CLOUD INFRASTRUCTURE MATTERS

GORDON HAFF

EXECUTIVE SUMMARY

New IT technologies and abstractions, such as cloud computing, layer on top of existing ones. This may make the foundational layers less visible, but they remain just as important (or even more so).

The capabilities of a specific operating system aside, the operating system as a technology layer remains highly relevant in a cloud computing world. It insulates the application from lower-level technology (such as hardware) changes, provides a rich set of services to applications, and provides a consistent, long-lived set of interfaces to which applications can write.

Red Hat Enterprise Linux provides applications with a consistent, certified environment across physical servers, virtual servers, private clouds, and public clouds through both technical certifications and business relationships.

Red Hat Enterprise Virtualization is leveraging all of the open source collaborative development that goes into Linux to rapidly advance on many fronts. It's the first virtualization that's built from the ground up for multi-tenant security based in part on SELinux capabilities, a project originally developed by the National Security Agency to meet the most stringent security requirements.

Red Hat Enterprise Linux 6 expands on its rich heritage to add an impressive array of features that deliver even better performance, scalability, and quality of service controls. Such capabilities are increasingly needed to support new types of workloads, such as cloud computing environments, as well as new processors and servers that are extremely powerful by any historical standard.

These software capabilities come together with the latest generation of x86 processors that not only continue to add processing cores, but also make huge steps forward in reliability, availability, and serviceability.

CHECKLIST FOR INFRASTRUCTURE THAT MATTERS

- ☐ Workload isolation with mandatory access controls
- ☐ Application API portability across physical servers, virtual machines, private, and public clouds
- ☐ Enterprise support, testing, and certifications
- ☐ Quality of Service features
- ☐ Support for a broad set of scalable modern hardware
- ☐ Broad ISV support
- ☐ Headroom to grow storage, networks, and systems
- ☐ Open APIs and open standards
- ☐ Choice of hardware architectures, vendors, and hypervisors
- ☐ Broad development community
- ☐ Proven reliability and robustness

INTRODUCTION

Cloud computing is fundamentally about adding abstraction and then using that abstraction to simplify IT management and speed bringing new applications and business services online. If you think of virtualization as abstracting computer hardware, you can think of cloud computing as abstracting at the resource level. In other words, rather than dealing with servers, or disks, or even with virtual machines per se, you manage and provision CPU horsepower, gigabytes of memory, terabytes of disk, and so forth. With cloud computing you start thinking about capacity from a logical or architectural perspective, rather than in a way that's primarily an artifact of how the hardware happens to be physically built.

Thus, in a sense, the physical systems fade into the background. Indeed, this major up-leveling of the way we think about resources and the requirements of the workloads using those resources is very much at the heart of why cloud computing is of interest to so many people. It shifts the management of computing closer to things that are relevant from a top-level business perspective rather than things that are simply a byproduct of the way the infrastructure is assembled.

Of course, foundations are anything but unimportant. Ask any homeowner whose house is falling down or any driver whose car has a balky engine.

In this white paper, we focus on some key parts of the foundation that underpins cloud computing. These include the interfaces needed to provide applications with a consistent and portable environment even though they no longer live and die on a specific physical server or maybe even a specific datacenter. Quality-of-service and scalability features ensure that business applications get the performance they need even though they no longer run on dedicated hardware. In a world where physical separation of workloads is increasingly the exception rather than the norm, security that's built-in becomes increasingly important. In Red Hat's case, these components are part of the Red Hat Enterprise Linux operating system and the KVM-based Red Hat Enterprise Virtualization platform that's built into the Red Hat Enterprise Linux kernel—along with the JBoss Enterprise Middleware stack that sits on top of the operating system.

It's also about the hardware. Processors and other server hardware incorporate their own performance and reliability features while also providing various assists to the software running on top. In recent years, x86 servers have steadily upped their game to the point that they now incorporate a large swath of the sort of features that were not so long ago considered to be exclusively in the province of "Big Iron." This paper looks at some of the ways that volume processors have improved their error-handling ability.

