TANEJA GROUP
TECHNOLOGY ANALYSTS

TECHNOLOGY VALIDATION

Hypervisor Shootout: Maximizing Workload Density in the Virtualization Platform

August 2010

THE NET NET

The summer of 2010 has once again brought changes to the server virtualization market, with the release of updated offerings from both VMware and Citrix. Twice in 2009, Taneja Group benchmarked the performance of the leading hypervisor platforms, with an eye to understanding which offered the best virtual machine (VM) densities, and we are pleased to update our findings in this new Technology Validation report.

Our testing again focused on VM Density, which is our measure of the number of VMs that can run simultaneously—executing a well-defined set of consistent application workloads—on a single hypervisor instance without disruptive performance impact (service-level breach). This time we expanded our testing to include the Red Hat Enterprise Linux 5.5 Kernel-Based Virtual Machine (RHEL 5.5 KVM) in addition to the latest available generation of vendor hypervisors: VMware vSphere 4.1 (ESXi 4.1), Microsoft Hyper-V R2 (Hyper-V) and Citrix XenServer 5.6 (XenServer).

Our density comparisons are based on a set of publicly available, open-source tests designed to evaluate machine performance under different levels of sustained workloads—similar to those generated by database-intensive business applications. We configured each hypervisor platform to allow for as much unhindered performance per virtual machine as possible, while taking full advantage of memory management features in each platform, to obtain fair results across hypervisors.

Why is VM Density important? Because it has a considerable impact on virtual infrastructure acquisition costs. Density will determine in large part the total cost of any infrastructure, because it determines how many physical machines must be maintained, and how many separate hypervisors must be supported, with attendant management software and other licensing costs. As we reported in 2009, a VM density advantage of 1.5:1 can yield a cost of acquisition savings of up to 29% for a typical virtual infrastructure. Depending on environment size, that savings may range from several thousand dollars to several hundreds of thousands of dollars.

Our latest testing using a workload that mimics real world SQL Server workloads better than ever before, suggests that ESX today has a VM density ratio well in excess of 1.5:1.

We found that VMware's ESXi 4.1 continues to lead the pack, delivering a density advantage of at least 2:1 and up to almost 3:1 versus Hyper-V R2 and between 1.7:1 and 2.3:1 vs. KVM. We also discovered that XenServer has closed the density gap in terms of number of concurrent VMs that can be run on a given host, coming to par with ESXi, but that this comes with a significant and unacceptable performance penalty. XenServer consistently delivers far less performance across the board (penalty ranging from 25% to 69%), and in our view gives ESXi as much as a 2:1 density advantage over XenServer, once we consider the ability of the hypervisor to access the full performance of the underlying hardware.

Test Procedures & Environment

For workload testing, we chose the open-source DVD Store Version 2 (DS2) test application, created by Dell. DS2 is a simulated on-line, transactional DVD movie rental/purchase web application which includes a back-end database component, load drivers, and a web application layer.

DS2 is a general-purpose stress and database testing tool that simulates users browsing a catalog and placing orders. We chose DS2 to provide broad-based stress tests that included variable execution patterns to more closely match production workloads seen in the enterprise.

DS2 exercises the database (SQL Server, in our tests) through fairly heavy use of dynamic SQL, and in our opinion satisfied our desire to test hypervisors under more "real-world" conditions than our previous tests, which relied on DBHammer and SPECjbb alone.

Test Configuration

Our test environment included:

- Server: HP BL460c G6 server blades with 24GB memory and dual quad-core hyperthreaded Intel Nehalem processors.
- Storage: EMC CX4-120 arrays (RAID5, with a 10-spindle RAID group)

Each VM on each hypervisor platform was configured to run a unique instance of DS2 in its "medium" database size configuration (1GB database) on Microsoft SQL Server 2008 R2 (64-bit). The medium database size simulates approximately 2 million customers, 100,000 products, and 100,000 orders/month.

Each VM was configured for 2 virtual CPUs and 2 GB RAM, with a Windows 2008 R2 image (64 bit). A separate virtual machine and hypervisor instance ran the drivers to generate loads, and custom scripts were developed to verify VM functionality before each test run, start multiple drivers, and collect performance results. DS2 reports performance in Orders per Minute (OPM) and utilizes many advanced database features (transactions, stored procedures, triggers, and referential integrity).

Test Procedures

For each of two load scenarios, the team launched a set of VMs executing DS2, increasing the number of VMs with successive runs and collecting OPM data for each VM in each run, until a maximum was reached beyond which no further VMs could be successfully launched on each hypervisor platform.

For each run, we also delayed collecting OPM data for a set "warm-up" interval to allow VM performance to level out while powered on and to allow time for each hypervisor's memory management features to have optimal effect.

Light Workload Tests

To evaluate performance under "light" workloads, we used a DS2 configuration of 2 threaded processes and a "think time" of 0.1 sec (simulates the time a user waits before responding to an event), with a 60-minute warm-up period for workload settling.

Heavy Workload Tests

Following this, the test was made more aggressive at each interval by reducing the think time to zero and the warm up to 10 minutes.

This significantly reduced the number of VMs that could be hosted on each platform (by dramatically increasing the load on each VM) and gave us greater insight into the differences among hypervisors under heavy loads.

Validation Testing Results

DVD Store Results: Light Workloads

The results for our light DS2 workload testing are shown in Figure 1, which charts the aggregate orders per minute (OPM) measured across all successfully launched VMs versus the number of concurrent workloads. We found performance to be fairly consistent and predictable for all the platforms tested under light workloads, though different for each hypervisor.

