BEST PRACTICES FOR OPTIMIZING YOUR LINUX VPS AND CLOUD SERVER INFRASTRUCTURE

Maximizing Revenue per Server with Parallels Containers for Linux

Q1 2012
Table of Contents

Overview ............................................................................................................................................... 3
Maximizing Density per Server ........................................................................................................... 4
  Balancing Quality of Service and Density ...................................................................................... 4
  Improving Resource Efficiency with Parallels Containers’ File System and Templates ............... 9
  Memory and IOPS Deduplication (Containers) .............................................................................. 10
  Summary of Best Practices for Maximizing Density ................................................................. 12
  Real-World Examples ................................................................................................................... 13
Optimizing Your Server Configuration and Management ............................................................ 15
  Best Practices for Maximizing Quality of Service and Profitability ......................................... 15
  Real-Word Examples of VPS Configurations ............................................................................. 16
Conclusion ......................................................................................................................................... 18
Appendices: Checklist for Optimizing Your VPS Infrastructure .................................................. 19
Overview

The purpose of this document is to provide you with best practices for optimizing the performance and cost of servers running Parallels Containers. Parallels Containers is one of the components of Parallels Cloud Server, which also includes Parallels Hypervisor and Parallels Cloud Storage. These best practices are the result of our years of experience in providing containers virtualization to our service provider partners, combined with the latest customer and VPS market research and our most recent product improvements. By implementing these best practices, you'll be able to increase the return on your infrastructure investment and maximize your revenue per server, while also offering high quality of service.

This report focuses on the two most critical factors for the profitability of your VPS business:

- **Maximizing density per server**. Optimizing the number of containers deployed on each hardware node is a key way to increase infrastructure efficiency. To ensure high quality of service for your customers while also maximizing your profitability, you'll need to ensure your server configurations support the container density and the VPS offerings you are deploying. You'll also need an understanding of the factors that influence container density in Linux environments, which you will gain from reading this paper. By employing the best practices recommended here, you'll be able to realize the most efficient density for your servers.

- **Optimizing your server configuration and management**. Another key factor in building a profitable VPS business is hardware configuration and management. You'll want to choose hardware that will ensure both high quality of service and ease of maintenance. By following best practices for your hardware configuration and allocated resources, you can significantly increase your operational efficiency.

This report covers best practices for both of these success factors, including real-world examples, to help you maximize your revenues per server.
Maximizing Density per Server

Because commercial VPS offerings are employed in many different use cases, there is no one single optimal configuration. Therefore, the most important factor in managing Parallels Containers servers efficiently is to match the hardware node configuration with the VPS product offerings, target market, and physical environments you plan to deploy.

In general, our partners have found that larger, more expensive VPS configurations typically have a higher usage level than smaller VPS configurations. That’s because larger VPSs are typically purchased to support shared hosting and business-critical functions, and as a result require access to all of their allocated physical resources a majority of the time. Smaller VPSs, in contrast, are typically used to support more random usage, such as software testing or less critical business applications, and therefore may only rarely need to access their maximum configured resources.

Understanding how your customers use their VPS offerings will help you ensure that your VPS offerings deliver a high quality of service while also optimizing the physical resources allocated to individual containers.

Balancing Quality of Service and Density
One of the key benefits of Parallels Containers is that it enables you to overcommit the physical hardware resources allocated to a server’s containers. This means that, on a given server, the total resources allocated to its containers may exceed the server’s total hardware resources. If managed properly, this overcommitment of resources will still deliver high levels of service, because the resource load across all the containers on any given server is typically randomized – which means it’s rare for all the containers on any given server to require the maximum amount of allowable resources at the same time. Many service providers choose to overcommit their resources because the higher density they achieve helps maximize their revenues.

How Do Container Limits Work?
Parallels Containers comes with a variety of controllers – for CPU usage, memory, disk I/O, disk space, network bandwidth, and more. These controllers ensure that services in one container can’t affect other containers on same the physical system. Of the various
resources the controllers manage, our partners tell us that the most important are memory and disk I/O.

In most cases, services running inside containers will consume resources below their assigned limit. However, should they approach or try to go over the limit, the container virtualization layer will prevent this – for instance, by pushing pages into VSwap in the case of memory limits, or by throttling disk transactions in the case of I/O bandwidth limits.

