N1 Grid Containers: Server Virtualization and Manageability

A Technical Brief for IT Executives and Managers

May 2004

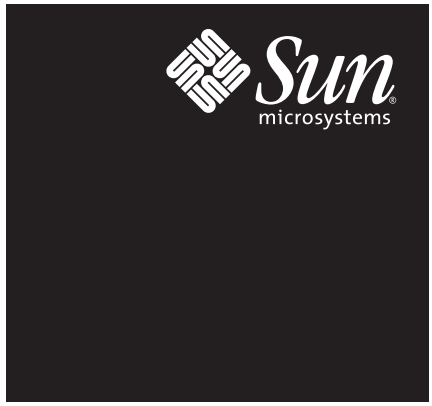


Table of Contents

Chapter 1

Executive Overview

After nearly a decade of experience with large-scale distributed computing, businesses have begun to understand how to build scalability and availability into their information technology (IT) infrastructure. Now, the escalating costs of managing vast networks of servers with software components installed on thousands of nodes have shifted the focus from increasing performance and availability to reducing the cost of IT infrastructure, increasing utilization of hardware, and better managing end-user service levels. Sun's vision is that businesses can accomplish this through N1™ Grid, a new approach for managing data centers. N1 Grid views a data center as a single pool of resources delivering business services. Business services may be run on any system within the pool that provides the resources needed to meet performance and service levels. Creating and managing this single pool of resources requires *server virtualization*, a concept that allows servers and the network to be flexibly partitioned into independent execution environments within the same server and physical network.

Business services often consist of software components that are distributed across multiple servers and operating system instances. To reduce costs, businesses are eager to consolidate these applications onto fewer servers, but are concerned about unexpected interactions between applications. Virtualization enables the management of data centers as a fabric of interconnected computing resources rather than as a room filled with individual systems.

N1 Grid™ Containers are a key element in the implementation of N1 and server virtualization, and provide flexible, software-defined boundaries for isolating software applications or services. These applications can then be managed independently of each other, even while running in the same instance of the Solaris™ Operating System.

N1 Grid Containers create an execution environment within a single instance of the Solaris Operating System (Solaris OS). As part of the Container roadmap *Solaris™ Containers* were

introduced with Solaris 9 OS to provide full resource containment and control for more predictable service levels. The next release of the Container technology, n1 Grid Containers, will also include fault isolation to minimize fault propagation and unplanned downtime, and security isolation to prevent unauthorized access as well as unintentional errors.

The Container environment also enhances resource usage accounting so that granular and extensive tracking of resources can support capacity planning and advanced client billing models.

Sun plans to have N1 Grid Containers as the fundamental, ubiquitous management object in the Solaris Operating System, to be used throughout Sun's SPARC and x86 products. This common approach will simplify service provisioning and make it easier to consolidate applications onto fewer servers, while addressing concerns about resource constraints, fault propagation, and security. The ubiquitous management model also simplifies service level management by permitting end-to-end services to be managed as a single unified object. The service management environment will be flexible and dynamic, and yet easy to understand and manage.

The primary benefits of N1 Grid Containers are:

- Reduced management costs through server consolidation, reduced numbers of OS instances, and end-to-end service level management
- Increased resource utilization with dynamic resource reallocation between containers
- Improved service availability by minimizing fault propagation and security violations between applications
- Increased flexibility because these software-based containers can be dynamically reconfigured
- Increased accuracy and flexibility in billing based on workloads rather than hardware or processes

Businesses can realize a significant reduction in total cost of ownership without sacrificing service levels through this intuitive, flexible management model.

Intended Audience

This paper describes an approach to resource management in a distributed computing environment. The intended audience is IT managers, architects, and executives in organizations that are implementing large-scale, distributed computing solutions. Chapter 3 discusses the concepts and key features of Solaris Containers. Chapter 4 illustrates how Containers may be used to implement various resource management scenarios. Chapter 5 takes a look at upcoming developments in Container technology. Chapter 6 provides a brief conclusion, and Chapter 7 provides references for further information on the software and technologies discussed in this paper.

Chapter 2

Introduction

As IT systems have evolved over the years, so have the expectations of both end users and IT staff. With each advancement in technology, previous breakthroughs are viewed as standard features that are soon taken for granted and provide a foundation for new developments. Early information systems were judged a success if they simply solved a business problem. It didn't matter if it required a team of programmers to maintain the system. As long as the business benefits could justify the cost, the systems were considered good investments.

As IT infrastructure matured, more users were given direct access to systems, and managers began to depend heavily on computing to conduct daily operations. Availability and scalability became key concerns of IT managers, and the net benefit of building availability and scalability into systems was measured by counting the lost productivity from either downtime or poor performance. Now, methods of implementing availability and scalability are becoming well understood. By replicating key software processes across a multi-tiered architecture of servers, both availability and scalability can be incrementally expanded.

Recently, the focus has been on obtaining additional availability or scalability in the most efficient way possible, with users taking for granted that their IT services will deliver adequate performance and always be available. However, when more components are replicated throughout an IT architecture to give it greater resiliency and throughput, the result is a sprawling, complex network of systems that are costly and difficult to manage.