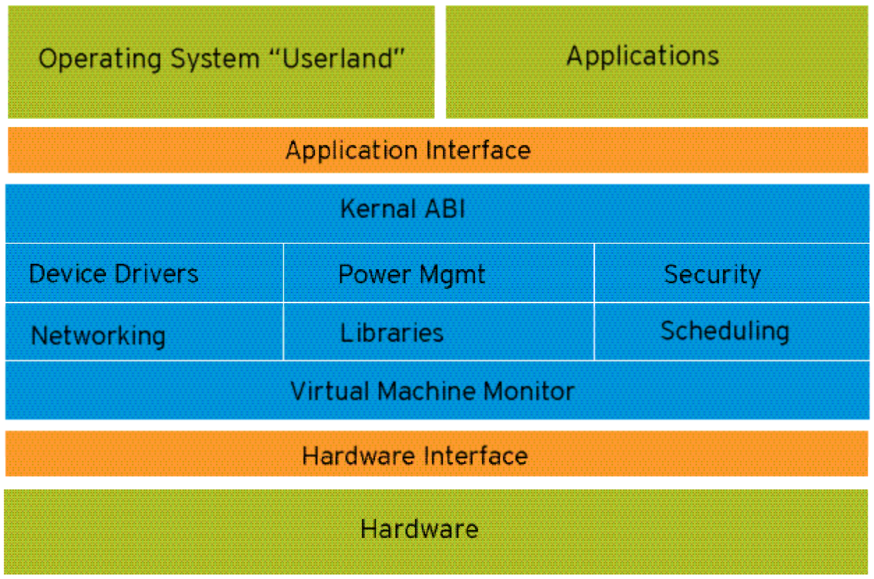
THE OPERATING SYSTEM INTERFACE

Because the operating system plays such a large part in this story, it's worth first stepping back and considering why we have operating systems anyway. After all, the first computers didn't even have operating systems. Users just ran programs that controlled the entire machine and explicitly specified how to perform just about every low-level task.

Operating systems have evolved to provide an increasingly rich set of services to applications so they don't have to reinvent the wheel. One important aspect of these services is that they stay relatively consistent and stable. Put another way, they constitute a sort of "contract" for applications that run on top. There's effectively a promise that if you write a program to the defined operating system interfaces things will still work even if you install a new type of storage device or if you double the number of processors.

They also essentially insulate applications making use of those services from the hardware and the low-level software that talks directly to hardware. That's a key value of abstraction. Done right, it allows innovation to occur in one layer of the hardware/software stack without forcing everything else to change.

Here's an example. A network card manufacturer switches to a new communications chip for better performance. This chip and card talk to the operating system through a device driver in a well-defined, largely standardized way. The change in this chip is typically transparent to the higher levels of the software stack. At most, it might require an update to the operating system's networking stack to take advantage of some new capability. But user applications? Only with truly radical changes to the communications architecture would they see any difference from having a new chip. (Which is why truly radical changes of this sort are rare indeed.)



Finally, operating systems handle basic processes and memory management so that application programs don't need to worry about where exactly on the physical server their instructions are executing, or on what disk drive, platter, and sector their data is stored. That may seem straightforward, but basic process scheduling on multi-processor systems and the many tasks handled by filesystems were once the responsibility of the programmer. Over time, operating systems have largely abstracted away those tasks.

To net it all out, the operating system has three primary tasks:

- Provide a robust and high-performance substrate for the applications and broader operating environment that runs on top;
- Offer abstractions and services that applications can make use of without having to re-create commonly used functions; and
- Create consistency for applications among hardware platforms—and, now, clouds.

CONSISTENCY OF ENVIRONMENT

We'll get to some of the underlying technical attributes of operating systems, but let's first drill down on consistency a bit more.