What is an “Overcommitted” System?
A physical system is overcommitted for a particular resource when the hard limits for that resource across all its containers add up to more than it can deliver. For instance, if you have 100 containers with hard limits of 2GB of memory apiece on a physical system, but the system has only 128GB of memory, that system’s memory is overcommitted. But as long as the actual resource usage of the containers adds up to less than 128GB, this overcommitment is only theoretical, and the system will operate normally. It’s only when the sum total of the resources requested by all containers on a physical system exceeds what the system can deliver that the overcommitment can create problems. A system in a state of actual overcommitment will degrade the service that each of the containers provides, often becoming unable to deliver on your Service Level Agreements (SLAs).

When Should You Overcommit?
This is one of the most important questions that your container placement policy needs to address. One possible answer is “Never,” meaning that you will place containers on physical systems only up to the point where their resource limits add up to the system’s total capabilities. While such a policy does ensure that the containers will always deliver on your SLAs, it’s likely to result in a system that is underutilized much of the time. This is why most service providers choose to overcommit their resources to some degree. In deciding whether – and how much – to overcommit, you should always run analyses and simulations with realistic customer workloads, to help you establish the right parameters.

Calculating a Static Overcommit
A static overcommit balances density with quality of service and is easily implemented when you initially provision the containers. For it to work well, however, you must have a
good understanding of how customer workloads vary over time and calculate that there will be a very low probability that the system will get into an actual overcommitted state.

For instance, suppose that a customer workload usually uses about 50% of its memory resource (excluding the page cache), but occasionally spikes to 100% usage for one minute in every hour – and suppose further that the maximum number of containers that can be placed on the system without overcommitting it is 100. Because the typical utilization is 50%, you could theoretically place 200 containers on the system and have it mostly perform well. But with each application spiking for one minute out of 60, about three or four of the 200 containers will be spiking at any given time, so the system will always be running in an actual overcommitted state.

However, if you alter your placement policy to allow up to 190 containers on the system (for a static overcommitment of 95%), you'll end up with room for five containers to spike at any given time without sending the system into an actual overcommitted state. Given that each container spikes for one minute in 60, we can calculate that there will be about a 11.9% probability that six or more containers will spike at once, giving you a 88.1% probability of meeting the SLA. But that's still not good enough if you're promising two nines of availability: in that case, you would want a static overcommit of 80% (180 containers), which would put the probability of meeting the SLA at 99.7%.

The foregoing numbers are just examples, of course. To get the actual numbers that apply to your situation, you will need to study how your customer workloads vary over time and calculate what level of static overcommit will give you a level of probability that will enable you to meet your SLAs.

It's also important to be aware that, while probabilistic analysis can tell you whether you are likely to be able to meet your SLAs or not, a standard container system probably runs a variety of different workloads, and the usual workload is less predictable than the calculations above assume. For that reason, you'll need to be very careful in managing and adjusting your overcommit policy. Once you perform the calculations for your situation and establish an overcommit policy, you should continuously monitor the system to determine what your actual level of SLA achievement is and adjust the static overcommit percentage as needed to get the results you want.
Setting Resource Limits
The most important server resources to manage are memory, CPU processing, and disk I/O. Parallels Containers lets you configure each of these resources as follows:

- **CPU Resource Scheduling.** Parallels Containers CPU-related parameters include *cpuunits* and *cpulimits*. These parameters both impact the CPU resources available to a container but are not directly related to each other. The *cpuunits* parameter defines the weighting of CPU resources between containers. An individual container will be able to consume up to 100% of the CPU resources if there are no other containers competing for the CPU and you haven’t defined the *cpulimit* parameter. If multiple containers are requesting CPU resources, then the container with the higher *cpuunits* value will have a higher chance to have CPU resources assigned. For example if a physical server can deliver 3000 *cpuunits* of computing resource and you provision 2 containers on the server, one with 2000 *cpulimits* and one with 1000 *cpuunits*, and both containers simultaneously request all available CPU resources, the container with 2000 *cpuunits* will get twice as many CPU cycles as the container with 1000 *cpuunits*. The *cpulimit* parameter defines a hard limit which controls the total percentage of CPU resource a container can utilize. In establishing both of these parameters, it's important to make sure that the values you use are consistent with the number of CPUs installed the server.