Businesses can decrease costs by improving resource utilization and lowering system management costs. Capacity planning generally includes extra capacity to handle occasional peak loads. For example, brokerage firms often design their systems to handle trading volumes far above those that have ever been experienced in order to capture as much revenue as possible

if a spike in demand occurs. This means that normal trading activity levels may only require a small fraction of the total available capacity.

By allowing other applications to borrow resources from the trading systems when those resources are not being utilized, a much more cost-effective implementation can be realized. In the event of a spike in trading volume, resources would be dynamically reallocated so that the less-critical applications might experience delays, but the primary trading system would function as needed. Sharing resources in this way improves resource utilization, reduces total capital costs, and lowers system management expenses by reducing the total number of systems to be managed. The struggle to make IT operations more efficient has thus turned the focus of IT managers to manageability and utilization of hardware and software.

In order for server consolidation to be truly effective, businesses must still be able to manage each application independently. This requires the ability to control resource utilization, isolate faults, and manage security for multiple applications on the same server. In other words, it requires the establishment of virtual server boundaries within the server.

Sun's Commitment

Sun understands that increasing IT efficiency and improving hardware utilization is key to the success of enterprises. Through its commitment to networking, client-server computing, UNIX™, and open systems and standards, Sun consistently delivers robust, flexible, and powerful network technologies, high-performance servers, and systems that meet the needs of a wide range of organizations. By providing technologies such as N1 Grid Containers which virtualize resources, Sun enables safer, easier, more manageable servers and infrastructure, provides extensive, flexible resource management facilities, and helps enterprises balance a complex set of needs.

Chapter 3

Server Virtualization

The lesson learned from mainframes years ago was that if an enterprise has application services with varying peaks and valleys of use, it is more efficient to aggregate these applications together and share resources. Through such consolidation, resources can be reallocated on the fly instead of having dedicated, but infrequently used, hardware to support each of these application services. Using this approach, resources can be logically moved when and where they are required. If done correctly, businesses can reduce the total expense of their computing infrastructure through reductions in the cost of capital equipment, maintenance, licensing fees, and system management.

Mainframes were also the first to divide server resources into partitions to isolate applications from one another. This enabled customers to protect production environments while system updates or revised applications were tested in an isolated workspace that could not affect the production system. In 1996, Sun took this idea one step further when it introduced Dynamic System Domains technology into the open systems arena.

With Sun's Dynamic System Domains, each domain runs its own copy of the Solaris Operating System and provides fault isolation so that system or application errors cannot impact applications running in other domains. A key difference between these domains and the early mainframe logical partitions technology is that domains can be dynamically partitioned. In the event of a resource shortage in one domain, the system can automatically borrow resources from another domain. This adds great flexibility to the data center while maintaining security and isolation from faults in other domains.

Dynamic System Domains virtualize individual servers. Now, N1 Grid is going to virtualize an entire data center into a single pool of resources.

N1 Grid: Sun's Vision for the Data Center

N1 Grid is Sun's vision, architecture, and products for reducing the complexity of enterprise data centers by making entire data centers appear as a single pool of resources rather than a collection of separate systems, software, and storage. Today, each application in a data center typically has its own resources, with its own excess capacity to meet peak demand times. Applications have not been designed to share resources, resulting in poor overall utilization of hardware resources and adversely impacting Total Cost of Ownership (TCO). With N1 Grid, data center resources may be managed as a single system delivering business services, putting idle resources to use, improving manageability, and decreasing TCO.

A business service typically spans multiple tiers. N1 Grid will enable the administrator to describe the dependencies and relationships between the different software components that comprise a business service or workload. N1 Grid will then manage the end-to-end availability of the business service as a whole. By allowing IT staff to manage services rather than the individual components that make up those services, N1 Grid will reduce management complexity and increase hardware utilization, thereby reducing operating expenses and lowering TCO for the organization.

In this paper, the terms "business service", "workload" or "application" may be used interchangeably to mean a collection of software components that provide a specific business function such as Enterprise Resource Planning (ERP) or financial management, for example. These software components run in separate processes, and may also be run on separate systems.

In N1 Grid, the process of mapping business services onto resources is known as provisioning. N1 Grid's services provisioning will automate software and hardware installation and configuration, and allow for faster, more flexible deployment of services with lower risk of manual errors. N1 Grid will dynamically provision resources for business services, borrowing resources as needed to meet peak demand. As demand drops off, resources will be released for use by other services, an ability not provided by current capacity-on-demand solutions. By dynamically managing resource allocations based on pre-defined business policy, N1 Grid will maintain consistent service levels and allow IT staff to better meet service level agreements and users' expectations without over-provisioning each business service to meet peak demand. Automating error-prone manual tasks will allow IT staff to focus on business needs, and more rapidly deploy business services to meet those needs.

Managing dynamically allocated resources at the level of the business service requires accounting on a per-service basis, not just on a per-system or per-process basis. N1 Grid will provide for per-service accounting, enabling accurate, usage-based billing and management reporting. The customer will have visibility into the actual costs per business service and will be able to make adjustments based on business objectives, enabling resources to be logically provisioned and reconfigured as often as needed to meet rapidly changing business needs.

