approaches, e.g., analytic Hierarchy Process (AHP) based multi-dimensional statistical approach can be used to undertake application portfolio assessment for cloud migration got published on IBM developer Works. However ever, there works rarely focus on the concrete enterprise software overall migration steps towards the current cloud computing platforms (systems). This article gives a detailed empirical survey of the enterprise software adoption for the current cloud computing systems, investigating the comparison between different cloud infrastructure and platform tools for enterprise adoption, analyzing it’s the advantages and disadvantages in each software reconstruction step, etc..

This article is organized as followed: Section 2 gives a general description of the five layers model of the current typical cloud computing system for enterprise adoption, elaborating each layer of its composition, limitations, and correlation with other layers; Section 3 presents a detailed a description and comparison of the current prevent platforms and tools in the cloud platform layer and cloud software layer, focusing on their performance, security and government issues for enterprise adoption and selection. Section 4 gives an experimental comparison study about virtualization technology using Xen, KVM and VMWare in the hardware and firmware layer of the cloud computing system, analyzing their feasibility, performance and security issues for software deployment. Section 5 presents a detailed description about software reconstruction towards the current cloud computing system, focusing on service automatic identification, service oriented architecture (SOA) design pattern for re-construction and performance, extensibility and security issues analysis for the migrated software. Conclusion and future work are stated in section 6.

**2 Cloud computing system overview**

There are many ontologies of cloud system. There are two popular models: Service Models and Service Deployment Models. They specify cloud architecture from different aspects and become widely accepted models.

**The Service Model**

The cloud service models give an ontological view of what a cloud service is. It is a statement of being. A cloud service system is a set of elements that facilitate the development of cloud applications. Here is a description of the three layers in the NIST service model description:

*Cloud Software as a Service (SaaS)*. The capability provided to the consumer is to use the provider’s applications running on a cloud infrastructure. The applications are accessible from various client devices through a thin client interface such as a web browser (e.g., web-based email). The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, storage, or even individual application capabilities, with the possible exception of limited user-specific application configuration settings.

*Cloud Platform as a Service (PaaS)*. The capability provided to the consumer is to deploy onto the cloud infrastructure consumer-created or acquired applications created using programming languages and tools supported by the provider. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage, but has control over the deployed applications and possibly application hosting environment configurations.

*Cloud Infrastructure as a Service (IaaS)*. The capability provided to the consumer is to provision processing, storage, networks, and other fundamental computing resources where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications. The consumer does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, deployed applications, and possibly limited control of select networking components.

The three service model elements should be deployed in a cloud environment with the essential characteristics in order to achieve a cloud status. Figure 1 gives an idea of how the service models are interrelated.

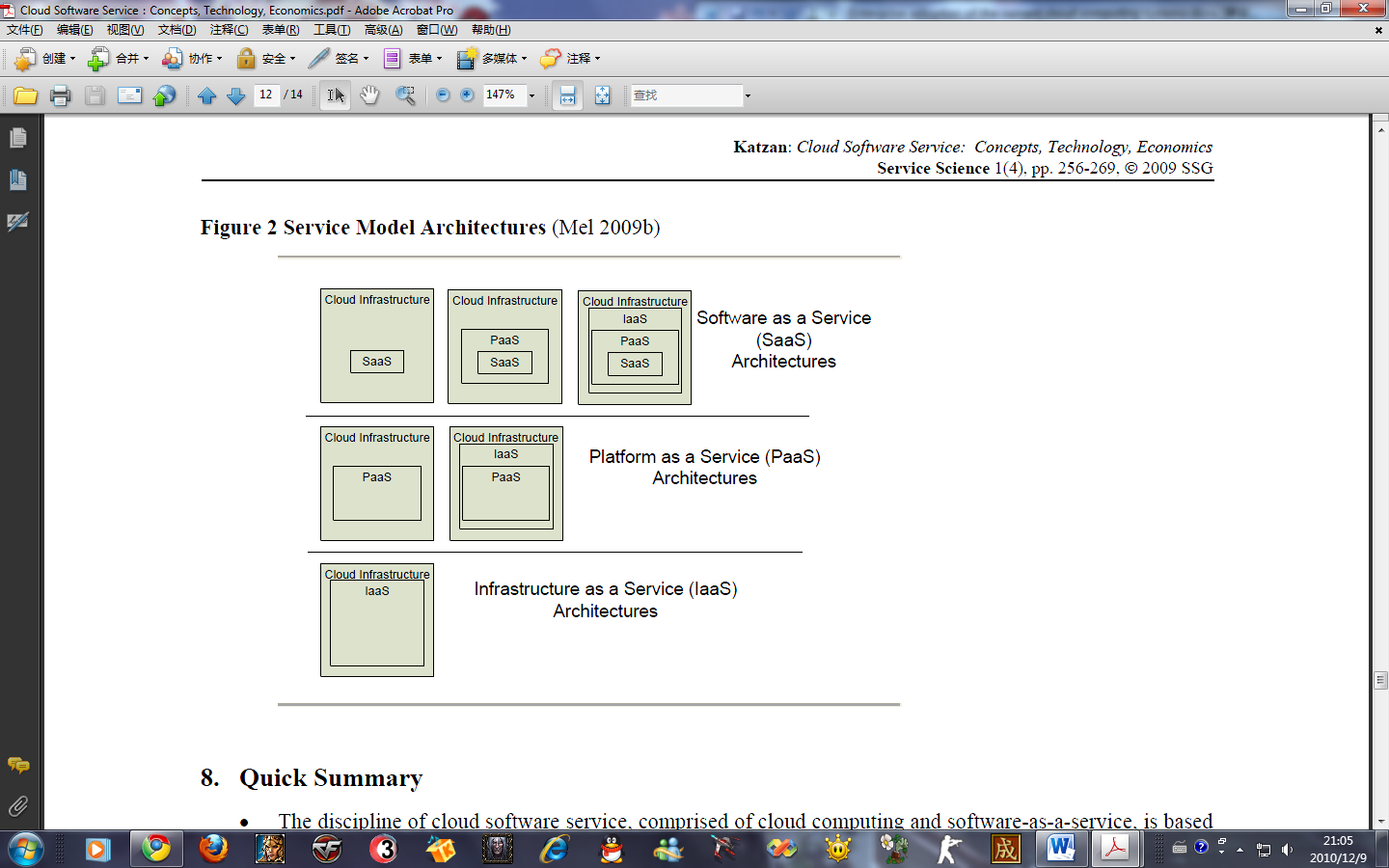


Figure1 Service models overview

**Service Deployment Models**

The essential elements of a cloud service system are given above. In order to develop enterprise-wide applications, a domain ontological viewpoint has to be assumed with deployment models from the following list:

*Private cloud*. The cloud infrastructure is operated solely for an organization. It may be managed by the organization or a third party and may exist on premise or off premise.