As expected from our previous test experience, **Hyper-V** drops off first due to a lack of any memory overcommit features. Therefore, Hyper-V testing saturated at 11 VMs (or 22GB of requested total VM memory).

With the addition of the Kernel Samepage Merging (KSM) feature, which de-duplicates memory across different VM processes, the **KVM** hypervisor (a new entrant in our testing) was able to support 14 concurrent VMs before reaching its memory and CPU overcommit limits.

XenServer 5.6 was the standout when compared to our 2009 test results. With the addition of the Dynamic Memory Control feature in 5.6, XenServer was able to keep up with ESXi 4.1 in density and performance, but for light workloads only (XenServer performance lagged by an average of 2.4%, which is statistically insignificant given our sample size).

Note that we configured XenServer DMC with default settings: a lower threshold of 512MB, which places an upper limit on how much memory can be reclaimed (1488MB) from each VM. In testing, we observed expected ballooning behavior – an equitable drawing down of memory from all existing VMs when a new VM was booted.

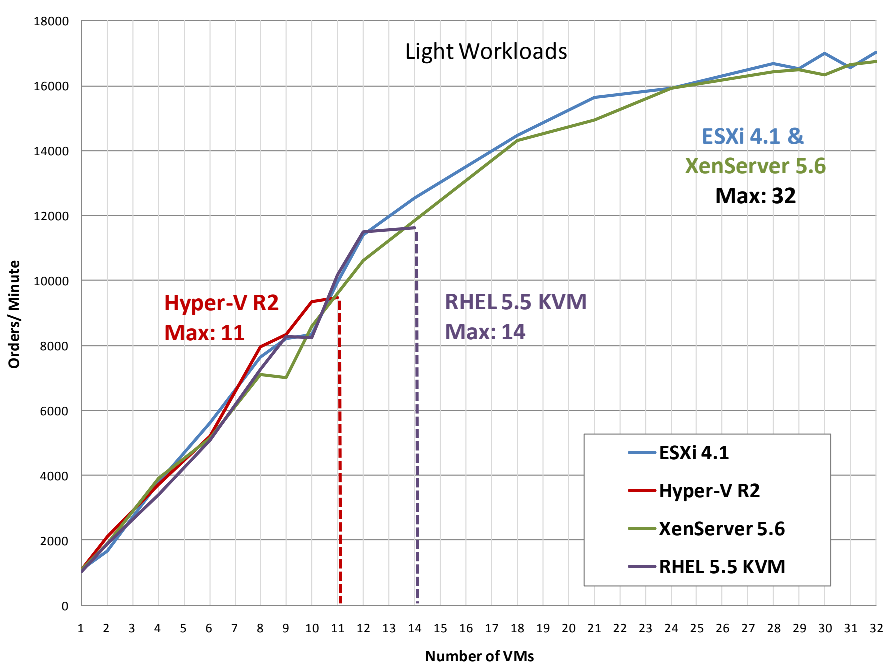


Figure 1: Performance under light DVD Store 2 workloads, in average orders per minute (OPM), vs. the number of concurrent VM workloads successfully launched.

Competitive Density Comparison: Light Workloads

Given these results, we can chart a macro-level VM Density comparison for light workloads (Figure 2). The impact of memory oversubscription is clear: XenServer has effectively matched ESXi's density numbers under light stress. However, both Hyper-V and KVM deliver unacceptable densities for these relatively light transactional workloads.

It would appear at this level that XenServer has closed the density gap, but is there more to the story? To dig deeper, we turned to our heavy workload tests to push stress beyond the baseline and to explore the limits of each hypervisor.

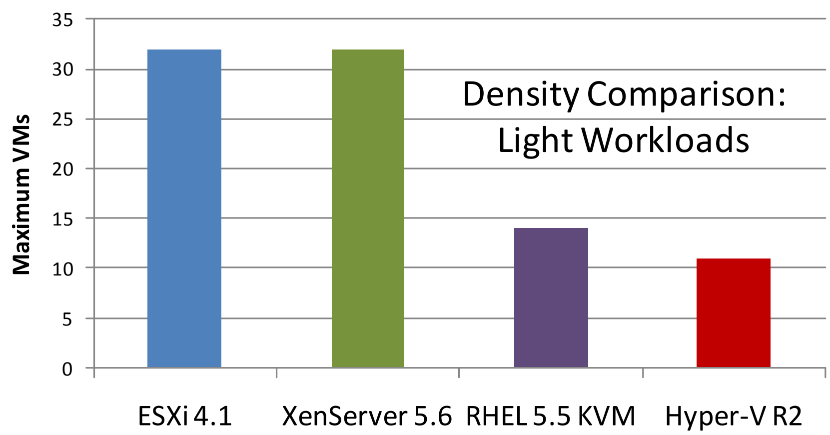


*Figure 2: Maximum VM Density under **Light DS2 workloads**: ESXi and XenServer: 32 VMs, KVM: 14, Hyper-V: 11.*

DVD Store Results: Heavy Workloads

Our DS2 results for much heavier workloads were less predictable, and the pressure from lowering the DS2 "wait time" parameter to zero made our results more erratic, as expected when each of multiple VMs is stressed significantly at high density. Regardless, significant density and performance differences emerged across platforms.

In Figure 3, we plot the OPM achieved for each platform under heavy load. We fully expected that the shorter warm-up time (10 minutes versus 60) would yield greater variance, as there was less time for the full effects of memory management to take effect (where applicable/available).

Note that we were still only able to launch 11 VMs on the Hyper-V platform, while KVM reached its maximum at 13 VMs, versus 14 in the lighter load test.