N1 Grid Containers: Advancing Server Virtualization

Sun has taken the server virtualization concept one step further by allowing servers and domains to be partitioned to sub-CPU granularity using N1 Grid Containers, a key element in the N1 Grid approach to managing IT infrastructure. An N1 Grid Container defines a complete execution environment for a set of software components, and provides a virtual mapping from the software components to the platform resources. The N1 Grid Container allows application components to be isolated from each other while sharing a single instance of the Solaris OS as well as physical resources. N1 Grid Containers offer execution environments whose definitions are transportable from one server partition to another with minimal management overhead, enabling great flexibility in provisioning. The isolation, transportability, and resource sharing enable easier consolidation of applications onto fewer servers and OS instances.

N1 Grid Containers establish boundaries for consumption of resources such as memory or CPU time, and will provide fault and security isolation in a future Solaris release. As processing

requirements change (for example, an unexpected world event occurs and causes a surge in hits against a news-oriented Web site), one or more of the boundaries of the Container can be expanded to accommodate a spike in resource consumption. Fault and security boundaries are maintained when resource boundaries are updated whether the update is done by an administrator or through predefined policies that result in automatic updates when certain conditions are met. The next chapter provides some examples of how Containers can be used.

Chapter 4

Containers At Work

With the release of the Solaris 9 Operating System, Containers were first released as Solaris Containers with many of the features necessary to create resource boundaries. These provide the ability to allocate resources such as CPU, physical memory, and network bandwidth within a single instance of the Solaris OS, and allow enterprises to take advantage of the benefits of Containers illustrated in the following examples.

Example: Managing CPU Resources

Many enterprises depend on multiple instances of a database, for example Oracle, which underlie critical applications such as Enterprise Resource Planning (ERP) systems. Each database instance and front-end application typically resides on its own server, with its own dedicated CPU, memory, and storage, sized to meet peak demand times. In this example, a large, multi-national enterprise has separate database instances, in a typical 3-tier architecture, for its European, Asian, and American operations. A simple view of this configuration is shown in Figure 4-1. Each of these systems is heavily used during the business hours for its operation's geography, but generally sees low usage outside of those hours, resulting in a significant investment in resources that are underutilized on each of the servers.

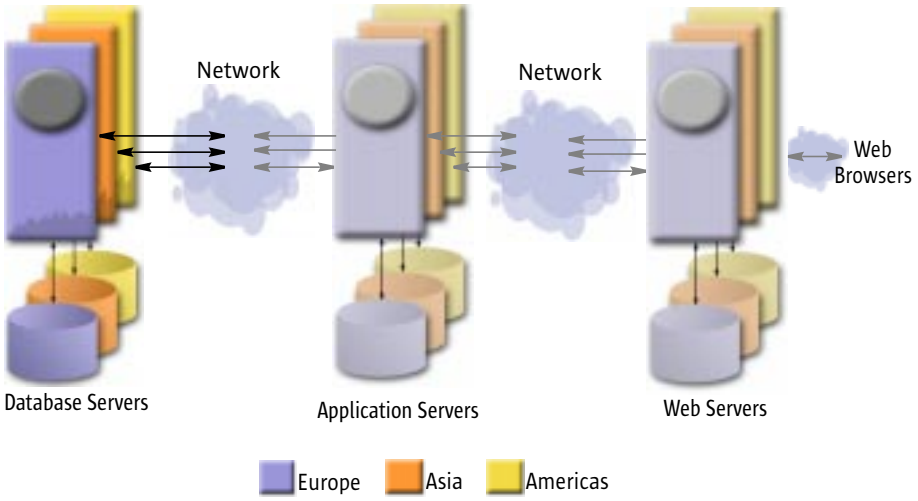


Figure 4-1: Traditional Resource Management using Separate Systems

Database software tends to be more mature and thus more stable and reliable than newer types of software systems. Database systems generally request enough physical memory to ensure that their processes will remain in memory, and avoid the impact on performance that results from data being swapped to disk. In this example, therefore, memory is assumed to be adequate and memory resource management is not a concern. The requirement is to manage CPU usage and ensure that each database instance receives the required resources during peak demand times. To manage CPU resources, the administrator can create a Container for each database instance, and provision those Containers on a single database server, as shown in Figure 4-2. The CPU resources are managed using, a specific piece of the Container technology, Solaris 9 Resource Manager.

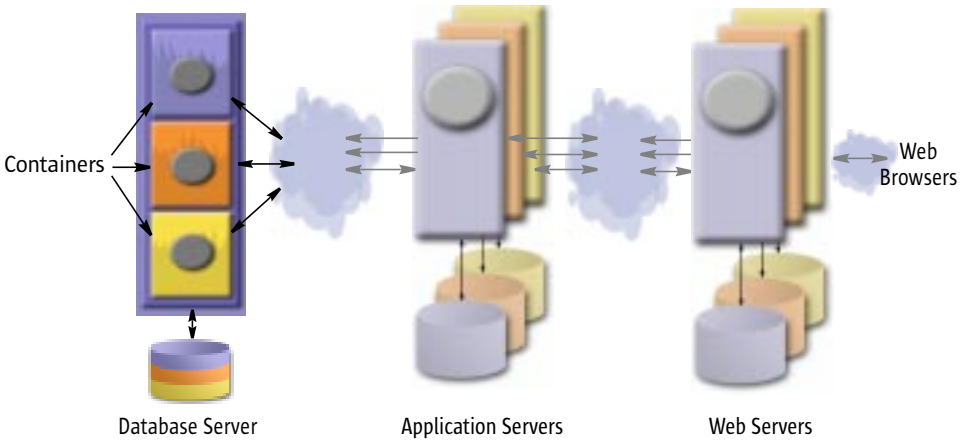


Figure 4-2: Database Servers Consolidated into Separate Containers

Solaris 9 Resource Manager

Solaris 9 Resource Manager provides two significant capabilities for managing CPU resources. These are resource pools and the Fair Share Scheduler (FSS).