- **Memory.** Parallels Containers 6.0 uses the RHEL6-based OpenVZ kernel, which includes a new memory management model called VSwap that supersedes User Bean Counters. With VSwap, two primary parameters – *physpages* and *swappages* – let you control the amount of allocated memory and swap space. All the other bean counters are now secondary and optional. The sum of the *physpages* and *swappages* limits is the maximum amount of allocated memory that a container can use. When the *physpages* limit is reached, memory pages belonging to the container are pushed out to virtual swap space. Once the total amount of allocated memory is consumed the container performance will start to degrade.¹

- **Disk I/O.** Two disk I/O parameters, *I/O Limits* and *IOPS Limits*, provide very granular and powerful controls to avoid performance degradation in situations when high disk I/O activities in one container could slow down the performance of other containers. It's

¹ To learn more about using VSwap, read our “VSwap Best Practices” white paper
a common best practice to limit the disk throughput per container to the maximum available network bandwidth. However, setting a value between 10 MB/sec (lowest limit) and 75% of the overall disk throughput (upper limit) will allow the physical machine to handle operations such as migration, backup, and reinstallation rapidly, even the disk I/O if a particular container begins to approach its limits.

- **Disk space.** In high-density environments, it's very important to make sure that disk quotas aren't exceeded, as exceeding them will degrade performance. Parallels Containers lets you assign two types of disk quotas: one limiting disk space for the entire container, and the other limiting the space available to individual users inside the container. Disk space quotas for individual users is managed exactly in the same way as in a non-virtualized Linux kernel, enabling you to continue to use your existing management tools.

### Example of an Overcommitted Strategy

Table 1 provides an example illustrating how you might establish different levels of overcommitment for different resources. These are provided for example only, the actual level of overcommit you configure in your system will depend on the factors we have discussed above.

---

2 To learn more about using setting Disk I/O limits read our “Disk I/O Limits Best Practices” white paper.
Table 1: Sample Overcommit Strategy

<table>
<thead>
<tr>
<th>Typical Configuration</th>
<th>RAM</th>
<th>CPU</th>
<th>DISK I/O</th>
</tr>
</thead>
<tbody>
<tr>
<td>Physical Resource</td>
<td>40 GB</td>
<td>16 Cores</td>
<td>100 MBpS</td>
</tr>
<tr>
<td>Number of containers per node</td>
<td>40</td>
<td>40</td>
<td>40</td>
</tr>
<tr>
<td>Resource committed per container</td>
<td>2 GB</td>
<td>2 CPU</td>
<td>10 MBpS</td>
</tr>
<tr>
<td>Total committed resources</td>
<td>80 GB</td>
<td>80 cores</td>
<td>400 MBpS</td>
</tr>
<tr>
<td>Overcommitment %</td>
<td>100%</td>
<td>400%</td>
<td>300%</td>
</tr>
</tbody>
</table>

Balancing Resource Use Efficiency against Performance
While overcommitment can optimize the use of your hardware resources, it's critical to manage this strategy carefully to ensure that all customers experience a high level of service. With multiple virtual servers sharing the same underlying physical server, performance can become an issue if the capacity of the server is overassigned to the point where individual virtual servers can't get enough processing power.

Improving Resource Efficiency with Parallels Containers’ File System and Templates
Both Parallels Container’s file system and its templates help you use your existing resources more efficiently, so you can do more without increasing your hardware investment.
Balancing Resource Use Efficiency against Performance

Parallels Cloud Server features a new format that stores all of a container’s data in a single image; the image has its own private file system journal. This format is implemented through a new storage technique called ploop (Parallels loopback device). By giving each container its own private file, ploop eliminates the problem of random file access to multiple files. This results in faster migration and backup, as well as enhanced performance. Other benefits gained from storing all of a container’s data in a single image include:

- The ability to take consistent snapshots of the container file system, which you can use for incremental backups or provide to end users for test purposes.
- The ability to make backups on top of snapshots using conventional cp or tar, rather than requiring special backup tools like Acronis True Image.
- A significant reduction in the time required to back up and migrate containers, since sequential reading of a single image is an order of magnitude faster than reading multiple separate small files on rotational drives.

With ploop each container stores its files in its own private disk location. In order to prevent this from causing excessive IO operations and memory usage by similar containers, Parallels Cloud Server also provides a deduplication mechanism described in the next section.

Memory and IOPS Deduplication (Containers)

Previous releases of Parallels Cloud Server implemented deduplication of container data via the VZFS file system, which centralized storage of template data on each hardware node and linked the container’s shared files to it. When containers modified files, a procedure called Copy-On-Write gave them their own private copy of the template files.

Parallels Cloud Server 6 has a more robust approach. It collects file-usage statistics in real-time to determine which files are most frequently accessed by containers, and puts commonly used files in cache. That way, if a container needs to access a commonly used file, it can go straight to the cache instead of the disk — an approach that improves performance by reducing the number of I/O operations. For even better performance, the cache can be placed on a solid-state drive (SSD).