Again, XenServer showed the greatest improvement overall, scaling up to 22 concurrent VM workloads—matching ESXi's density—but also showing a dramatic performance penalty that became even more pronounced at higher VM densities.

In order to highlight the key differences in density and performance among the hypervisors, Figure 4 adds best-fit trend lines to simplify visual comparison.

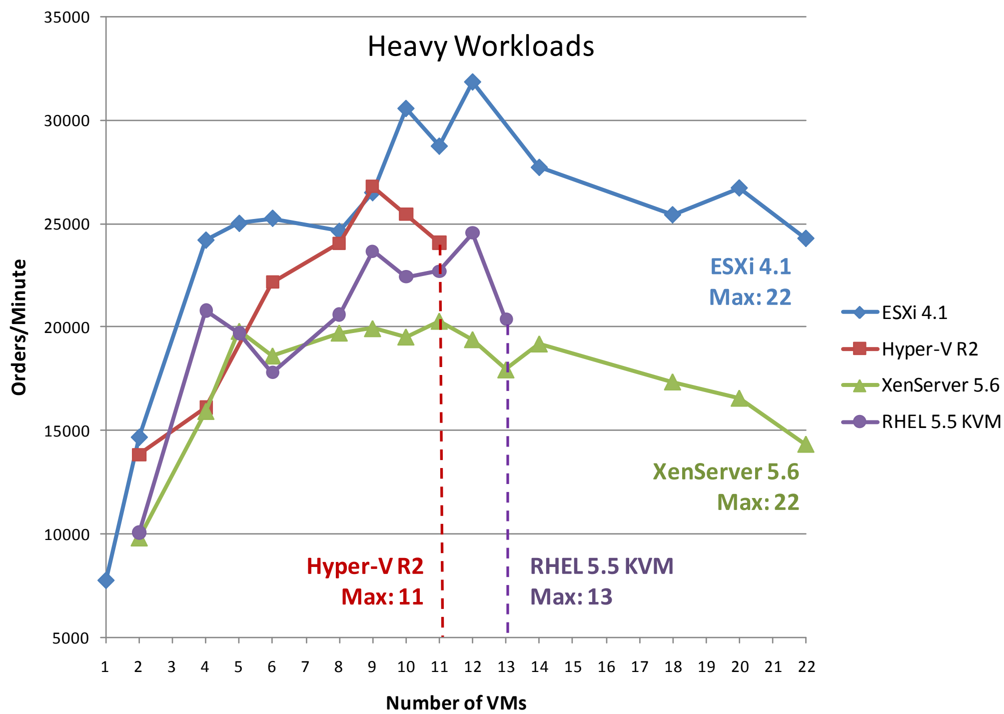


Figure 3: Performance in Orders/Minute under ***Heavy DS2 workloads*** *vs. number of concurrent VMs*

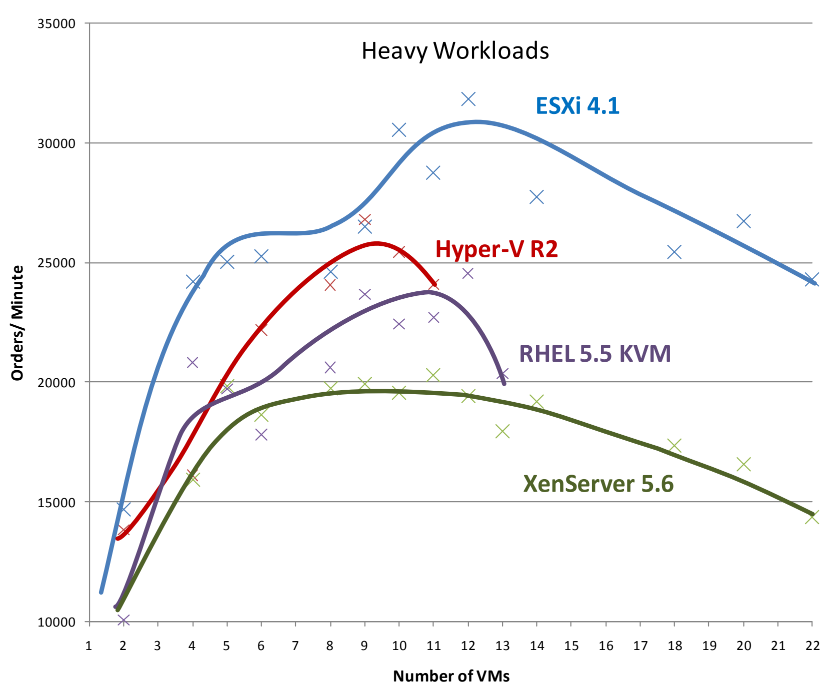


Figure 4: Performance under ***Heavy DS2 workloads****, with trend lines overlaid.*

Memory Management: Why Architectural Maturity Matters

Our data highlight the criticality of advanced memory management for high-performance, large-scale workload virtualization. The hypervisor platform must not only allocate CPU resources equitably and intelligently as workload demands increase, it must also deal intelligently with changing demand for shared memory, which is arguably the most precious resource in a highly consolidated infrastructure.

While many factors play a role in determining VM Density, VMware certainly has a significant engineering lead and has demonstrated maturity in both CPU and memory overcommit innovations. The company's competitors are only now claiming to deliver advanced features to match those in ESXi, and these claims remain limited. Red Hat, for example, warns strongly against CPU overcommitment beyond the point where the total number of virtual CPUs exceeds the available physical CPUs—a significant limitation with respect to ESXi.