Resource Pools

Solaris 9 Resource Manager uses *resource pools* to control system resources. Each resource pool may contain a collection of resources, known as *resource sets,* which may include CPUs, physical memory, or swap space. A *processor set* generally is assigned a subset of the total number of

CPUs available on the system or in a Dynamic System Domain. For example, on a four CPU system, an administrator could define two resource pools, each containing a processor set, with pool A's processor set containing one CPU and pool B's processor set containing the remaining three CPUs. Resources can be dynamically moved between resource pools as needed.

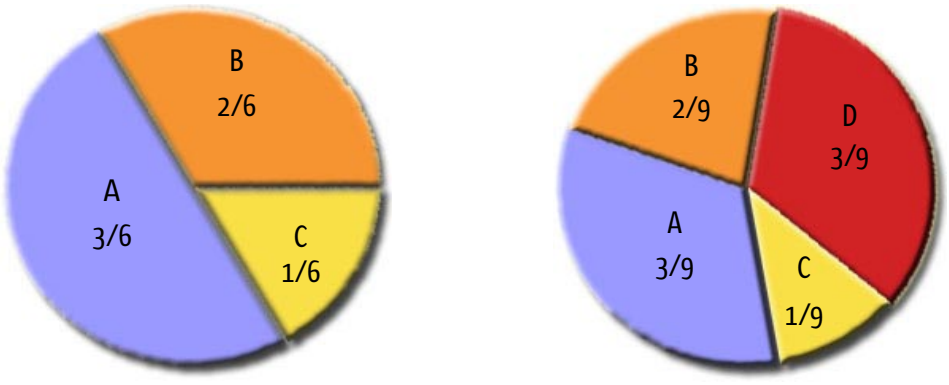
In order to assure the performance of a critical workload, it can be assigned to pool A by itself to avoid contention for resources with other workloads. All other workloads running on that system are assigned to pool B, and share the three processors assigned to pool B's processor set. The use of processor sets in effect establishes the maximum CPU resources that may be consumed by a workload. The workload assigned to resource pool A, while it does not share the processor set, nevertheless has only one CPU available. The workloads running on resource pool B will contend for the CPU resources in pool B's processor set. This contention is managed through the default Solaris timesharing scheduler, or it may be managed with the Fair Share Scheduler.

Fair Share Scheduler

Solaris 9 Resource Manager incorporates an enhanced Fair Share Scheduler (FSS) which may be used within a resource pool. When using the FSS, an administrator assigns CPU *shares* to a workload which may be composed of one or more processes. Shares allow the administrator to specify the relative importance of one workload to another, and the FSS translates that into the ratio of CPU resources reserved for a workload. If the workload does not request CPU resources, those resources may be used by other workloads. The assignment of shares to a workload effectively establishes a minimum reservation of CPU resources.

For example, three workloads share a processor set using FSS. Workload A is assigned three shares, workload B is assigned two shares, and workload C is assigned one share, for a total of six shares. FSS allocates the CPU resources using the assigned shares. Workload A receives 3/6 of the available CPU resources, B receives 2/6, and C receives 1/6. When a fourth workload, D, is added and assigned three shares, the ratios change to 3/9, 2/9, 1/9 and 3/9 for each workload respectively, as illustrated in Figure 4-3. The relative importance of each workload remains the same.

Figure 4-3: Fair Share Scheduler CPU Share Assignments



Original CPU Share Assignments

CPU Share Assignments After Adding An Additional Workload

By setting minimum reservations (shares) and maximums (processor set resources) for each workload, the service level needs for each operation can be flexibly managed, while providing a consistent level of service, and meeting peak demand as needed.

In an environment where many workloads are sharing resources, enterprises will need to accurately allocate costs to the various business units using those resources, as well as generate

accounting information for resource utilization analysis and capacity planning. With Solaris 9 Resource Manager, extended accounting and reporting features are available that allow for tracking resource usage by business service workload, not just by process. Chapter 7 lists references for further information on Solaris 9 Resource Manager.

Using Containers and Dynamic System Domains

Prior to the availability of Containers, much of the consolidation described in Figure 4-2 could have been achieved with each database instance running in its own Dynamic System Domain, with its own copy of the Solaris OS, on a single server. Using Containers with Dynamic System Domains reduces the number of Solaris OS copies to be managed, resulting in a reduction in the system management workload, and allowing for flexible reconfiguration of resources between Containers.

The enterprise in this example can go a step further, using Containers along with Dynamic System Domains. The production database instances can be consolidated into Solaris Containers within a single Dynamic System Domain running the Solaris OS release required for the production release of the database software. A second Domain provides an isolated environment to test a newer release of the database software which requires a different Solaris OS release. This scenario, illustrated in Figure 4-4, minimizes the number of operating system copies, isolates the production and test environments, and allows the hardware in the test domain to be reallocated to the production domain if needed.