The benefits of this approach to deduplication are multiple. The approach is:
• **Reliable and robust**, since it removes the need to download old templates (which may no longer be available) in order to restore older versions of containers.

• **Configurable**, with the ability to apply parameters to specify factors such as cache size, directories to be used for caching, and expiration times for cached files.

• **Always up to date**, because a daemon that tracks file access ensures that the cache always has the most frequently accessed data.

• **Easy to keep at top performance**, because it comes with a configuration tool that lets you modify various options to enhance performance.

### How Parallels Container Templates Improve Memory Efficiency

A template (or package set) in Parallels Containers is a set of original application files repackaged for rapid container creation. Using templates within containers can greatly improve the efficiency of server memory utilization. Parallels Containers provides tools for creating templates, installing them, upgrading them, adding them to a container, and removing them from a container.

Using templates lets you:

• Share RAM among similar applications running in different containers, saving hundreds of megabytes of memory.

• Share the files that make up a template among different containers, saving gigabytes of disk space.

• Reduce IOPS by allowing containers which access the same file from a template to read it only one time from the disk and then allow all other containers to access it from memory.

Parallels Containers has two types of templates:

• **OS templates**, which consist of an operating system plus the standard set of installation applications. Parallels Containers uses OS templates to create new containers with a preinstalled operating system.

• **Application templates**, which consist of a set of repackaged software packages – optionally accompanied by configuration scripts. Parallels Containers uses application templates to add extra software to an existing container. For example, you can create a container on the basis of the CentOS 5 OS template and add MySQL to it using the MySQL application template.
For detailed information on PCS templates, see the “Parallels Cloud Server Templates Management Guide.”

Summary of Best Practices for Maximizing Density
Wrapping up what we've covered so far, here are the best practices you'll want to follow in order to maximize resource utilization and scale your Parallels Containers servers most effectively:

- **Determine the appropriate overcommit level for each type of VPS offering.** Because larger, more expensive VPS offerings are typically used at or near their resource limits much more frequently than smaller offerings, you should segment your offers and related service plans based on CPU and memory allocations. That way, you can provision larger VPSs on servers with lower overcommit levels, and smaller VPS on servers with higher overcommit levels.

- **Determine what level of overcommit you're comfortable with.** The higher the degree of overcommitment, the more intensively you'll need to monitor the performance of your physical hardware and containers.

- **Make sure your implementation plans include all of the following:**
  - Monitoring usage of CPU, RAM, and disk I/O on your host hardware.
  - Ensuring that the CPU cores and limits assigned to each container are consistent with the actual CPU configuration installed in the server hardware.
  - Managing your disk I/O limits as described in Parallels Disk I/O Limits Best Practices white paper.
  - Using VSwap to set the amount of physical memory and virtual swap space allocated per container (see VSwap Best Practices white paper for guidelines).
  - Provisioning VPSs via templates to ensure the most efficient use of memory.

- **Monitor the server as the number of deployed VPSs increases to ensure that you're maintaining quality of service.** As you approach the server's resource limits, you have several choices:
  - Stop provisioning additional VPSs on the server.
  - Upgrade the server with additional resources.
  - Move containers with a high level of resource usage (e.g., high disk I/O) to another hardware node with lower overcommit levels.
  - Contact customers whose VPS are approaching their allocated limits and upsell them, adding more RAM or CPU to their subscriptions.
Real-World Examples
While maintaining the optimal number of virtual containers on each physical node is critical to being able to both deliver a high quality of service and leverage your infrastructure investment efficiently, there is no one answer as to what is the optimal number. Our partners deploy a wide range of density levels, depending on the type of service they’re offering. Some focus on reselling large, powerful VPSs to web hosters who want to maximize the number of shared accounts on each container, and therefore maintain lower densities per node. Others use smaller VPSs for testing applications, offering hosted desktops, or providing backup services, and therefore are able to support much higher densities.