Memory Management Overview

In a dynamic, virtualized infrastructure, memory contention is likely to change rapidly, and without clear visibility. This can quickly create memory oversubscription, either due to high consolidation requirements or workload mobility, both planned and unplanned. Therefore, it's important to understand how each hypervisor platform has addressed memory overcommit in order to understand its impact on the level of VM density the platform can comfortably support.

In general, virtualized memory resource management consists of some combination of VM guest and hypervisor memory allocation and reclamation technologies, which vary among the leading platforms. These include:

Memory Oversubscription (Ballooning): Typically this is done by utilizing a per-VM driver to enable the hypervisor reclaim memory by artificially increasing memory pressure inside VM guests. This per-VM driver is commonly called a balloon driver. The intelligence behind a balloon driver may vary, from simple reclamation and swapping memory to disk (just as host page files do), to more advanced techniques which target less heavily-used memory and/or which allow allocations across guests to change more frequently. The implementation here is important: CPU utilization can be adversely affected by the guest operating system's response time for memory allocation or the I/O overhead involved in waiting for swapping transactions to complete, or both.

Memory Page Sharing: Beyond ballooning (the virtualization of memory so that it appears there more than is physically present, and the forced de-allocation of memory from a VM guest), memory page sharing is another effective mechanism for reducing the memory footprint for a collection of VMs. With page sharing, redundant data in memory is identified and shared across all VMs, using copy-on-write techniques to accommodate modifications by individual VMs. This technique is particularly effective when many VMs on a hypervisor share the same operating system and/or application set.

Hypervisor Swapping: When ballooning and page sharing are not sufficient to reclaim memory, the hypervisor itself can employ swapping for reclamation. Hypervisor swapping should be a last resort, due to page selection problems, double paging problems, and high swap-in latency, which can all severely penalize guest performance. In a mature hypervisor environment, however, swapping can be useful to respond quickly to changing memory requirements, when ballooning isn't fast enough.

Memory Compression: This technique attempts to compress swapped-out pages and store them in a compression cache located in main memory. If this is possible, the next page access will only cause a page decompression, which can be much faster than disk access. Compression, deployed judiciously, should improve application performance when the hypervisor is under heavy memory pressure.

Memory Management in ESXi

VMware pioneered advanced memory management in the hypervisor, and has continued to enhance its patented, multi-level approach to memory resource management over the last decade. ESXi 4.1 is the only hypervisor we tested that makes use of all four techniques described above.

ESXi Transparent Page Sharing (TPS) employs hashing to efficiently identify redundant pages by their contents, with advanced settings that allow scan rates and the maximum number of per-VM scanned pages to be specified. Ballooning leverages the guest operating system to intelligently make decisions about the memory to be paged out when placed under memory pressure by the hypervisor. Hypervisor swapping is used as a last resort when the host system is under severe memory pressure, and memory compression (new in 4.1) utilizes a per-VM compression cache, which is transparent to the guest OS.

ESXi provides automatic, dynamic page sharing and memory ballooning as an integral component of the hypervisor platform architecture. These mechanisms are mature, well-orchestrated, and highly configurable.

Indeed, all of these memory management techniques are enabled by default in ESXi, and customers typically enjoy performance benefits without the need to enable or configure them—they just work.

Memory Management in the Competitive Hypervisors

Microsoft, in Hyper-V R2, does not support memory over-commitment, but has announced upcoming support for Dynamic Memory, a technology the company claims is different from page sharing mechanisms and which allows memory to be treated as a "dynamically scheduled" (rather than overcommitted) resource. We did not test this feature, as Windows Server 2008 R2 Service Pack 1 was not available at test time. We look forward to exploring Microsoft's claims in the future.

KVM claims support for memory ballooning as well as page sharing, via KSM. Note that the page sharing technology in KSM (implemented as a user-space daemon and applicable across any Linux processes, including KVM VMs) has similar objectives to VMware's patented TPS, but is implemented differently (does not use hash tables to identify common pages, for example). The KSM code is relatively new in the Linux kernel and has therefore undergone less rigorous testing than TPS. Regardless, we were unable to confirm the availability of ballooning drivers for Windows guests and assume no ballooning was occurring for RHEL 5.5 KVM during our testing.

Citrix has exposed the balloon driver in its Xen hypervisor with the recent release of **XenServer 5.6**. XenServer Dynamic Memory Control allows an administrator to configure high and low memory thresholds per VM. Under memory pressure, the hypervisor will reclaim memory from running VMs via ballooning, enabling memory overcommit (higher density). A XenCenter interface allows memory ranges to be modified at run-time, without rebooting running VMs. Memory page sharing (via KSM or otherwise) is not available in XenServer 5.6.

Technology Validation: Results Analysis

Analysis Overview

It's clear to us that following years of claiming that VMware's memory overcommitment technology, while elegant and powerful, wasn't necessary or desirable for most virtualization projects, the leading competitive hypervisor developers are now in a heated contest to take all or part of that message back.

This makes sense: following the initial virtualization payback due to light workload consolidation, further gains must come from some degree of resource oversubscription.

The economic drivers and opportunities of commodity cloud computing are certainly driving this: in the cloud, compute, storage, and memory will often **need** to be oversubscribed in order for service providers to cost-effectively provide enough resources to multiple customers simultaneously—each one with highly variable workload demands—without provisioning for a total peak load.

The quest for commodity compute efficiencies naturally leads us from consolidation to overcommitment. All hypervisors tested support overbooking of CPU resources (with the caveat for RHEL 5.5 KVM mentioned earlier), and thin provisioning allows users to overbook storage. Memory overcommitment is a logical extension, but until recently VMware's competitors have claimed this was too risky for the average enterprise customer with typical business-critical workloads.