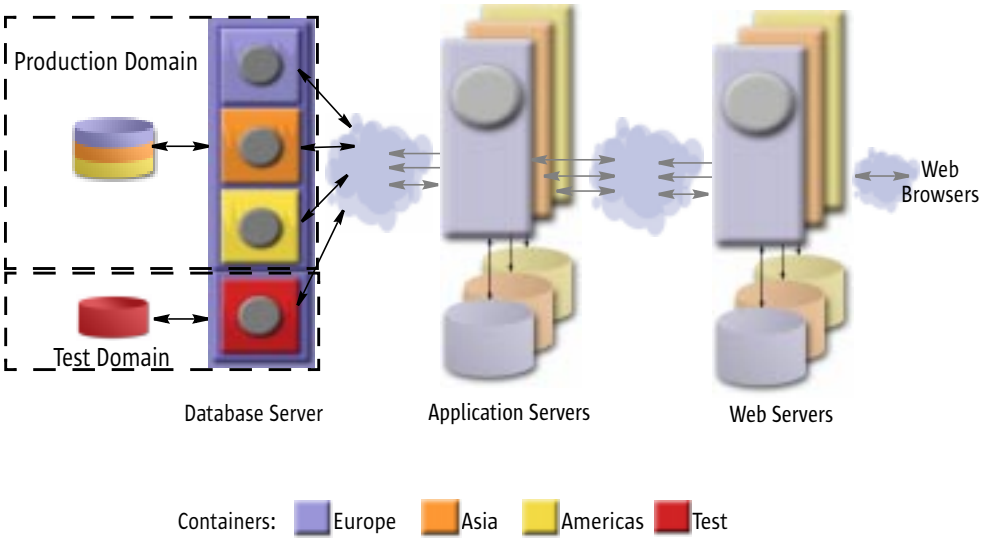


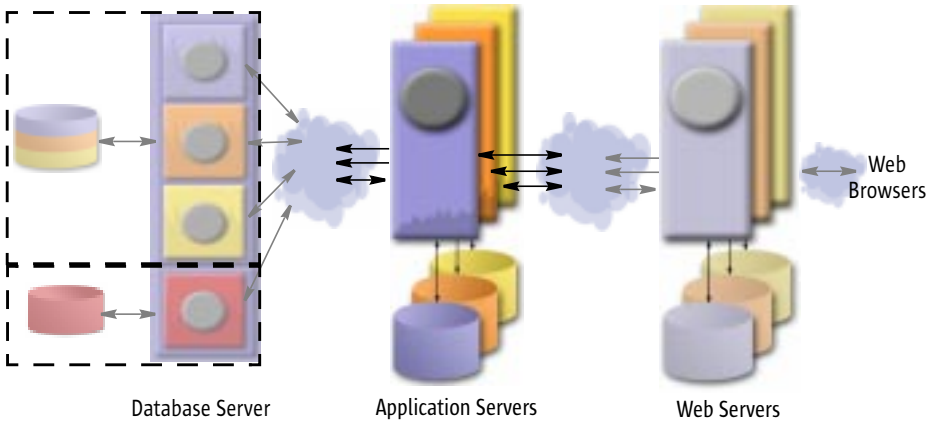
Figure 4-4: Database Servers Consolidated into Separate Containers within Dynamic System Domains

Because the peak utilization times are different for each geography, the production domain can be configured as if for a single database instance. This configuration achieves a significant improvement in hardware utilization and a corresponding reduction in the number of OS instances that must be managed, while maintaining performance levels and service availability.

Example: Managing Memory and Network Bandwidth

Many enterprises depend on multiple copies of application server software to support their critical business applications such as financial management applications, customer management, and many others. A configuration typical of this environment is depicted in Figure 4-5, which shows each component hosted on its own server.

Figure 4-5: Application Servers on Separate Systems



This example focuses only on the application server components and how they might be consolidated, and how memory and network resources can be managed as well as CPU resources.

Memory Resource Management

Because the business services provided by these application servers are critical to the running of the enterprise, it is extremely important that service levels are maintained and that performance meets expectations. Because most application server software is comparatively new, it may be somewhat less stable than software packages, such as database management systems, that have been through more revisions. Application server memory usage may grow over time, or grow as the volume of users increases, or both. For these reasons, it may be prudent to place a cap on the amount of memory each application server can use, to prevent excess memory usage by one application server workload from impacting the other workloads. Solaris 9 Resource Manager provides the interface for managing memory resources.

Network Bandwidth Management

The performance of application server software also frequently depends on available network bandwidth in addition to memory resources. Network resources can be managed through Solaris 9 Resource Manager's network bandwidth management facilities. Also known as Solaris 9 IP Quality of Service (IPQoS), this network management technology is fully integrated with the Solaris 9 OS. It provides the ability to classify and prioritize network traffic in accordance with the Internet Engineering Task Force (IETF) Differentiated Services (DiffServ) and IEEE 802.1d standards. These standards allow for the classification of network traffic on the basis of which application is generating the traffic, allowing the system administrator to more easily assign priorities. Network traffic limitations are handled either by the Solaris system or by a DiffServ- and 802.1d-compliant switch, and in a way that provides predictable performance levels for critical applications.