*Community cloud*. The cloud infrastructure is shared by several organizations and supports a specific community that has shared concerns (e.g., mission, security requirements, policy, and compliance considerations). It may be managed by the organizations or a third party and may exists on-premises or off premises.

*Public cloud*. The cloud infrastructure is made available to the general public or a large industry group and is owned by an organization selling cloud services.

*Hybrid cloud*. The cloud infrastructure is a composition of two or more clouds (private, community, or public) that remain unique entities but are bound together by standardized or proprietary technology that enables data and application portability (e.g., cloud bursting for load-balancing between clouds).

Many cloud software service application domains will be synthesized from a combination of the deployment models.

It’s a picture of VMware’s service deployment model:

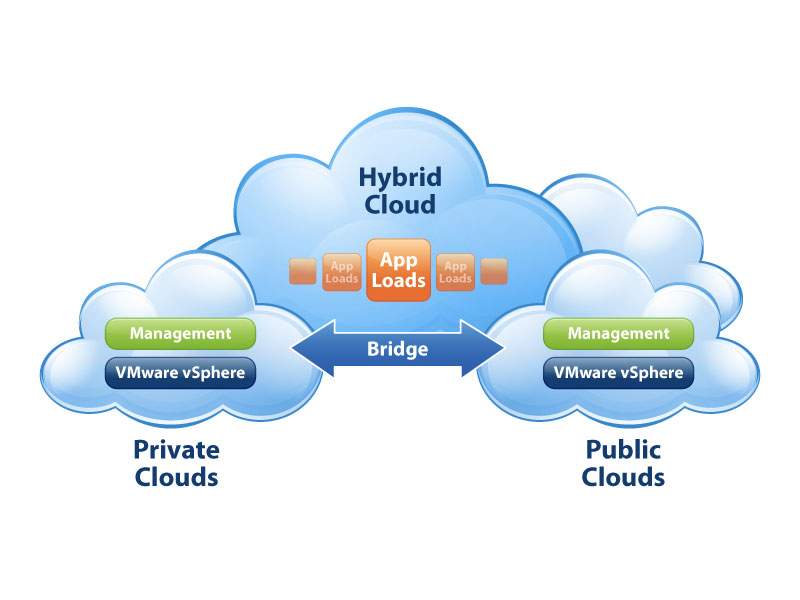


Figure 2 VMware’s service deployment model

**The model we established**

**Methodology**

In order to define ontology of cloud computing, we had to choose a methodology to classify the different cloud systems. We opt to use composability as our methodology, as it enables our proposed ontology to capture the inter-relations between the different cloud components.

**A stack of five layers**

For simplicity, we define our proposed ontology to be envisioned as a stack of layers. Each layer encompasses one or more cloud services. Cloud services belong to the same layer if they have equivalent levels of abstraction, as evident by their targeted users. For example, all cloud software environments target programmers, while cloud applications target end users. Therefore, cloud software environments would be classified in a different layer than cloud applications.

By composability, we classify one cloud layer to be higher in the cloud stack, if its services can be composed from the services of the underlying layer. Consider the cloud application layer as an example. Since we can develop cloud applications using cloud software environments, we say that cloud applications are composable from cloud software environments, and that the cloud application layer is higher in the cloud stack. Using this rationale, we formulate our proposed cloud ontology as shown in below figure (Figure 3).