We're pleased to see that over the last year, each competitive hypervisor platform has been extended to demonstrate, at least rudimentarily, that it supports memory overcommitment, or has announced such support.

However, hypervisor memory management is a relatively new technology, and our testing indicates that each vendor and platform differs in both maturity and innovation. And these differences have a direct impact on VM density and aggregate performance.

ESXi vs. Hyper-V

Hyper-V continues to lag the field with no memory overcommitment features in the production release that was available at the time of testing (Microsoft estimates that Hyper-V R2 SP1 will be generally available in the first half of 2011).

When put up against ESX 4.1, which includes additional core performance enhancements implemented since 4.0, Microsoft's VM density disadvantage has increased since our testing last year: under typical light workloads, ESX 4.1 can now run almost *three* times as many VMs as Hyper-V (Fig. 5).

Hyper-V has made performance gains, however. Until it reaches hard memory limitations, Hyper-V is the best performing competitor of ESXi under heavy workloads, reaching near parity with ESXi (around 8-9 VMs) after a slow ramp-up. Unfortunately, performance drops precipitously as memory pressure increases, and ESXi is still able to deliver an overall VM density advantage of two-to-one (Fig. 5).

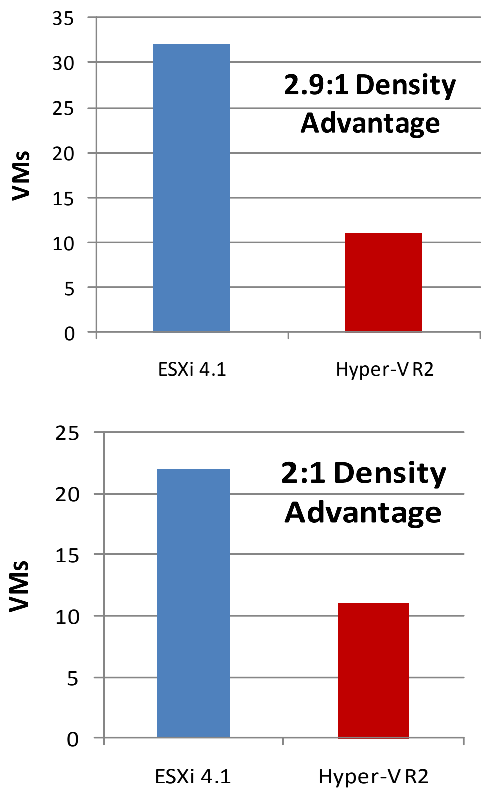


Figure 5: ESXi vs. Hyper-V Density results under Light (top) and Heavy (bottom) DS2 workloads.

At its maximum density (11 VMs), Hyper-V lagged ESXi workload performance by 19% (Figure 6), and ESXi was able to deliver performance matching the best value we recorded for Hyper-V at almost twice as many VMs. Hyper-V also showed a steeper decline in performance when nearing memory limits.

Note, with the slower ramp-up, and the precipitous drop-off, we're led to conclude that Hyper-V still demonstrates some resource time-slicing behavior that favors certain increments of workloads rather than scaling smoothly - this was in fact was also a key finding of our last testing in September of 2009.

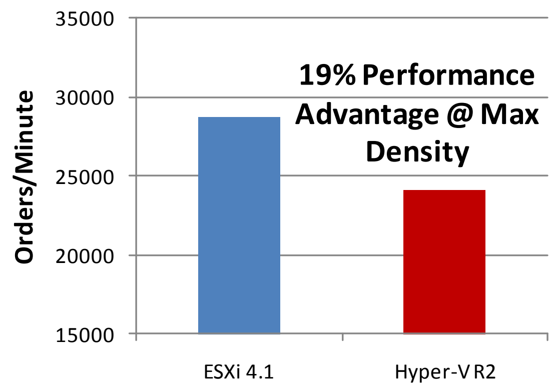


Figure 6: ESXi performance advantage over Hyper-V under Heavy workloads.

At its maximum density (11 VMs), Hyper-V lagged ESXi workload performance by 19% (Figure 6), and ESXi was able to deliver performance matching the best value we recorded for Hyper-V at almost twice as many VMs. Hyper-V also showed a steeper decline in performance when nearing memory limits.

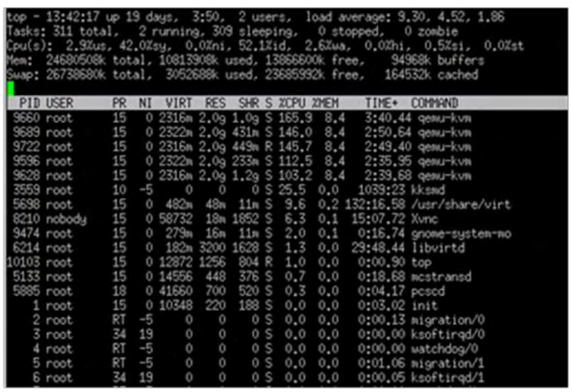
Given these results, we're hard-pressed to confer any performance advantage for Hyper-V, with light or heavy application loads. We are eager to evaluate Microsoft's planned Dynamic Memory features when they are available. What is the key take-away here?

Density still matters, and without memory oversubscription, Hyper-V is actually losing ground in the face of ESXi capabilities, which are marching steadily forward in performance. *Without memory oversubscription, per-VM performance isn't necessarily being left on the table, but the potential to manage additional workloads certainly is.*

ESXi vs. KVM

Enabling KSM gave Red Hat Enterprise Linux 5.5 KVM a density advantage over Hyper-V, but results still significantly lagged both ESXi and XenServer.