Multiple application servers can be consolidated onto a single system, as shown in Figure 4-6, with IPQoS used to manage network resources.

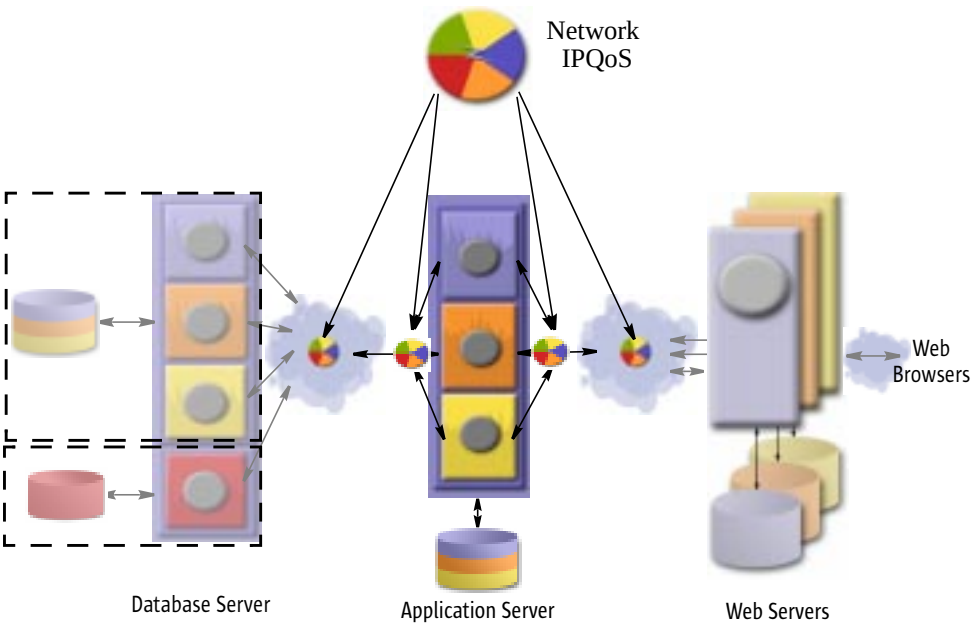


Figure 4-6: Application Servers Consolidated into Separate Containers

In this example, Solaris 9 IPQoS on the server provides the ability to classify and prioritize network traffic, while a DiffServ- and 802.1d-compliant switch provides the same capabilities on the network.

Containers provide many of the features needed to create resource boundaries and flexibly manage CPU, memory, and network bandwidth. The ability to create multiple Containers within a single instance of the Solaris OS allows enterprises to easily reconfigure resources to meet rapidly changing business needs.

Chapter 5

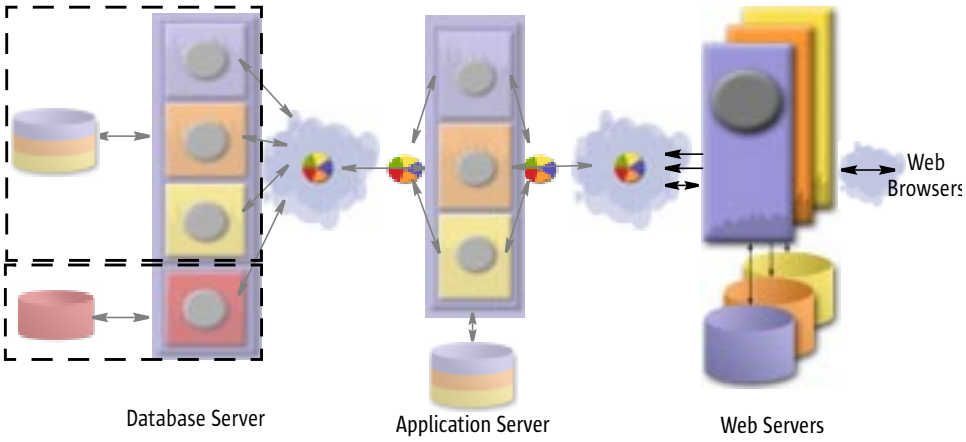
The Evolution of Containers

Sun's intention is to integrate the Container concept throughout its software and system management product lines and to provide enhanced fault isolation and security isolation in the next release of the Solaris OS, called N1 Grid Containers. The following example illustrates the usefulness of the next phase of fault isolation and security isolation.

Example: Fault Isolation and Security Isolation

A typical enterprise may provide a variety of services via external websites. These services could include an employee portal for access to internal networks, a Web store for selling products to customers, and a vendor portal for accessing supply chain applications. Each of these services has common elements, but must be isolated from one another to manage security and prevent errors caused by one service from affecting other services. Historically, isolation has been achieved by hosting these services in their own hardware and software environments. Figure 5-1 illustrates this environment. The database and application server pieces of the 3-tier architecture are already isolated on their servers, as shown in previous figures. This example addresses the Web server segment of the architecture.