Fundamentally, it's the operating system's layer of abstraction that determines whether a given application can run on one type of hardware or many. Turn back the clock far enough, and applications were pretty much specific to a given generation of a given vendor's hardware because they did a lot of twiddling with registers and other low-level hardware components and those details varied from one computer model to another. Today, at least for a given instruction set architecture (ISA) such as x86, applications can mostly run across a broad range of system hardware. They write to the operating system's application programming interfaces (APIs) and the operating system provides consistency across underlying platforms.

In the case of Linux, these APIs are open source and they're based on open standards. They also run across a broad swath of computer architectures. This brings in a large community of developers and users and eliminates the possibility of being locked into any single vendor's API as was the historical norm.

Of course, any general statement about "works with X" comes with qualifiers. It may come down to: "It ought to work." And that's not the kind of reassurance the IT person charged with running production applications wants to hear.

Getting rid of "mostly" is, in part, a technical task that relates to maximizing backward compatibility between operating system versions. It involves testing and certifying against a wide range of hardware platforms. It involves certifying with the Independent Software Vendors (ISVs) who write the applications. In short, it's about getting from "ought to work" to "does work."

However, not to minimize the development and qualification work involved, certifications very much require business relationships as well. It's not just a matter of the operating system vendor saying that something works; the ISV has to buy off as well. When there is some problem, clear support responsibilities need to be laid out and escalation processes in place.

Cloud computing has introduced yet another certification wrinkle. What if, rather than just moving an application across systems in a datacenter, you want to move it into a public cloud? That's where an offering like Red Hat Cloud Access comes in. It lets qualified enterprise customers migrate their current subscriptions for use at select Red Hat Premier Certified Clouds. This gives customers the ability to make use of Red Hat support, relationships, and technology on certified clouds, while maintaining a consistent level of service and support across all certified deployment infrastructures with consistent and predictable pricing.

SECURE VIRTUALIZATION

Both the underlying hardware and the application software environment place greater demands on the operating system over time. There are many examples of this: multicore processors, faster networks, bigger data, demand for faster response times, and more.

One of the biggest changes is the ascendancy of server virtualization that, among other benefits, lets multiple workloads be consolidated onto a single physical server—thereby increasing server utilization and lowering costs. Because it breaks the bonds between applications and specific physical servers, virtualization is also a key foundational technology for cloud computing.

Clouds are dynamic and they're large-scale. This makes virtualization performance an important consideration. Red Hast Enterprise Virtualization excels in this regard. It was used for the first published results in the Standard Performance Evaluation Corporation's (SPEC) virtualization benchmark, SPECvirt_sc2010.[1]

One of the advantages of Red Hat Enterprise Virtualization is that it is a modern hypervisor based on Kernel-Based Virtual Machine (KVM) virtualization technology, which can be deployed either as the standalone bare metal hypervisor or as Red Hat Enterprise Linux installed as a hypervisor host and managed through Red Hat Enterprise Virtualization Manager for Servers. In either form, it's part of mainline Linux kernel development and thereby directly benefits from the work of thousands of developers worldwide who are tuning, hardening, and adding new capabilities to the kernel.

Security—especially relevant to cloud computing—illustrates this dynamic nicely.

Isolating workloads is always important in the context of virtualization. With many applications sharing a single physical server, it wouldn't do to have one virtual machine be able to inadvertently trample on another or to be able to peek into another virtual machine's contents. That would violate the principle of a virtual machine acting just like a physical one as far as its tenant was concerned.

When services are not virtualized, machines are physically separated. Any exploit is usually contained to the affected machine, with the obvious exception of network attacks. When services are grouped together in a virtualized environment, extra vulnerabilities emerge. For example, if there is a security flaw in the hypervisor that can be exploited by a guest instance, this guest may be able to not only attack the host, but also other guests running on that host. Attacks can therefore extend beyond the guest instance and could expose other guests to attack.

sVirt, which derives from Security-Enhanced Linux (SELinux), is an effort to isolate guests and limit their ability to launch further attacks if exploited.