We were able to verify that memory de-duplication (memory page sharing) was taking place on the KVM VMs, and could see during test execution that shared memory indeed varied across VMs and was changing dynamically (Figure 7).



```
top - 13:42:17 up 19 days,  3:50,  2 users,  load average: 9.30, 4.52, 1.86
Tasks: 311 total,   2 running, 309 sleeping,   0 stopped,   0 zombie
Cpu(s):  2.9%us, 42.0%sy,  0.0%ni, 52.1%id,  2.6%wa,  0.0%hi,  0.5%si,  0.0%st
Mem:  24680508k total, 10813908k used, 13866600k free,    94968k buffers
Swap: 26738680k total,  3052688k used, 23685992k free,   164532k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 9660 root      15   0 2316m 2.0g 1.0g S 165.9  8.4   3:40.44 qemu-kvm
 9689 root      15   0 2322m 2.0g 431m S 146.0  8.4   2:50.64 qemu-kvm
 9722 root      15   0 2316m 2.0g 449m R 145.7  8.4   2:49.40 qemu-kvm
 9596 root      15   0 2322m 2.0g 233m S 112.5  8.4   2:35.95 qemu-kvm
 9628 root      15   0 2316m 2.0g 1.2g S 103.2  8.4   2:39.68 qemu-kvm
 3559 root      10  -5     0    0    0 S 25.5  0.0   1039:23 kksmd
 5698 root      15   0  482m  48m  11m S  9.6  0.2 132:16.58 /usr/share/virt
 8210 nobody    15   0 58732  18m 1852 S  6.3  0.1  15:07.72 Xvnc
 9474 root      15   0  279m  16m  11m S  2.0  0.1   0:16.74 gnome-system-mo
 6214 root      15   0  182m 3200 1628 S  1.3  0.0  29:48.44 libvirtd
10103 root      15   0 12872 1256  804 R  1.0  0.0   0:00.90 top
 5133 root      15   0 14556  448  376 S  0.7  0.0   0:18.68 mcstransd
 5885 root      18   0 41660  700  520 S  0.3  0.0   0:04.17 pcscd
    1 root      15   0 10348  220  188 S  0.0  0.0   0:03.02 init
    2 root      RT  -5     0    0    0 S  0.0  0.0   0:00.13 migration/0
    3 root      34  19     0    0    0 S  0.0  0.0   0:00.00 ksoftirqd/0
    4 root      RT  -5     0    0    0 S  0.0  0.0   0:00.00 watchdog/0
    5 root      RT  -5     0    0    0 S  0.0  0.0   0:01.06 migration/1
    6 root      34  19     0    0    0 S  0.0  0.0   0:00.05 ksoftirqd/1
```

Figure 7: KVM Kernel Samepage Merging in effect across VM processes (SHR column).

However, even with the added benefit of this relatively new feature in KVM, memory limitations capped the hypervisor at 14 concurrent VMs under light loads and 13 under heavy loads, yielding a high density advantage for ESXi, which scaled to 32 and 22 concurrent workloads, respectively (Figure 8).

Although KVM's density disadvantage relative to ESXi was less than that of Hyper-V, its *performance disadvantage at maximum density was much greater* (Figure 9). In our view, KVM's recently added memory overcommitment mechanisms (KSM) improve its density beyond Hyper-V, but not by much, and fail to bring it anywhere near parity with VMware.

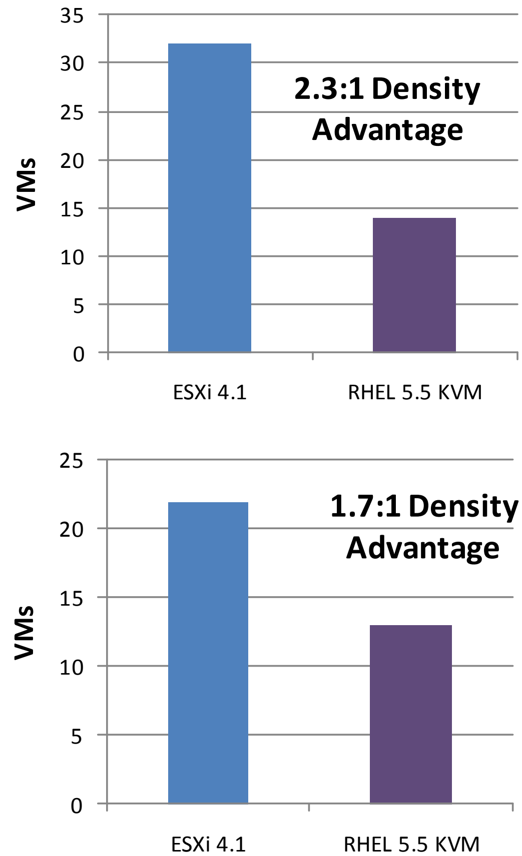


Figure 8: ESXi vs. RHEL 5.5 KVM Density results under Light (top) and Heavy (bottom) DS2 workloads.