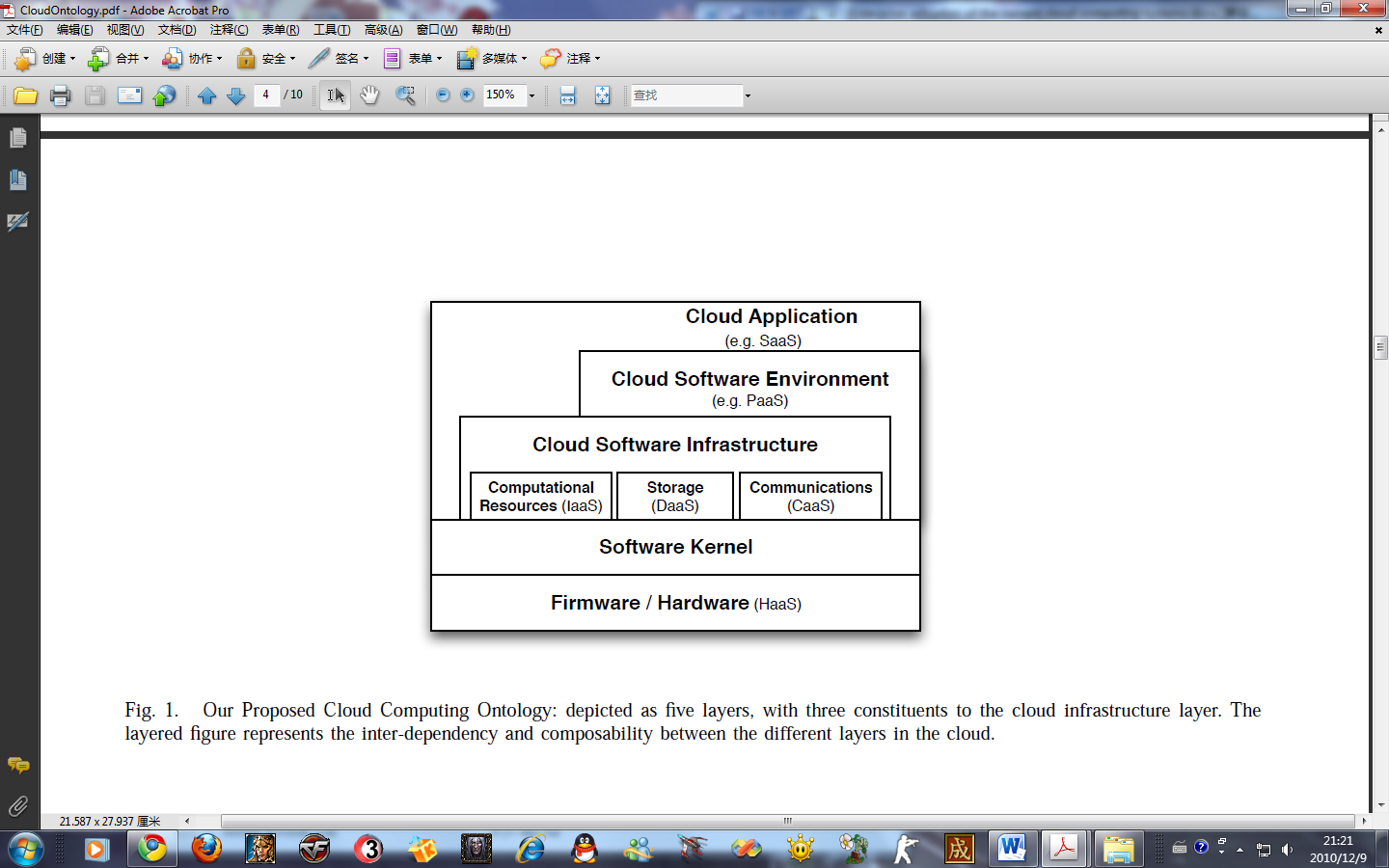


Figure 3 cloud ontology

**Characteristics**

Capturing cloud computing systems as composable services allows researchers in cloud computing to define a more robust interaction model between the different cloud entities, on both the functional and semantic levels. Further, it facilitates recognizing the inter-dependency between the different cloud systems, which in turn, accent opportunities for collaboration between different services and enhance QoS based analysis techniques that can result in better guarantees for the service levels of the different cloud systems.

**Conclusion**

Cloud computing systems fall into one of five layers: applications, software environments, software infrastructure, software kernel, and hardware. Obviously, at the Bottom of the cloud stack is the hardware layer which is the actual physical components of the system. Some cloud computing offerings have built their system on subleasing the hardware in this layer as a service. At the top of the stack is the cloud application layer, which is the interface of the cloud to the common computer users through web browsers and thin computing terminals. We closely examine the characteristics and limitations of each of the layers in the next five subsections.

**2.1 Cloud application layer**

The cloud application layer is the most visible layer to the end-users of the cloud. Normally, the users access the services provided by this layer through web-portals, and are sometimes required to pay fees to use them. This model has recently proven to be attractive to many users, as it alleviates the burden of software maintenance and the ongoing operation and support costs. Furthermore, it exports the computational work from the users’ terminal to data centers where the cloud applications are deployed. This in turn lessens the restrictions on the hardware requirements needed at the users’ end, and allows them to obtain superb performance to some of their cpu-intensive and memory-intensive workloads without necessitating huge capital investments in their local machines.

**Advantages**

As for the providers of the cloud applications, this model even simplifies their work with respect to upgrading and testing the code, while protecting their intellectual property. Since a cloud application is deployed at the provider’s computing infrastructure (rather than at the users’ desktop machines), the developers of the application are able to roll smaller patches to the system and add new features without disturbing the users with requests to install major updates or service packs. Configuration and testing of the application in this model is arguably less complicated, since the deployment environment becomes restricted, i.e., the provider’s data center. Even with respect to the provider’s margin of profit, this model supplies the software provider with a continuous flow of revenue, which might be even more profitable on the long run. This model conveys several favorable benefits for the users and providers of cloud applications, and is normally referred to as *Software as a Service (SaaS)*. As such, the body of research on SOA has numerous studies on composable IT services which have direct application to providing and composing *SaaS*.

**Relationship with other layers**

Our proposed ontology illustrates that cloud applications can be developed on the cloud software environments or infrastructure components (as discussed in the next two subsections). In addition, cloud applications can be composed as a service from other cloud services offered by other cloud systems, using the concepts of SOA. For example, a payroll application might use another accounting SaaS to calculate the tax deductibles for each employee in its system without having to implement this service within the payroll software. In this respect, the cloud applications targeted for higher layers in the stack are simpler to develop and have a shorter time-tomarket. Furthermore, they become less error-prone since all their interactions with the cloud are through pretested APIs. Developed for a higher cloud-stack layer, the flexibility of the applications is however limited and this may restrict the developers’ ability to optimize their applications’ performance.

**Some deployment issues**

Despite all the advantageous benefits of this model, several deployment issues hinder its wide adoption. Specifically, the security and availability of the cloud applications are two of the major issues in this model, and they are currently avoided by the use of lenient service level agreements (SLA). Furthermore, coping with outages is a realm that users and providers of *SaaS* have to tackle, especially with possible network outage and system failures. Additionally, the integration of legacy applications and the migration of the users’ data to the cloud is another matter that is also slowing the adoption of *SaaS*. Before they can persuade users to migrate from desktop applications to cloud applications, cloud applications’ providers need to address end-users’ concerns about security and safety of storing confidential data on the cloud, users authentication and authorization, up-time and performance, as well as data backup and disaster recovery and provide reliable SLAs for their cloud applications.

**Products:**

As shown in Table 1.

Table 1 Cloud application products examples

|  |  |
| --- | --- |
| Acrobat.com  Google Apps  BigMachines  Google Docs  Icloud  KnowledgeTree  GetApp.com  Gigya  Imaginatik | RightNow Technologies  Salesforce.com  Smartsheet  Web 2.0(Twitter, facebook)  Yudu Media  Office web apps  Online office suite  Microsoft Forefront Online protection for Exchange  MobileMe |

**2.2 Cloud software environment layer**

The second layer in our proposed cloud ontology is the cloud software environment layer (also dubbed the software platform layer). The users of this layer are cloud applications’ *developers*, implementing their applications for and deploying them on the cloud. The providers of the cloud software environments supply the developers with a programming-language-level environment with a set of well-defined APIs to facilitate the interaction between the environments and the cloud applications, as well as to accelerate the deployment and support the scalability needed of those cloud applications. The service provided by cloud systems in this layer is commonly referred to as *Platform as a Service (PaaS)*. One example of systems in this category is Google’s App Engine , which provides a python runtime environment and APIs for applications to interact with Google’s cloud runtime environment.