SELinux is an implementation of a mandatory access control mechanism in the Linux kernel, checking for allowed operations after standard discretionary access controls (DAC) are checked. An example of DAC is standard Linux file permissions; the potential issue with DAC is that anyone with access can propagate information. It was created by the National Security Agency and can enforce rules on files and processes in a Linux system, and on their actions, based on defined policy.

1 http://www.spec.org/virt_sc2010/results/res2010q3/virt_sc2010-20100616-00013-perf.html

SELinux introduces a pluggable security framework for virtualized instances in its implementation of Mandatory Access Control (MAC). The sVirt framework allows guests and their resources to be uniquely labelled. Once labelled, rules can be applied which can reject access between different guests.

Like other services under the protection of SELinux, sVirt uses process-based mechanisms and restrictions to provide an extra layer of security over guest instances. It can:

- Control network access (IPtables)
- Control data and file access (SELinux)
- Secure remote access (IPsec, SSL, SSH)
- Sandboxing (Red Hat Enterprise Linux 6)
- Encrypt (dmcrypt and luks)
- Audit comprehensively and remotely (auditd)

Using sVirt, and all the other hardening built into Linux, Red Hat Enterprise Virtualization is the only virtualization that's built from the ground up for the multi-tenant security levels required for cloud computing.

PERFORMANCE AND QUALITY OF SERVICE

Performance-related attributes cover a broad swatch of attributes including efficiency, quality of service, and total scalability.

EFFICIENCY

The days when Linux could see big, broad speedups though single "magic bullet" features is long past. Like other proven, enterprise-class operating systems, the low-hanging fruit has been picked. Instead, we're left with tuning and overhauling in a myriad of different places, a task well-suited to the huge worldwide Linux development community.

We see examples of this in Red Hat Enterprise Linux 6 where the wide range of performance enhancements reach into just about every component of the platform. What are they? Here are a few:

- Per-LUN flush daemons that enable dramatic performance improvements for enterprise applications deployed on large I/O subsystems;
- Kernel multi-queue network device support is used to provide QoS-based transmission and virtualization performance improvements; and
- Ticketed locks provide spinlock acquisition fairness in large-scale SMP systems (typically NUMA systems with >16 CPUs).

Which is probably more than you wanted to know.

What these features and many more like them mean is that customers and application providers will achieve industry-leading performance across all aspects of the Red Hat Enterprise Linux 6 computing environment: processor, NUMA/SMP, networking, storage, filesystem, virtualization, system services, and applications.

QUALITY OF SERVICE

The quality of service (QoS) provided by an infrastructure is certainly partly determined by its overall performance and efficiency. However, it's also a function of control—the ability to prioritize specific workloads and allocate to them the resources they need to meet required service levels.

The new Control Group (cgroups) feature of Red Hat Enterprise Linux 6 offers a powerful way to allocate processor, memory, and I/O resources among applications and virtual guests. Cgroups provide a generic framework for plug-in controllers that manage resources such as memory, scheduling, CPUs, network traffic, and I/O.

Management of control groups in userspace is provided by libcgroup, enabling system administrators to create new control groups, start new processes in a specific control group or set control group parameters. All of the other tools in the Linux kernel require the basic notion of a grouping/partitioning of processes, with newly forked processes ending in the same group (cgroup) as their parent process. Control groups are therefore a kernel feature that allows for more precise resource control and allocation but do not require the use of specific tools to do so.

Cgroups become increasingly important as system sizes grow, by ensuring that high-priority tasks are not starved of resources by lower priority tasks. Cgroups gives customers fine grained control of resource utilization of physical and virtualized environments in terms of memory consumption, IO (storage and networking) utilization and process priority—enabling the establishment of policies that provide QoS guarantees.