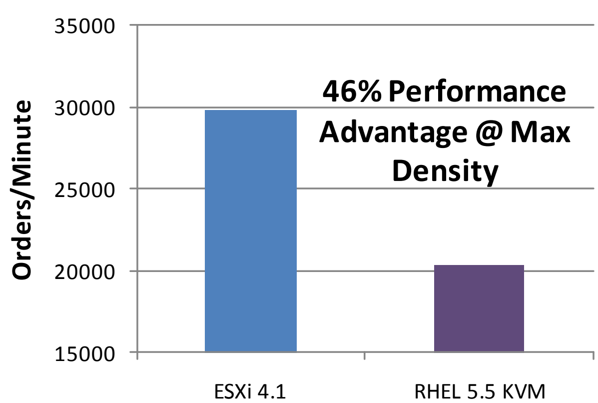


Figure 9: ESXi performance advantage over RHEL 5.5 KVM under Heavy workloads.

While RHEL KVM *performance* eventually rivals Hyper-V above 10 concurrent VMs, drop-off is swift soon after, and KVM's limited memory management features (combined, perhaps, with issues handling CPU overcommit, as discussed) yield a hard upper limit of 13 VMs in our heavy workload testing, with significant performance variability.

Up to this limit, ESXi handily outperforms KVM at each interval, and our final results give ESXi a density advantage over KVM of 2.3:1 (light) and 1.7:1 (heavy). KVM's rapid drop-off also gives ESXi a 46% performance advantage at KVM's 13-VM maximum.

ESXi 4.1 can run more than twice as many light workloads and 70% more heavy workloads than RHEL KVM. At maximum density, the performance penalty KVM pays to achieve overcommit is unacceptable. We think KSM has promise, however, and we look forward to tracking its maturity.

ESXi vs. XenServer

The big gainer over last year's results is XenServer 5.6, which has finally made the leap into memory overcommit and has leveled the density playing field for light workloads (matching ESXi's density of 32 concurrent VMs).

Under both our light and heavy workloads, XenServer 5.6's memory management features enabled the platform to match ESXi's VM density numbers, but with a performance penalty that was surprisingly high and unfortunately consistent for our heavy application loads at every number of VM workloads.

Figure 10 illustrates this difference clearly (ESXi and XenServer data only, extracted from Fig. 3), while Figure 11 plots XenServer's performance penalty as the number of VMs increases, showing a growing disparity at higher VM density:

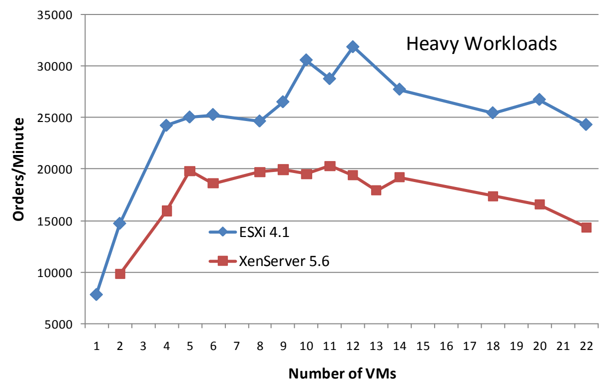


Figure 10: ESXi performance advantage over XenServer under DS2 Heavy load.

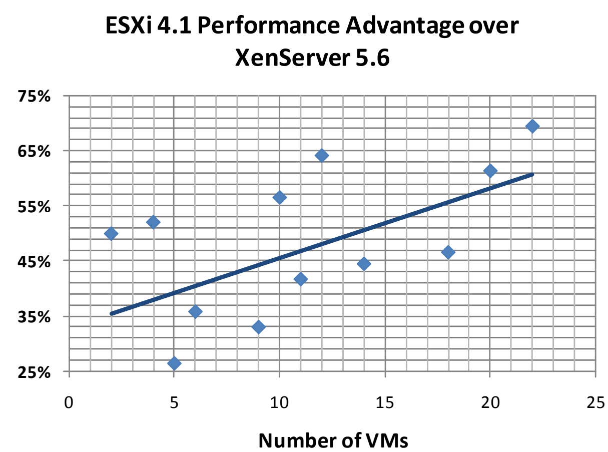


Figure 11: ESXi's performance advantage over XenServer5.6 increases as VM density increases.

Given an average performance penalty of a whopping 47%, with a high of an even more astonishing 69%, we feel that **ESXi's density advantage over XenServer is at least 1.5:1**. In other words, you'd need 50% more capacity on the host to run an equivalent set of workloads at scale with XenServer versus ESXi.

XenServer's performance under heavy loads, in our view, provides key insights into the differences in maturity and sophistication of its memory management features with respect to VMware's. While we were able to continue launching additional VMs to match ESXi's overall density, XenServer's performance across the board was the poorest, leveling out at approximately the same performance ESXi delivered at 3 VMs, holding steady at that level for a bit, then declining as density continued to increase.

At every density level, XenServer delivered between 25% and 65% less performance than VMware. In our view, the recently-exposed balloon driver in XenServer 5.6 is allowing many more VMs to be launched (higher density), but the application performance impact is unacceptable. The balloon drivers are giving each VM enough memory to boot, but are not effectively managing memory dynamically across a set of VM peers to deliver adequate performance.

We suspect that XenServer is degrading performance in the face of application contention for CPU resources much more gracefully than it handles degradation due to memory pressure. While we haven't delved into kernel-level details, our observations of surface behaviors suggest that XenServer handles overbooking with a very singular, flat policy across all virtual machines. At scale, XenServer pays a price for this.

Citrix's term for this feature is Dynamic Memory Control, but the capability is not dynamic in the sense we expected. Upon further investigation, DMC is actually a static, pre-allocated memory overcommit that handicaps every VM equally. There is no dynamic, workload-based memory management. The name instead refers to the ability of an administrator to manually adjust the minimum and maximum memory thresholds (within which ballooning operates) of a running VM on the fly without rebooting.