**Advantages**

Developers reap several benefits from developing their cloud application for a cloud programming environment, including automatic scaling and load balancing, as well as integration with other services (e.g. authentication services, email services, user interface) provided to them through the *PaaS*-provider. In such a way, much of the overhead of developing cloud applications is alleviated and is handled at the environment level. Furthermore, developers have the ability to integrate other services to their applications on-demand. This in turn makes the cloud application development a less complicated task, accelerates the deployment time and minimizes the logic faults in the application. In this respect, a Hadoop deployment on the cloud would be considered a cloud software environment, as it provides its applications’ developers with a programming environment, i.e. map reduce framework for the cloud. Similarly, Yahoo’s Pig, a high-level language to enable processing of very large files on the hadoop environment may be viewed as an open-source implementation of the cloud platform layer. As such, cloud software environments facilitate the process of the development of cloud applications.

**Products**

Table 2 Cloud software environment products examples

|  |  |
| --- | --- |
| Amazon Web Services  AppScale  Azure services Platform  Caspio  Cordys Process Factory  Engine Yard  Force.com  Google App Engine  Amazon SimpleDB | Heroku  Hybrid Web Cluster  OrangeScape  Rackspace Cloud  Rollbase  Squarespace  Sun Cloud  Vertebra  Wolf Frameworks |

**2.3 Cloud software infrastructure layer**

The cloud software infrastructure layer provides fundamental resources to other higher-level layers, which in turn can be used to construct new cloud software environments or cloud applications. Our proposed ontology reflects the fact that the two highest levels in the cloud stack can bypass the cloud infrastructure layer in building their system. Although this bypass can enhance the efficiency of the system, it comes at the cost of simplicity and development efforts.

Cloud services offered in this layer can be categorized into: computational resources, data storage, and communications.

**Computation resources**

Virtual machines (VMs) are the most common form for providing computational resources to cloud users at this layer, where the users get finer-granularity flexibility since they normally get super-user access to their VMs, and can use it to customize the software stack on their VM for performance and efficiency. Often, such services are dubbed Infrastructure as a Service (IaaS). Virtualization is the enabler technology for this cloud component, which allows the users unprecedented flexibility in configuring their settings while protecting the physical infrastructure of the provider’s data center. Recent advances in OS Virtualization have made the concept of IaaS plausible. This was specifically enabled by two virtualization technologies: paravirtualization and hardware-assisted virtualization. Although both virtualization technologies have addressed performance isolation between virtual machines contending on common resources, performance interference between VMs sharing the same cache and TLB hierarchy cannot yet be avoided. Further, the emergence of multicore machines into mainstream servers exacerbates this performance interference problem. In turn, the lack of strict performance isolation between VMs sharing the same physical node has resulted in the inability of cloud providers to give strong guarantees for performance to their clients. Instead, they offer them unsatisfactory SLAs in order to provide a competitive pricing for the service. Such weak guarantees, unfortunately, can inject themselves up the layers of the cloud stack, and affect the SLAs of the cloud systems built above the IaaS’s SLAs.

**Data storage**

The second infrastructure resource is data storage, which allows users to store their data at remote disks and access them anytime from any place. This service is commonly known as Data-Storage as a Service (DaaS), and it facilitates cloud applications to scale beyond their limited servers. Data storage systems are expected to meet several rigorous requirements for maintaining users’ data and information, including high availability, reliability, performance, replication and data consistency; but because of the conflicting nature of these requirements, no one system implements all of them together. For example, availability, scalability and data consistency can be regarded as three conflicting goals. While those features are hard to be met with general data storage systems, DaaS-providers have taken the liberty of implementing their system to favor one feature over the others, while indicating their choice through their SLA. These implementations have borrowed their fundamental ideas from proceeding research and production systems. Some examples of data storage systems are: distributed file systems, replicated relational databases (RDBMS) and key value stores. RDBMS, for example opt to present a stricter consistency model at the cost of the availability of the data, while key-value stores have placed more importance on the availability of the data while relaxing the consistency model for the storage. In this respect, the cloud DaaS has inherited the different characteristics of today’s data storage systems.

**Communication**

As the need for a guaranteed quality of service (QoS) for network communication grows for cloud systems, communication becomes a vital component of the cloud infrastructure. Consequently, cloud systems are obliged to provide some communication capability that is service-oriented, configurable, schedulable, predictable, and reliable. Towards this goal, the concept of Communication as a Service (CaaS) emerged to support such requirements, as well as network security, dynamic provisioning of virtual overlays for traffic isolation or dedicated bandwidth, guaranteed message delay, communication encryption, and network monitoring. Although this model is the least discussed and adopted cloud service in the commercial cloud systems, several research papers and articles have investigated the various architectural design decisions, protocols and solutions needed to provide QoS communications as a service. One recent example of systems that belong to CaaS is Microsoft Connected Service Framework (CSF). VoIP telephone systems, audio and video conferencing as well as instant messaging are candidate cloud applications that can be composed of CaaS and can in turn provide composable cloud solutions to other common applications.

Several common design features are shared between the three infrastructure components. To name a few, security of the services, their availability and quality are among the most commonly addressed concerns for these cloud infrastructure components. Encryption through X509 certificates is customary for these systems. However, providing other security mechanisms for service-oriented architectures is a rich area of research with little focus so far from the SOA and security communities. User-interface to the cloud infrastructure components varies substantially from one system to another. SOAP and REST are examples of interface protocols used with some cloud computational resources. However, an encompassing characteristic of all cloud software infrastructure constituents is their service-oriented implementation. Therefore, designing a unified interface, and interacting with it through a web portal that communicates with the web services can be a plausible approach to providing a standardized interface to the cloud services at this level of the cloud.