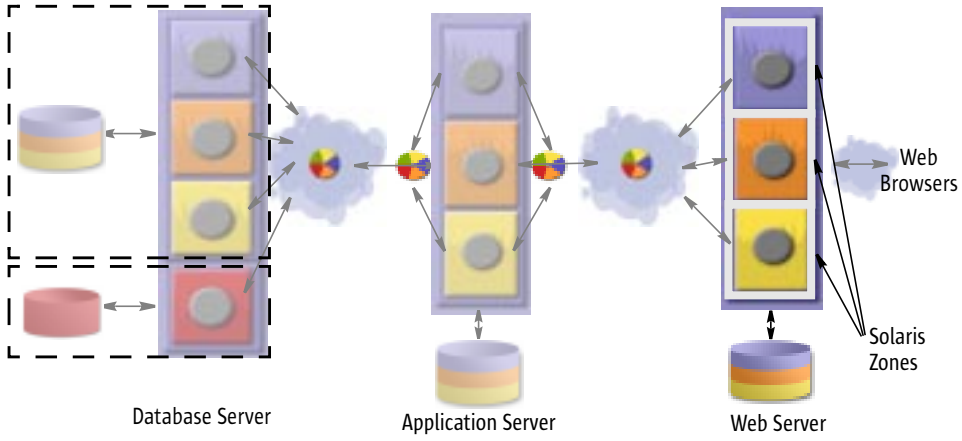
Figure 5-1: Web Servers on Separate Systems



Solaris Zones

In the near future, with the introduction of N1 Grid Containers, Sun intends to provide a way to virtualize the OS. N1 Grid Containers will provide the ability to run multiple virtualized OS instances on a single instance of the real OS, and will offer enhanced security and fault isolation in addition to the flexible resource boundaries offered by Containers in Solaris 9. The specific piece of the Container technology that provides this virualiztion is called *Solaris Zones*. These Solaris Zones will allow for safer consolidation of business services by providing separate name space, including its own IP addresses, filesystems, unique root user password and password file, name server, and so on. Communication between Solaris Zones will use the same standard interfaces, such as network connections and shared filesystems, used in communication between systems. Errors in a Solaris Zone that would normally cause a server to reboot instead affect only the Solaris Zone, which can restart, automatically or manually, without affecting any other Solaris Zones, and without the overhead associated with a hardware reboot. A problem in one Solaris Zone that requires a restart affects only its workloads and not others, and will thereby increase the overall service levels. The ability to restart a single Solaris Zone also addresses a significant problem in server consolidation: applications that were designed to run by themselves on their own hardware are now able to share hardware, increasing overall hardware utilization and lowering TCO for the enterprise. Figure 5-2 depicts consolidated environments using Solaris Zones.

Figure 5-2: Web Servers Consolidated into N1 Grid Containers

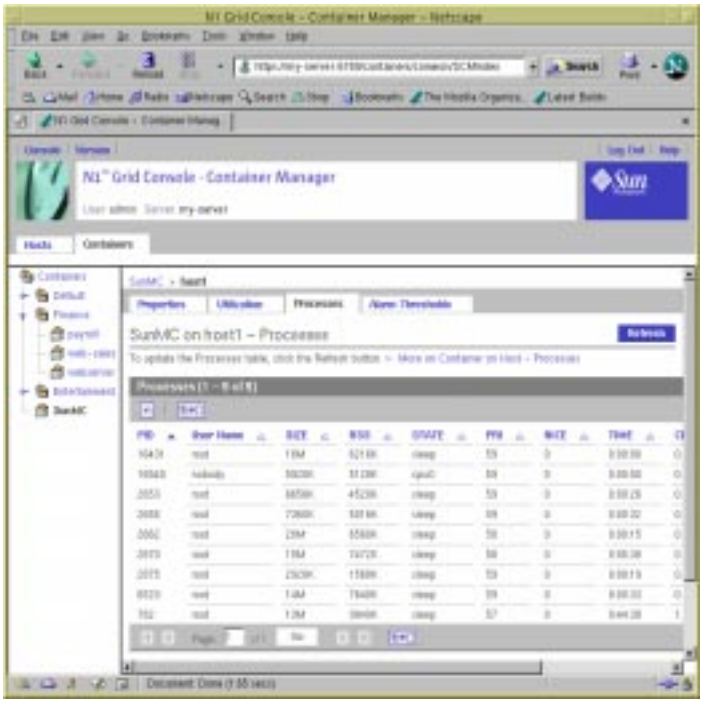


Because Solaris Zones will provide a complete virtual OS, Solaris Zones will appear to be completely separate systems from a user's perspective. In order for a user to access a Solaris Zone, the user must login, just as would be done with a separate system. Thus, workloads in a Solaris Zone will be protected from common manual errors, such as unintentionally shutting down all databases instead of only a specific database instance. This separation also applies to common operating system tools. When used in a Solaris Zone, tools such as `ps`, which normally lists all processes running on a system, will list only processes running in the Solaris Zone, assuring the same level of security that would be expected between systems.

Other Applications and Features

In addition to an integrated command language interface, Sun will offer a system management application based on Sun Management Center software. The enhanced system management application will provide a high-level service management facility that automates many of the Container management tasks and removes the administrator from management details. The first release of this application, called N1™ Grid Console - Container Manager 1.0 and shown in Figure 5-3, is out now and supports the resource management parts of the Containers in the Solaris 9 OS and that can be added to the Solaris 8 OS by purchasing Solaris Resource Manager 1.3. Sun's intent is that the next release of the N1 Grid Console will provide the management tools necessary to manage N1 Grid Containers so that these details will be transparent to the administrator, and easily managed within a graphical tool.