At these levels, XenServer's memory overcommit performance represents very little competitive threat to the multi-tiered and mature capabilities available in ESXi 4.1. We suspect that VMware's advanced page sharing capabilities (combined with ballooning) are the more effective mechanism, and yields higher levels of per-VM memory optimization. XenServer's ballooning implementation doesn't take run-time workload demands into account, and for this reason we hesitate to call it "dynamic."

Indeed, XenServer's current memory management approach may rapidly cause issues in a dynamic virtual infrastructure, as it requires additional CPU overhead—we saw this in the performance penalty paid at scale. XenServer's current implementation, in our experience, deprives all VM guests equally and blindly, and relies solely on deprivation to make room for additional workloads as density increases. This distributes and potentially amplifies the impact of memory oversubscription as the pain is shared across all guests, regardless of workload profile.

When memory becomes significantly oversubscribed, XenServer will throttle all systems, by making each guest swap and deal with the processing consequences of the swap. Because of impact upon other system resources and all workloads in the system, the last thing you want to utilize poorly is memory. Dynamic Memory Control is a step in the right direction, but is not very useful in its current form, and we await additional intelligence in the memory management layer before we can comfortably state that XenServer memory oversubscription has arrived.

Analysis Summary

Memory overcommitment isn't an easy process, nor is its implementation simple. There are multiple factors and actors in play within hypervisors when it comes to optimizing the use of shared memory, and it seems clear that more is needed than a VM-level ballooning driver. Our testing has shown that ballooning, page sharing, and compression require a comprehensive approach and that first-generation implementations have a long way to go.

KVM and XenServer have both implemented very limited versions of what ESXi already does, and has done for many years. One clear observation from our testing is that the interplay of compute and memory sharing yields different behaviors on different hypervisors, and the competitive platforms still struggle to bring sophistication and visibility to their memory oversubscription features (if those features exist at all).

We feel that memory played a role well beyond what is reflected in the numbers gathered in this single validation exercise. Specifically, we saw wide variation in the behavior of memory reclamation efforts.

XenServer, for example, essentially *reallocates* memory to new guests at boot time (and/or when memory is adjusted on any one of the guests) if the total requested memory exceeds that available, and does so using a brute-force approach that forces guests to swap their memory to disk.

We expected less elegance, however, and we are encouraged that competitors have been investing time and energy to closing the density gap. Nonetheless, implementation deltas still exist, and as is expected, hypervisors that are just now gaining memory management features do not exhibit the most well-balanced use of such features.

In particular, we observed much less equitable workload balancing on the competitive platforms, and found that ESXi simply had more configurable parameters, more deployment options, more automation, and more advanced heuristics—well beyond bulk memory reclamation simply based on high-low thresholds set by an administrator.

Beyond the Hypervisor Platform

We'd also like to make it clear that while hypervisor density and efficiency are essential elements for reducing the total cost of ownership of a virtual infrastructure, higher-order workload management and resource scheduling capabilities also play a large role.

VMware's high-performance CPU and memory sharing features are integrated tightly with the vCenter Server management framework.

Through this extensive management platform, VMware delivers additional workload efficiency via pool management (Distributed Resource Scheduling), High Availability, and live workload and storage migration (vMotion and Storage vMotion), to name a few features.

While competitive hypervisor vendors continue in their attempts to replicate these vCenter capabilities, we believe that VMware remains in the lead, and that careful consideration should be paid to how a comprehensive management suite can reduce total cost of ownership.

Taneja Group Opinion

Maximizing VM density is a rapidly changing objective and a quickly moving target for all of the major hypervisor platforms. After several years of dismissing VMware's claims that memory oversubscription is not only valuable but essential for cost-effective virtualization at scale, we feel that the economics of cloud computing are pressuring the virtualization leader's competitors to change their message as quickly as possible. We're encouraged to see competitors working on memory overcommit solutions, rather than continuing to claim they don't matter.

The highly consolidated, cost-efficient, and optimally utilized cloud computing model rapidly emerging both inside and outside the corporate firewall demands some critical features from its underlying virtualization platform. Among these are support for very high densities, reliable and multi-tiered memory management, effective CPU scheduling, predictable scaling, graceful performance degradation, and automated resource allocation and reclamation optimized for a range of workload types.

Critical to success for the hypervisor platforms will be their support for and implementation of memory overcommitment technologies. Optimized shared memory management is as important moving forward as CPU sharing has been to the first wave of workload virtualization. Our testing indicates that VMware is further along in the journey than its major competitors, all of whom have only recently released first-generation memory management features (if they've released them at all). The mixed results we've observed indicate that while progress is being made, virtual shared memory optimization is not easy. We expect these early attempts from VMware's competitors to mature over time.

KVM has made progress, XenServer has made more, and Microsoft has yet to release its answer to the memory management challenge for Hyper-V. In our view, however, VMware remains the only vendor currently offering an enterprise-class, comprehensive, automated and high-performance memory overcommit solution. VMware continues to enhance and innovate in this area, recently adding memory compression to its arsenal of memory management technologies. And, ESXi continues to deliver the highest VM density ratios with the greatest level of overall performance per VM under heavy application loads of any vendor in the market segment.

We recommend that every organization undertaking a server virtualization initiative sharpen its pencils and review the VM density findings in this report. Using our results, combined with an associated cost per application (see the approach outlined in Taneja Group's March 2009 VM Density report, or use VMware's cost per application calculator), customers can identify the true cost of the virtualization solutions they are considering. Viewing the many competing virtualization solutions on the market through this lens will help you figure out how to do more with less.