**Products**

Table 3 Cloud software infrastructure products

|  |  |
| --- | --- |
| Amazon EC2, S3  AppNexus  Cloud.com  Cloud.bg  Cloudera  Cloudkick  EnStratus  Enomaly Inc  Eucalyptus  GoGrid  Turkey Linux Virtual App Lib | Hadoop  Host1Free  IIand  Jitscale  Kaavo  Nimbus  OpenStack  Rackspace CLoud  RightScale  Scalr  Sun Cloud |

**2.4 Hardware and firmware layer**

The bottom layer of the cloud stack in our proposed ontology is the actual physical hardware and switches that form the backbone of the cloud. In this regard, users of this layer of the cloud are normally big enterprises with huge IT requirements in need of subleasing *Hardware as a Service* (*HaaS*). For that, the *HaaS* provider operates, manages and upgrades the hardware on behalf of its consumers, for the life-time of the sublease. This model is advantageous to the enterprise users, since they do not need to invest in building and managing data centers. Meanwhile, *HaaS* providers have the technical expertise as well as the cost-effective infrastructure to host the systems. One of the early examples *HaaS* is Morgan Stanley’s sublease contract with IBM in 2004. SLAs in this model are more strict, since enterprise users have predefined business workloads whose characteristics impose strict performance requirements. The margin benefit for *HaaS* providers materialize from the economy of scale of building huge data centers infrastructures with gigantic floor space, power, cooling costs as well as operation and management expertise. *HaaS* providers have to address a number of technical challenges in operating and managing their services. Efficiency, ease and speed of provisioning such large scale systems, for example is a major challenge. Remote scriptable boot-loaders is one solution to remotely boot and deploy complete software stacks on the data centers. PXE and UBoot are examples of remote bootstrap execution environments that allow the system administrator to stream a binary image to multiple remote machines at boot-time. One example of such systems is IBM Kittyhawk, a research project that uses UBoot to script the boot sequence of thousands of remote Bluegene/P nodes over the network. Other examples of challenges that arise at this cloud layer include data center management, scheduling, and power-consumption optimizations.

**Products**

Table 4 Cloud hardware products

|  |  |
| --- | --- |
| KVM  XEN  VMware  Sun xVM | Oracle VM  Parallels Workstation  VirtualBOx  Microsoft Virtual PC |

**3 A comparison study of cloud infrastructure and platform tools**

**3.1 Cloud infrastructure as a service (IAAS)**

**3.1.1 An introduction of cloud infrastructure**

Cloud infrastructure products are the most general offering that Amazon has pioneered and where RightScale offers its management platform. Developers and system administrators obtain general compute, storage, queuing, and other resources and run their applications with the fewest limitations. This is the most powerful type of cloud in that virtually any application and any configuration that is fit for the internet can be mapped to this type of service. Of course it also requires more work on the part of the buyer, which is where RightScale comes in to help with set-up and automation.

**3.1.2 Cloud infrastructure products list**

|  |  |  |
| --- | --- | --- |
| **Cloud infrastructures** | **If OpenSource** | **References** |
| Amazon EC2, S3 | No | <http://aws.amazon.com/ec2/> |
| Cloud.com | Yes | <http://www.cloud.com/> |
| Cloud.bg | No | <http://www.cloud.bg/en/> |
| Cloudera | Yes | [http://www.cloudera.com](http://www.cloudera.com/) |
| Cloudkick | No | <https://www.cloudkick.com/> |
| EnStratus | No | [http://www.enstratus.com](http://www.enstratus.com/) |
| Enomaly Inc | No | <http://www.enomaly.com/> |
| Eucalyptus | Yes | <http://www.eucalyptus.com/> |
| GoGrid | No | <http://www.gogrid.com/> |
| TurnKey Linux Virtual Appliance Library | Yes | <http://www.turnkeylinux.org/> |
| Host1Free | No | <http://www.host1free.com/> |
| Jitscale | No | <http://www.jitscale.com/> |
| Kaavo | No | <http://www.kaavo.com/> |
| Nimbus | Yes | <http://www.nimbusproject.org/> |
| OpenStack | Yes | <http://www.openstack.org/> |
| Rackspace Cloud | Yes | <http://www.rackspacecloud.com/> |
| RightScale | No | <http://www.rightscale.com/> |
| Scalr | No | <https://www.scalr.net/> |

The details of cloud infrastructure and cloud infrastructure comparisons are shown in “Cloud infrastructure list.docx” and “Cloud infrastructure comparison”.

**3.1.3 Cloud infrastructure product research**

According to research purpose, we did research on four open source products.

* Cloudera
* TurnKey Linux Virtual Appliance Library
* Eucalyptus
* RightScale

The research will focus on the follow contents.

* Performance
* Security
* Governance concerns
* Economic value
* Industry landscape
* Tool sets(tool supporting)
* Scalability
* Others

The research results on the four target products:

“Research on Rackspace.doc”;

“Research on TurnkeyLinux”;

“Research On Cloudera”;

“Research on Eucalyptus”.

**3.2 Cloud software as a service (SAAS)**

**3.2.1 SAAS products list**

There are list of solution which provides Cloud Infrastructures for Hardware as a service (HAAS) or Software as a Services(SAAS).

**AllenPort**  
 AllenPort’s technology handles file management chores like backup, file sharing, disaster recovery, remote access and managing user requirements.

**AppZero**

AppZero offers OS-free Virtual Application Appliances that are self-contained, portable units, meaning enterprises can experiment with moving applications to the cloud while avoiding cloud lock-in.

**Boomi**

Boomi and its AtomSphere connect any combination of cloud and on-premise applications without software or appliances.

**CA**

NetQoS’s monitoring prowess and Cassatt’s data center automation and policy-based optimization expertise, CA can boost the functionality of its Spectrum Automation Manger to let it manage network and systems traffic in both public and private cloud computing environments.

**Cast Iron Systems**

Cast Iron offers an option for integrating SaaS applications with the enterprise. That method, which involves configuration, not coding, can in some cases slash integration costs up to 80 percent.

**Citrix**

Citrix Cloud Center (C3) ties together virtualization and networking products, arming cloud providers with a virtual infrastructure platform for hosted cloud services. The service, which is available on a monthly, usage-based pricing model and support mode, is an architecture comprising five key components: a platform powered by Citrix XenServer; applications and desktop services via Citrix XenApp; delivery powered by Citrix NetScaler; a bridge using Citrix Repeater; and orchestration through Citrix Workflow Studio.