Typical Server Configurations
We recently conducted a survey of Parallels partners to determine the range of hardware configurations they are currently deploying. Although the results differed depending on the target markets each service provider was addressing, we found many service providers achieving success with the configurations described in Table 2.
### Table 2: PCS Best Practice Configurations

<table>
<thead>
<tr>
<th>CPU</th>
<th>GB RAM</th>
<th>GB HDD</th>
<th>No. of Containers</th>
<th>VPS Offering and Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>One dual-quad AMD Operon</td>
<td>16</td>
<td>4x500GB RAID 10</td>
<td>7-30</td>
<td>L: 1GB RAM, 20 GB HDD XL: 2GB RAM, 40 GB HDD XXL: 4 GB RAM, 80GB HDD</td>
</tr>
<tr>
<td>One dual-quad CPU AMD Operon</td>
<td>6-32</td>
<td>4x500GB RAID 10</td>
<td>10-30</td>
<td>L: 1GB RAM, 20 GB HDD XL: 2GB RAM, 40 GB HDD XXL: 4GB RAM, 80GB HDD</td>
</tr>
<tr>
<td>i7 (2x8) Xeon X5550 or i7 (2x8 or 2x4) Xeon Q9550</td>
<td>12-32</td>
<td>4x500GB RAID 10</td>
<td>20-60</td>
<td>HVD Standard: 1 GB RAM, 2GB HDD HVD Professional: 2GB RAM, 10GB HDD</td>
</tr>
<tr>
<td>i7 (2x8 of 2x6) Xeon X5550</td>
<td>128</td>
<td>4x500GB RAID 10</td>
<td>80-110</td>
<td>L: 1GB RAM, 50GB HDD XL: 2 GB RAM, 100GB HDD XXL: 2 GB RAM, 100GB HDD</td>
</tr>
<tr>
<td>i7 (2x8) Xeon X5550</td>
<td>48</td>
<td>4x500GB RAID 10</td>
<td>40-50</td>
<td>2GB RAM, 20GB HDD</td>
</tr>
<tr>
<td>i7 (2x8) 9200</td>
<td>72</td>
<td>4x500GB RAID 10</td>
<td>70-80</td>
<td>L: 1GB RAM, 30GB HDD XL: 2GB RAM, 50GB HDD XXL: 4GB RAM, 100GB HDD</td>
</tr>
</tbody>
</table>
Optimizing Your Server Configuration and Management

Providing customers with reliable service – which means an absolute minimum of service disruptions – is one of the most important factors in service providers’ success. At the same time, to enhance profitability, you have to minimize your maintenance costs. Achieving both goals can be difficult – but not when you use Parallels Cloud Server.

Best Practices for Maximizing Quality of Service and Profitability
The following best practices will help you ensure a high quality of service while also enhancing your profitability:

- **Optimize your environment.** Key steps you can take to do so include:
  - Using compatible OS environments on your host machine – for example, deploying Parallels Cloud Server – to ensure optimum performance and reliability.
  - Using the same hardware in all your nodes. Including disk controllers and other peripherals, will reduce the risk of driver-related problems.
  - Avoiding the use of third-party drivers on any of your nodes. In our experience, many third-party drivers have not been validated across all the variations of hardware and software that can be deployed on a machine, and therefore can negatively affect system reliability.
  - Keeping all configurations synchronized across all your Parallels Containers servers, to simplify troubleshooting and maintenance.

- **Streamline deployment.** Parallels Cloud Server (PCS) streamlines deployment by enabling you to use a single disk to deploy the complete software stack, including the operating system and containers, on a bare-metal server. This approach not only speeds installation, but also improves reliability, by ensuring that all machines are configured consistently. PCS also lets you provide VMs for those customers that want to maintain their own operating system on the server. You can also streamline deployment by automating network-based installation – a step made possible by the compatibility between PCS and PXE boot.

- **Reduce the impact of disk failures.** Hard drives are typically the most unreliable server component. Using hot-plug-compatible disk bays and battery backup on RAID controllers will reduce the impact of hard drive failures. Also, avoid the use of fans on
controllers, as they decrease reliability – and when they fail, they can generate data failures.

- **Optimize hard drive configuration.** The most commonly deployed hard drive configuration today is a 4x500GB RAID 10 array, which provides you with about 1TB of usable storage on the server. For most of our partners, this is a cost-effective configuration for initial deployment and allows for significant VPS storage growth. You should also configure the swap space so that when the resource load on the machine begins to approach the limit of allowable memory, the swap space will be big enough to handle the extra load. Table 3 shows our recommendations for configuring swap space.