Figure 5-3: N1 Grid Console - Container Manager



Additional features are planned for Sun's resource management tools, such as the ability to establish memory sets and swap sets, along the same lines as processor sets in Solaris 9 Resource Manager. Also planned are dynamic resource pools, which will provide the ability to specify an action to take when a particular event occurs. This could include such events as:

- In case of a CPU failure, reallocate a CPU from a certain other processor set
- When a CPU board is added to the system, where and how to allocate the new resources
- In the event of a cluster failover, how to change the resource allocations

Sun intends to enhance all of its products to support this new management paradigm while continuing to provide application programming interfaces (APIs) that promote open standards.

Chapter 6

Conclusion

In order to prosper, businesses must not only maintain high service levels to satisfy end users, but also must reduce their IT expenditures. These seemingly opposing objectives require being able to intelligently manage infrastructure and systems. N1 Grid and N1 Grid Containers provide a leap forward in manageability that will enable businesses to reduce costs through improved efficiency and better resource utilization. This new approach will enable cost savings while continuing to meet the increasing demands of end users for high service levels.

N1 Grid Containers can have a significant impact on total cost of ownership for IT systems. They enable businesses to better manage large-scale systems through consolidation without concern about resources, faults, or security. The primary benefits of N1 Grid Containers include:

- *Reduced management costs* – Applications can be more safely consolidated onto fewer servers, resulting in reduced management complexity and reduced management costs through reductions in OS instances and associated software licensing fees. Further, standardization on the N1 Grid Container model throughout the Sun product line simplifies service provisioning, and reduces training and reconfiguration expenses.
- *Increased resource utilization for lower cost of operations* – Dynamic resource reallocation improves resource utilization by allowing unused resources to be shifted to other containers as needed. In addition, with fault and security isolation provided by N1 Grid Containers, poorly behaved applications no longer require a dedicated and underutilized system, but may be consolidated with other applications.
- *Improved service availability* – Fault isolation and security isolation protect applications from error propagation as well as intentional or unintentional intrusions that can affect perfor-

mance or availability. The greater consistency and simplicity of this environment also reduces the chance of human errors that might impact availability.

- *Increased flexibility* – N1 Grid Containers make it safer for businesses to implement server consolidation with less worry about applications being affected by resource constraints, faults, or security issues. In addition, N1 Grid Containers are easy to reconfigure, and can greatly simplify the task of migrating applications from one set of hardware to another.

Because N1 Grid Containers isolate multiple execution environments in a single instance of the operating system, they extend the concept of server virtualization down to a sub-CPU level of granularity without unnecessary performance or management overhead. Some other vendors are approaching the need for containment by running separate copies of the operating system on top of a virtual machine. This means that each instruction executed from a contained application must pass through the virtual machine as well as through the operating system, which creates unnecessary system overhead and adds to management complexity by requiring administration of additional copies of the operating system.

In contrast, multiple N1 Grid Containers can run within a single copy of the Solaris OS to achieve isolation without the performance overhead of a virtual machine and without adding complexity to the management environment.

N1 Grid Containers change the management model to be more service-centric and allow more standardized automation of underlying resource and system management tasks. Using N1 Grid Containers allows enterprises to achieve greater systems management efficiency and improvements in resource utilization.

Chapter 7

References

Sun Microsystems, Inc. posts product information in the form of data sheets, specifications, and white papers on its Internet World Wide Web Home page at: http://www.sun.com.

Look for the these and other Sun technology white papers:

Title	Location
Software Solutions - N1 Grid	wwws.sun.com/software/solutions/n1/index.html
N1 Grid - Introducing Just in Time Computing	wwws.sun.com/software/solutions/n1/wp-n1.pdf
N1 Grid Containers - Feature Story	http://www.sun.com/2004-0330/feature/index.html
N1 Grid Console	http://wwws.sun.com/software/products/grid_console/index.html
Solaris 9 Operating System	wwws.sun.com/software/solaris/index.html
Solaris 9 Resource Manager Data Sheet	wwws.sun.com/software/solaris/ds/ds-srm/index.html
Solaris 9 Resource Manager Software	wwws.sun.com/software/whitepapers/solaris9/srm.pdf
Resource Management in the Solaris 9 Operating Environment	www.sun.com/solutions/blueprints/0902/816-7753-10.pdf

Title	Location
Solaris 9 Resource Manager Technical FAQ	wwws.sun.com/software/solaris/faqs/resource_manager.html
Solaris 9 Network Resource Management Using IPQoS	wwws.sun.com/software/whitepapers/solaris9/ipqoswp.pdf
RFC 2475: An Architecture for Differentiated Services	www.ietf.org/rfc/rfc2475.txt?number-2475
System Administration Guide: N1 Grid Containers, Resource Management, and Solaris Zones	http://docs.sun.com/db/doc/817-1592
Sun/Oracle Best Practices	www.sun.com/solutions/blueprints/0101/SunOracle.pdf

Sun Microsystems, Inc. 4150 Network Circle, Santa Clara, CA 95054 USA Phone 1-800-555-9786 or 1-800-555-9SUN Web sun.com