**Elastra**

Elastra makes software that enables enterprises to automate modeling, deployment and policy enforcement of the application infrastructure. Its products tie in with provisioning and virtualization tools. Elastra’s Enterprise Cloud Server software handles the management and provisioning of complex systems. Users can quickly model and provision application infrastructure; automate changes to the system deployment process; efficiently utilize internal, external and virtualized resources on demand and enforce IT policy rules. Elastra Cloud Server can also run on Amazon Web Services.

**EMC**

With its Atmos and Atmos onLine offerings, EMC is evangelizing its approach to the cloud to deliver scalability, elasticity and cost savings by building, virtualizing and deploying services and applications. Atmos onLine is a cloud storage service built on Atmos, EMC’s policy-based information management platform. EMC Atmos onLine provides Cloud Optimized Storage, or COS, capabilities for moving and managing large amounts of data with reliable service levels and in a secure fashion.

**Informatica**

Informatica basically pioneered cloud computing for data integration, offering a host of offerings for customers of various shapes and sizes. It offers fast and easy pay-as-you-go and pay-for-use options that let users move data into or out of the cloud or manage data within the cloud of from one app to another.  
  
**NetApp**  
 Call it IT-as-a-Service (ITaaS) or call it an enterprise cloud infrastructure. Data ONTAP 8, NetApp’s latest cloud computing infrastructure, ties together its two previously separate platforms: Data ONTAP 7G and Data ONTAP GX. It delivers improved data management functions and tighter integration with data center management systems. Ultimately, NetApp Data ONTAP 8 enables storage, server, network and applications layers to talk to each other.  
  
**New Relic**  
 New Relic is running full throttle with its RPM offering, an on-demand performance management tool for Web applications. It takes only minutes to implement and offers visibility and code-level diagnostics for Web apps deployed in both private and public clouds, along with traditional and dedicated infrastructures, and any combination thereof. With RPM, New Relic delivers real-time metrics, unlocking the ability to monitor, troubleshoot and fine tune app performance in the cloud.  
  
**Novell**  
 Novell is looking to the cloud to tie together all things IT. It is combining products like Moblin, a cloud-centric desktop OS developed by Novell and Intel; the SUSE Appliance Program, a program for ISVs to build software appliances and receive go-to-market support; Novell Cloud Security Service; and PlateSpin Workload Management Solutions for IT managers.  
  
**Open Nebula**  
 This open-source toolkit fits snuggly into existing data center environments to build any type of cloud deployment. OpenNebula can be used to manage virtual infrastructure in the data center or to manage a private cloud. It also supports hybrid clouds to combine local infrastructure with public cloud infrastructure for hosting environments. Additionally, it supports public clouds by offering cloud interfaces to expose its functionality for virtual machine, storage and network management.  
  
**OpSource**  
 OpSource is all about cloud operations, offering everything from an enterprise-grade cloud infrastructure to fully managed hosting and apps management. Essentially, OpSource Cloud is a virtual private cloud within the public cloud, giving users control over their degree of Internet connectivity. Meanwhile, OpSource On-Demand combines technical operations, application operations and business operations into a Web operations offering that includes application management, compliance and business services. Lastly, OpSource Billing CLM is a self-service offering for SaaS and Web customer on-boarding, subscription management and payment processing.  
  
**Paglo**  
 This IT search and management service startup recently launched its Log Management application to let IT managers capture and store their logs as well as search and analyze them in the cloud. Paglo compares it to a Google-like search for logs, collecting data from all network devices. Paglo has also recently launched a new application to monitor Amazon EC2 application instances, such as disk reads and writes, CPU utilization and network traffic. Users can access the cloud-based information from any Web browser.  
  
**RightScale**  
 RightScale’s Cloud Management Platform eases deploying and managing apps in the cloud and enables automation, control and portability. The platform helps users get into the cloud quickly with cloud-ready ServerTemplates and best-practice deployment architectures. And users retain complete visibility into all levels of deployment by managing, monitoring and troubleshooting applications. Lastly, RightScale’s Cloud Management Platform helps users avoid lock-in by letting them choose their deployment language, environment, stack, data store and cloud for portability.  
  
**Stoneware**  
 Stoneware’s mission is simple: To enable organizations to move from a client-centric to a Web-based, private cloud computing environment. With products aimed specifically at core verticals education, healthcare, manufacturing, legal, financial and enterprise Stoneware offers private cloud technology that is being used to create solutions that enable organizations to access applications, content, data and services from anywhere in a secure fashion.  
  
**VMware**  
 Last August, VMware acquired SpringSource which provides Web application development and management services. SpringSource speeds the delivery of applications in the cloud using a process that has become known as lean software. VMWare also acquired Hyperic, an open-source monitoring and troubleshooting vendor. The VMWare-SpringSource-Hyperic trifecta creates an amalgamation that ties together VMWare’s virtualization vision, SpringSource’s strong development tools and application servers as well as Hyperic’s monitoring.  
  
**Zeus Technology**  
 Zeus gives users the ability to create, manage and deliver online services in cloud, physical or virtual environments, letting companies visualize and manipulate the flow of traffic to Web-enabled apps. And early this year, they will release the Zeus Cloud Traffic Manager so customers can monitor and control cloud usage, offering a single control point for distributed applications, reporting on datacenter usage and allowing for goals like cost, SLA, security and compliance to be applied.

**3.2.2 SaaS product selection**

Please see the attachment: “SaaS selection.docx”

**3.3 Cloud platform as a service (PAAS)**

**3.3.1 PAAS products list**

Please see the attachment: “Cloud platform tools.docx”.

**3.3.2 PAAS comparisons**

Please see the attachment: “Comparison of Several Cloud Computing Platforms.doc”.