<table>
<thead>
<tr>
<th>Memory</th>
<th>Recommended Swap space</th>
</tr>
</thead>
<tbody>
<tr>
<td>4GB of RAM or less</td>
<td>a minimum of 2GB of swap space</td>
</tr>
<tr>
<td>4GB to 16 GB of RAM</td>
<td>a minimum of 4GB of swap space</td>
</tr>
<tr>
<td>16GB to 64GB of RAM</td>
<td>a minimum of 8GB of swap space</td>
</tr>
<tr>
<td>64GB to 256GB of RAM</td>
<td>a minimum of 16GB of swap space</td>
</tr>
<tr>
<td>256GB to 512GB of RAM</td>
<td>a minimum of 32GB of swap space</td>
</tr>
</tbody>
</table>

**Real-Word Examples of VPS Configurations**
Many different VPS configurations are deployed in the market, and no one configuration, price, infrastructure, or service level is appropriate for every application. Table 4 provides a
few real-world examples from our service provider partners, illustrating the wide range of options, hardware sizes, and densities that can be effectively deployed with Parallels Containers. The small VPS configuration achieves very high densities with a high level of service by carefully managing and monitoring resource usage on the server. The large VPS configuration achieves lower VPS densities but still requires careful management of overcommitted resources to ensure a high quality of service. The third configuration is unique in that it allows customers to deploy as many VPS instances as they like without a per-container fee, charging instead based on total RAM usage across all containers.

Table 4: Three Real-World Offerings and Their Underlying VPS Configurations

<table>
<thead>
<tr>
<th>Component</th>
<th>Small VPS</th>
<th>Large VPS</th>
<th>VPS Based on Innovative Pricing Paradigm</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VPS Offering &amp; Features</strong></td>
<td>512 MB RAM</td>
<td>4 GB RAM</td>
<td>50 GB storage</td>
</tr>
<tr>
<td></td>
<td>20 GB storage</td>
<td>650 GB storage</td>
<td>1TB network transfer per month</td>
</tr>
<tr>
<td></td>
<td>100 GB network transfer per month</td>
<td>1 TB network transfer per month</td>
<td>No per-VPS fee</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Free load balancing and Control panel</td>
</tr>
<tr>
<td><strong>VPS Pricing</strong></td>
<td>$30 per VPS</td>
<td>$90 per VPS</td>
<td>$35 for 256 MB RAM</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$70 for 512 MB RAM</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$105 for 1 GB RAM</td>
</tr>
<tr>
<td><strong>Processor</strong></td>
<td>Dual-quad core</td>
<td>16-core Intel Xeon</td>
<td>Dual-quad-core</td>
</tr>
<tr>
<td><strong>CPU</strong></td>
<td>2.4 GHz</td>
<td>2.13 GHz</td>
<td>2.4 GHz</td>
</tr>
</tbody>
</table>
**Conclusion**

Leveraging the best practices detailed in this report – both for achieving maximum density per server and for optimizing server configuration and management – will enable your business to increase the revenue per server that your VPS offerings generate. And by diversifying your VPS offerings and aligning them optimally with your hardware configurations, management processes, and capabilities, you will differentiate yourself from your competition, enabling you to grow your business as well as maximize your profitability.
Appendices: Checklist for Optimizing Your VPS Infrastructure

☐ Have you set CPU limits, disk I/O, memory parameters, and VSwap settings appropriately?
☐ Have you reviewed the “VSwap Best Practices” whitepaper?
☐ Have you limited disk throughput per container and aligned it to your network bandwidth?
☐ Have you reviewed the “Disk I/O Best Practices” white paper?
☐ Have you set disk space and network I/O appropriately for your environment?
☐ Are your VPS offerings mapped appropriately to the underlying container configuration?
☐ Have you documented your level of overcommitment and mapped it to your specific hardware?
☐ Have you set up the necessary tools and approaches for monitoring usage of key resources?
☐ Do you continue to monitor the server as you scale your VPS levels?
☐ Have you reached out to customers approaching their allocated limits and upsold them on additional VPS resources?
☐ Have you minimized driver-related problems by using the same hardware in all nodes?
☐ Have you leveraged PSBM to streamline deployment?
☐ Do you provide VMs for customers that want to maintain their own operating system on the server?
☐ Do you use PXE Boot to automate network-based installation?
☐ Have you avoided fan usage in your controllers and implemented hot-plug-compatible disk bays and battery backup on your RAID controllers?
☐ Have you optimized your hard drive configuration and swap levels in accordance with best practices?
☐ Have you set the appropriate swap space for your memory levels?
☐ Have you implemented clustering to improve reliability and let you deliver high-performance services?
☐ Have you reviewed “PCS Cluster Guide”?
☐ Do you have multiple VPS offerings, aligned to your underlying technology capabilities?
☐ Have you introduced innovative pricing paradigms or other unique approaches?