This lets customers deploy flexible virtual and cloud environments, by easily mixing critical enterprise applications with low-priority background applications while ensuring that the resources needed by both are properly allocated. Features such a cgroups take Red Hat Enterprise Linux to the next level as the strategic platform choice for the very largest IT deployments.

SCALABILITY

Red Hat Enterprise Linux 6 scales to the largest systems on the market today with plenty of headroom for systems expected in the next decade. Tested and supported limits will grow in step with top-of-the-line hardware capabilities, while theoretical limits are exceptionally high. For example, for x86-64 systems, limits of up to 4,096 CPUs, 33,000 IRQs, 64TB of memory, 4 million processes, and 32,000 thread per process, give an idea of the capacity of the system. If some of those numbers seem well beyond the bounds of real-world needs, think back. Lots of limits that once seemed almost infinitely large came to be constraints after not that many years. So limits like these are needed to provide headroom for the foreseeable future. Scalability extends into the storage arena as well. The next-generation Ext filesystem, Ext4, is the default filesystem for Red Hat Enterprise Linux 6. Ext4 combines the stability of Ext3 with significant scalability (up to 16TB) and performance enhancements. The optional XFS filesystem is available for customers deploying even larger, specialized environments with high-end servers and storage arrays. Larger and faster filesystems—combined with online, dynamic storage management that eliminates reconfiguration downtime—are all part of the Red Hat Enterprise Linux scalability story.

And scalability features are not restricted to physical systems: Red Hat Enterprise Linux 6 also provides industry-leading scalability of virtual guests. Physical, virtual, and cloud deployments can scale to meet business requirements, eliminating the need to switch platforms when an existing platform reaches its maximum capacity. And customers with smaller configurations can be confident in the knowledge that the platform has been engineered to meet the requirements of the world's largest systems.

ROBUST HARDWARE

Volume x86 servers are running larger and more important workloads and those workloads need to be protected from failures more than ever. Of course, the operating system, virtualization software, and the applications themselves bear a lot of the responsibility for maintaining uptime. But the processor has responsibilities as well. And x86 processor vendors have been busy beefing up their products.

Take the Intel Xeon 7500 (a.k.a. "Nehalem-EX") for example; Intel counts some 20 new reliability, availability, and serviceability (RAS) features. Some are features of silicon and don't require explicit software support. Others need help from the operating system, which Red Hat Enterprise Linux has been quick to incorporate.

Machine Check Architecture (MCA) Recovery provides a good example of this cooperation. MCA Recovery is a mechanism whereby the silicon works with the operating system to allow a server to recover from uncorrectable memory errors that would have caused a system crash in prior generations.

In this first implementation on Xeon processors, MCA Recovery allows the operating system to recover when uncorrectable errors are discovered during either an explicit write-back operation from cache, or by a memory patrol scrub that examines every server memory location daily. When an uncorrectable error is detected, the silicon interrupts the operating system and passes it the address of the memory error. The operating system then determines how best to recover from the error and continue operation of the system. The operating system marks the defective memory location so it will not be used again and resets the error condition; the system can keep running in many cases. In Red Hat Enterprise Linux 6, the notification is interrupt-based, rather than based on polling, for faster handling.

Just as Linux now has a multitude of capabilities that were once the province of expensive proprietary systems (and, indeed, moved beyond them in many cases), so too do processors like Xeon.

CONCLUSION

Modern computing wouldn't be possible without abstractions; computing would just be too complicated. That's why the adding of abstractions in an ongoing process that's essentially been the history of the computer industry. Cloud computing is just the latest chapter, albeit an important one.

It's natural to start thinking that "not visible" means "not important." Of course, foundations are important. Indeed, as we pile abstraction atop abstraction, it becomes increasingly important that the underlying foundation just works without the need for a lot of active monitoring and manipulation. And it's a foundation that has the characteristics needed to support what you're building on top.

DLT SOLUTIONS

The Red Hat Team
877.742.8358
www.dlt.com/redhat
redhat@dlt.com