**4 An experimental comparison study between Xen, KVM and VMWare**

**4.1 Comparisons between Xen, KVM and VMWare**

Table 5 Comparisons between Xen, KVM and VMWare

|  |  |  |
| --- | --- | --- |
|  | **KVM** | **Xen** |
| **Linux Environment** | KVM uses a set of Linux kernel modules to provide VT support. It can run on a stock Linux kernel, but it must runs on x86 and x86-64 systems with hardware supporting virtualization extension. | Xen requires a heavily patched Linux kernel, on which development lags behind the mainline kernel. We need to choose a Linux distro that ships with Xen support. |
| **Virtualization Approach** | KVM is a full virtualization solution. This approach allows the guest operating system to run without modifications. The kernel component of KVM is included in mainline Linux. KVM supports I/O Para-virtualization. | Xen can work both in Para-virtualization or HVM mode. Through Para-virtualization, Xen can achieve very high performance. Para-virtualization actually modifies the guest operating system code. |
| **Virtualization**  **Model** | Hypervisors run as a normal program inside a normal operating system. This OS is known as the host. Each guest OS runs as a process in the host OS. These processes can be manipulated just like any other process. | Hypervisors directly interface with the system hardware. All operating systems run inside a virtual machine. There is usually a special, privileged virtual machine that can manage the others. |
| **Scalability** | KVM is not able to maintain performance as the number of guests increased. | Xen has excellent scalability and that Xen is able to share resources among guests well. |
| **Management** | Ease of management is achieved by using [**libvirt**](http://libvirt.org/) which will present the same interface whether you use Xen or KVM. | |
| **Performance**  **Isolation** | KVM shows good isolation properties for all of the stress tests and unexpectedly good performance for the network sender. However, KVM shows unexpectedly poor performance for the disk test and the network receiver test. | Xen shows good isolation properties for the memory, fork, CPU, and disk stress tests as seen in the Normal VM column. Xen shows very little isolation for the network sender and no isolation for the network receiver. |
| **CPU**  **Performance** | CPU performance provided by KVM comparable to the one provided by xen and in some cases it’s even better. | |
| **Network**  **Performance** | KVM shows some strange asymmetric behavior, but anyway we consider them acceptable. | Xen is symmetric in every circumstance and so proves to be a better choice in situations where network outbound speed is critical. |
| **I/O**  **Performance** | I/O performance is of great importance to a hypervisor. I/O is also a huge maintenance burden, due to the large number of hardware devices that need to be supported, numerous I/O protocols, high availability options, and management for it all. Xen with Para-virtualization access is by far the best solution, particularly reading from disk. And perform very badly in disk writing as well as KVM, leaving a lot of room for improvement in future release of the hypervisor. | |
| **Security** | 1. **General Security Issues**  |  |  |  | | --- | --- | --- | | **Security concerns** | KVM | Xen | | Size of software stack | Medium | Medium to High | | Number of interfaces | Medium | High | | Assurance of Development | Linux Kernel: High  QEMU: Medium | Medium |  1. **Security Comparison Based on Scenarios**  |  |  |  |  | | --- | --- | --- | --- | | **Scenarios** | KVM | Xen | VMWare | | Assurance of protection against VM accessing unassigned resources  mediated by Para-virtualized drivers | Medium | Low | Low | | Assurance of protection against VM accessing unassigned resources mediated by full virtualization support software | Medium | Low | N/A | | Assurance of protection against subversion of trusted VMM software – subversion of Hypervisor | High | High | Medium | | Assurance of protection against Subversion of trusted VMM software – subversion of other virtual machines | Medium | Medium | N/A | | Assurance of protection against Subversion of trusted VMM software – subversion of boot process | High | Medium | N/A | | Assurance of protection against one VM causing a DoS of other VMs | High | Medium | Medium | | Support for sandboxing usage | High | Medium | Low | | VMs belong to different security domains | Low | Medium | Low | | |
| **Scalability** | |  |  |  | | --- | --- | --- | | **vmware** | **kvm** | **xen** | | In the latest vmware, like vSphere 4.0 edition, the host machine can can support 320 vm guest systems, and 64 logical processor, 1TB physical memory, each guest systems can supports 8 CPUs and 256GB memory. It has the best scalability. | In the latest kvm edition, with the guest systems grows, some guest systems will be crashed. In the test, the kvm’s scalability cannot display good. In the three types virtual machine, it has the worst scalability. | As the guest systems increase, kernel compile time linear growth with the number of the client machines. And all the guest systems can well share the hardware resources. Xen has a good scalability. | | |

**4.2 Quantitative test**

**4.2.1 Baseline data**

For our initial set of tests, the experimental setup consisted of CentOs Linux 5.1 Intel 32 on the base machine. The Linux kernel 2.6.24-18-53.e18, Xen 3.2.1+2.6.24-18-53.e18-xen, KVM 62 were all installed from Centos packages and Vmware5.5 Workstation for Linux. All guests were automatically created by a benchvm script that called debootstrap and installed Centos5.1 Intel 32. The guests were then started with another benchvm script that passed the appropriate kernel (2.6.24-18-xen for Xen and 2.6.24-18 for KVM). The hardware system was a Dell OptiPlex 755 with a 2.33 GHz Intel Core 2CPU E6550, 3 GB of RAM, 160 GB hard drive, and one 100 M Ethernet cards.

**4.2.2 Overall performance**

**A. Application Test**

We compared the three Virtual Machines numbers against the non-virtualized (native) Linux baseline, shown below in Table 6:

Table 6 Summary of Application Test Results compared the three Virtual Machines numbers against the non-virtualized (native) Linux baseline. For the first and second column we define that number the bigger it is, the better. In the other columns ware opposite

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Linux | XEN | KVM | VMWARE |
| CPU | 1.00 | 0.99 | 0.94 | 0.96 |
| Memory | 1.00 | 0,71 | 0.51 | 0.68 |
| Fork | 1.00 | 1.13 | 3.38 | 1.08 |
| Gzip Compression | 1.00 | 2.38 | 1.59 | 1.81 |
| LAME Compilation | 1.00 | 2.18 | 6.91 | 3.91 |
| LAME Encoding | 1.00 | 1.02 | 1.11 | 1.10 |

**B. Disk Intensive Test**

For a disk intensive test, we chose not to write our own, but rather to use IOzone Benchmark specifically. IOzone is file system benchmark tool. The benchmark generates and measures a variety of file operations. IOzone has been ported to many machines and runs under many operating systems. We use IOzone each running an alternating read and write pattern (iozone -a -s 64m -i 0 -i 1) and change the Reclen size from 4KB to 16384 KB. The results of this test were quite interesting.

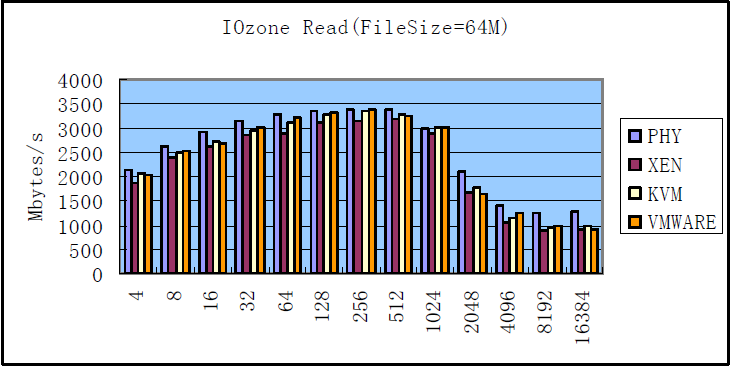


Figure 4 IOZone Read

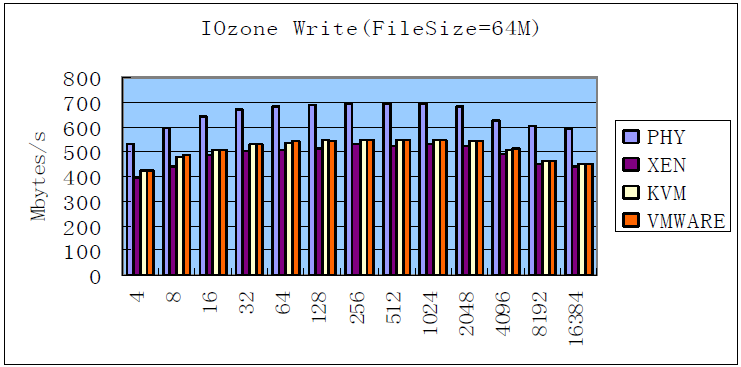


Figure 5 IOZone Write

In the figure 1 and figure 2, Kvm and Vmware had higher write and read performance than Xen according to our results. We believe that Kvm and Vmware may have performed better than Xen in terms of I/O due to disk caching, but they still ware more degradation than native Linux.

**C. Network I/O Intensive Test**

For the network I/O intensive test, we use Netperf as benchmark for the performance of systems .It is a benchmark that can be used to measure the performance of many different types of networking.

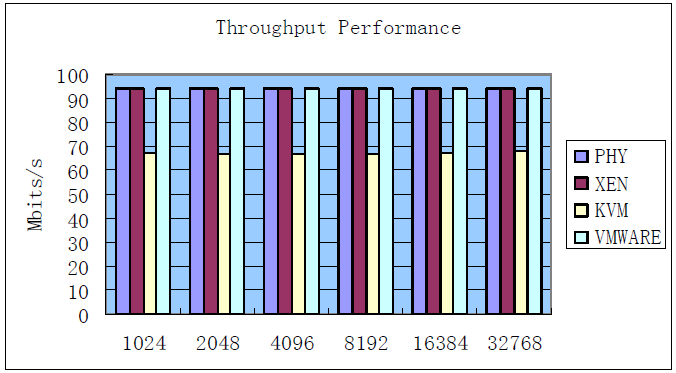


Figure 6 Netperf Throughput Performance

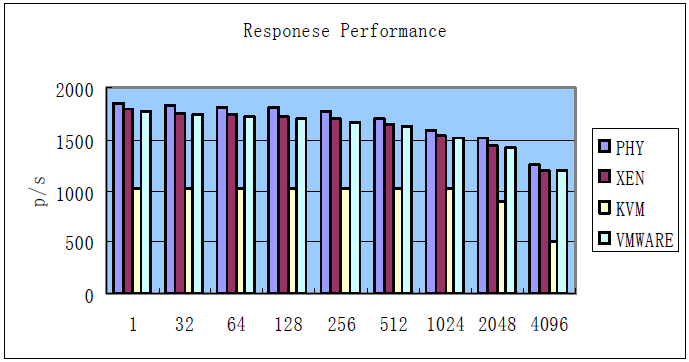


Figure 7 Netperf Responses Performance

We compared the three Virtual Machines of throughput and responses with native Linux in the graph 6 and 7. We can see the difference between Kvm and others. Kvm performance was lagged significantly behind the native Linux and other virtual machines.

**4.3 Performance isolation**

In this Comparison, we examine three virtualization environments–Xen, Kvm and VMware. On each, we ran a series of tests that stress a variety of system sources including CPU, memory, process creation, disk I/O and network I/O, we host one netperf as server and create four the same guests in their own virtual machine as client.

In CPU Intensive Test, we test stressed CPU usage with a tight loop containing both integer and floating point operations. In Memory Bomb test, we began with a stress test which simply loops constantly allocating and touching memory. In Fork Bomb test, we also ran a program that loops, creating new child

Processes: In Disk Intensive Test, as the disk intensive stress test, we use IOzone as before, we ran threads of IOzone each running an alternating read and. In Server Transmits Data test, as the server transmitting test, we started 4 threads which each constantly sent 60K sized packets over UDP to external receivers. write pattern (iozone -i 0 -i 1 -r 4 -s 64M -t 10). The test results are shown in the Table 7.

Table 7 Summary of stress test results

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | XEN | | KVM | | VMWARE | |
|  | Normal | Stress | Normal | Stress | Normal | Stress |
| CPU | 0 | 0.02 | 0 | 0.03 | 0 | 0 |
| MEMORY | 0 | DNR | 0 | DNR | 0 | 87.2 |
| FORK | 0.02 | DNR | 0 | DNR | 0 | DNR |
| DISK Intensive | 2.61 | 15.67 | 0.01 | 26.2 | 0 | 40.31 |
| NETWORK Sender | 0.02 | 0.43 | 0.14 | DNR | 0 | 48.73 |
| NETWORK Receiver | 0.03 | 0.12 | 0 | 0 | 0 | 0 |

Percent of degradation in good response rate: For each test, the percent degradation for either the bad or misbehaving VM is shown, as well as, the average degradation across the three good or well-behaving VMs .DNR indicates the Netperf client reported only an error and no actual results because of the unresponsiveness of the server it was testing.

**Scalability**

KVM: support virtual machines with up to 16 virtual CPUs and 256GB of ram and host systems with 256 cores and over 1TB of RAM.

Xen: 128 vcpus per guest, 1 TB of RAM per host, up to 1 TB of RAM per HVM guest or 512 GB of RAM per PV guest, 128 physical CPUs per host (as a default, can be compile-time increased to lots more).

VMWare:

