EMC® ScaleIO®

Version 2.0.x

Performance Fine-Tuning Technical Notes

P/N 302-002-680

REV 04

EMC²

Copyright © 2016-2017 Dell Inc. or its subsidiaries. All rights reserved.

Published March 2017

Dell believes the information in this publication is accurate as of its publication date. The information is subject to change without notice.

THE INFORMATION IN THIS PUBLICATION IS PROVIDED "AS-IS." DELL MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WITH RESPECT TO THE INFORMATION IN THIS PUBLICATION, AND SPECIFICALLY DISCLAIMS IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. USE, COPYING, AND DISTRIBUTION OF ANY DELL SOFTWARE DESCRIBED IN THIS PUBLICATION REQUIRES AN APPLICABLE SOFTWARE LICENSE.

Dell, EMC, and other trademarks are trademarks of Dell Inc. or its subsidiaries. Other trademarks may be the property of their respective owners. Published in the USA.

EMC Corporation
Hopkinton, Massachusetts 01748-9103
1-508-435-1000 In North America 1-866-464-7381
www.EMC.com

CONTENTS

Figures 5

Preface 7

Chapter 1 Overview 9
Introduction 10
Tuning ScaleIO for Best Performance 10

Chapter 2 ScaleIO Performance Tuning 11
Performance tuning pre-installation 12
Performance tuning post-installation 12
Upgrades 12
Tuning considerations 12

Chapter 3 ScaleIO System Changes 15
Using the set_performance_parameters utility for MDM and SDS 16
Caching Updates for ScaleIO 2.x 17
Read RAM Cache settings for SDS 17
Read Flash Cache settings for SDS 18
Manual SDC configuration/verification 19
Jumbo Frames and the potential impact on performance 20
Jumbo Frame configuration for Linux 20
Jumbo Frame configuration for Windows 21
Jumbo Frame configuration for ESXi 23
Jumbo Frame configuration for ScaleIO Virtual Machine (SVM) 23

Chapter 4 Operating System Tuning Recommendations 25
Optimizing ESXi 26
Optimizing Linux 26
Optimizing the ScaleIO Virtual Machine (SVM) 26
Optimizing VM guests 27
I/O scheduler 27
Paravirtual SCSI controller 28
Queue length 28
Optimizing Windows 29

Chapter 5 Appendix 31
ScaleIO Performance Parameters 32

FIGURES

1 Test ping to ensure mtu setting....21
2 Show interface output.... 21
3 Adapter properties....22
4 Ping output....22
5 Test ping to ensure mtu setting....24
6 Virtual Machine Properties.... 27
7 Customize Power Plan....30

Preface

As part of an effort to improve its product lines, EMC periodically releases revisions of its software and hardware. Therefore, some functions described in this document might not be supported by all versions of the software or hardware currently in use. The product release notes provide the most up-to-date information on product features.

Contact your EMC technical support professional if a product does not function properly or does not function as described in this document.

Note

This document was accurate at publication time. Go to EMC Online Support (https://support.emc.com) to ensure that you are using the latest version of this document.

Related documentation

The release notes for your version includes the latest information for your product.

The following EMC publication sets provide information about your ScaleIO or ScaleIO Ready Node product:

- ScaleIO software (downloadable as ScaleIO Software <version> Documentation set)
- ScaleIO Ready Node with AMS (downloadable as ScaleIO Ready Node with AMS Documentation set)
- ScaleIO Ready Node no AMS (downloadable as ScaleIO Ready Node no AMS Documentation set)
- VxRack Node 100 Series (downloadable as VxRack Node 100 Series Documentation set)

You can download the release notes, the document sets, and other related documentation from EMC Online Support.

Typographical conventions

EMC uses the following type style conventions in this document:

Bold	Used for names of interface elements, such as names of windows, dialog boxes, buttons, fields, tab names, key names, and menu paths (what the user specifically selects or clicks)
Italic	Used for full titles of publications referenced in text
`Monospace`	Used for: • System code • System output, such as an error message or script • Pathnames, filenames, prompts, and syntax • Commands and options
Monospace italic	Used for variables
`Monospace bold`	Used for user input
[ ]	Square brackets enclose optional values

|	Vertical bar indicates alternate selections - the bar means “or”
{ }	Braces enclose content that the user must specify, such as x or y or z
...	Ellipses indicate nonessential information omitted from the example

Where to get help

EMC support, product, and licensing information can be obtained as follows:

Product information

For documentation, release notes, software updates, or information about EMC products, go to EMC Online Support at https://support.emc.com.

Technical support

Go to EMC Online Support and click Service Center. You will see several options for contacting EMC Technical Support. Note that to open a service request, you must have a valid support agreement. Contact your EMC sales representative for details about obtaining a valid support agreement or with questions about your account.

Your comments

Your suggestions will help us continue to improve the accuracy, organization, and overall quality of the user publications. Send your opinions of this document to techpubcomments@emc.com.

CHAPTER 1

Overview

This chapter includes:

- Introduction 10
- Tuning ScaleIO for Best Performance 10

Introduction

ScaleIO is a software-only solution that uses existing servers' local disks and LAN to create a virtual SAN that has all the benefits of external storage—but at a fraction of the cost and complexity. ScaleIO utilizes the existing local internal storage and turns it into internal shared block storage. For many workloads, ScaleIO storage is comparable to, or better than external shared block storage.

This document describes best practices for maximizing performance in high-performance (more than 60,000 IOPs) ScaleIO v2.x environments.

Tuning ScaleIO for Best Performance

Users can improve ScaleIO system performance in terms of IOPs, latency, and bandwidth, by making environment-specific fine-tunings on the operating system, network, and ScaleIO components. However, newer releases of ScaleIO (versions 2.x and beyond) have been enhanced to include built-in performance tuning features that reduce the number of, and in some cases, completely eliminate many of the manual tasks that were once advised. This document describes these performance-related best practices.

Note

Performance tuning is very case-specific. To prevent undesirable effects, it is highly recommended to thoroughly test all changes. For further assistance, contact https://support.emc.com.

CHAPTER 2

ScaleIO Performance Tuning

This chapter includes:

- Performance tuning pre-installation 12
- Performance tuning post-installation 12

Performance tuning pre-installation

This section describes the necessary steps to take prior to beginning an installation of ScaleIO to enhance performance.

The following table describes performance optimization during the installation. For additional installation information, refer to the *ScaleIO Installation Guide*.

Installation method	Task
Installation Manager (Gateway Installer Utility)	In the CSV file, set `perfProfileFor<SDS/SDC/MDM>= High` See note below. **Note** When installing ScaleIO via gateway, users may set each component (SDS/MDM/SDC) for Normal or High by entering these words in the PerfProfile column per component on each row of the CSV file. This will instruct the installation manager to send the command and set the profile to normal (for default), or high (for high_performance) during the configure stage. More details related to performance profiles are discussed in Tuning considerations on page 12.
VMware deployment wizard	No user action required. Wizard includes the 2.x performance profile plugin.
Manual installation	Refer to ScaleIO specific instructions herein.

Performance tuning post-installation

This section describes steps to take after completing a successful installation to enhance performance.

Upgrades

For any ScaleIO upgrade from 1.3x to 2.x, all performance related settings will need to be reset. Users should implement performance tunings by following the guidelines described in this document.

Tuning considerations

New installation procedures have been incorporated in versions 2.x and above which eliminate many of the manual tasks involved with performance tuning ScaleIO. Users may configure a high performance profile which will change the default parameters. In the past, users were instructed to modify the SDS and/or MDM configuration (conf.txt) files. However, those changes are no longer required. In fact, any changes made to the 2.x configuration files will no longer take effect in ScaleIO.

The main difference between the performance and standard (default) profiles are the amount of server resources (CPU and memory) that are consumed. A performance profile (or configuration) will always consume more resources.

This document will describe commands using the ScaleIO command line interface (scli) to quickly and easily modify the desired performance profile.

Users will achieve optimum performance by always setting the performance profile to High_performance. A complete list of parameters comparing the Default and the High_performance profiles is available in the Appendix.

CHAPTER 3

ScaleIO System Changes

This chapter includes:

- Using the set_performance_parameters utility for MDM and SDS 16
- Caching Updates for ScaleIO 2.x 17
- Read RAM Cache settings for SDS 17
- Read Flash Cache settings for SDS 18
- Manual SDC configuration/verification 19
- Jumbo Frames and the potential impact on performance 20

Using the set_performance_parameters utility for MDM and SDS

New installation procedures have been instituted in versions 2.x and above which eliminate many of the manual tasks involved with performance tuning ScaleIO. Users may configure a High_performance profile which will change the default parameters.

The following table describes the commands for specific tasks:

Task	Command
To change the performance profile from Default to High_performance	Execute the command: `scli --set_performance_parameters --all_sds --all_sdc --apply_to_mdm --profile high_performance`
To change the profile back to normal defaults	Execute the command: `scli --set_performance_parameters --all_sds --all_sdc --apply_to_mdm --profile default`
To view current settings	Execute the command: `scli --query_performance_parameters`
To view full parameter settings for an MDM	Execute the command: `scli -- query_performance_parameters -- print_all`
To view full parameter settings of a specific SDS (this also shows the MDM settings)	Execute the command: `scli -- query_performance_parameters -- sds_name <NAME> --print_all`
To view full parameter settings of a specific SDC (this also shows the MDM settings)	Execute the command: `scli -- query_performance_parameters -- sdc_name <NAME> --print_all`

Note

Refer to the ScaleIO Performance Parameters on page 32 for a list containing all default and performance profile parameters.

Caching Updates for ScaleIO 2.x

ScaleIO offers the following types of caching, for the purpose of enhancing system performance:

- RAM Read Cache (using a server's DRAM memory)
- Read Flash Cache (using SSDs or flash PCI cards)

Note

SSDs used for caching cannot be used for storage purposes.

The following table summarizes information about the caching modes provided by the system.

Mode	Description	Considerations	Default Setting
RAM Read Cache (rmcache)	Read-only caching performed by server RAM.	RAM Read cache, the fastest type of caching, uses RAM that is allocated for caching. Its size is limited to the amount of allocated RAM.	Enabled
Read Flash Cache (RFCache)	Read-only caching performed by one or more dedicated SSD devices or flash PCI drives in the server.	Read Flash Cache uses the full capacity of SSD or flash PCI devices (up to eight) to provide a larger footprint of read-only LRU (least recently used) based-caching resources for the SDS. This type of caching reacts quickly to workload changes to speed up HDD Read performance. Several SSD devices can be allocated to a shared cache pool, and therefore the cache size is limited in size only to the amount of SSDs allocated for this purpose. **Note** The RFCache driver must be installed during deployment. Caching devices can be defined either during the installation process, or after deployment.	Disabled

Read RAM Cache settings for SDS

In version 2.x Read RAM Cache is enabled by default on the Storage Pool. Recommendation: disable it for SSD/Flash pools. For HDD pools, Read RAM Cache can help increase performance. If the node is storage only (in other words; is the node is only used for ScaleIO), then the recommendation is to turn on Read RAM Cache for HDD pools and use as much of the server DRAM as possible.

In a converged configuration (where ScaleIO is sharing the server with other applications), depending on the available DRAM resources, it may also help to turn on Read RAM Cache for HDD pools and increase the cache size from the default.

If users wish to enable/disable Read RAM Cache, perform either of the following steps. Read RAM Cache may be enabled/disabled on the Protection Domain, or for each SDS in the cluster.

Task	Command
To enable Read RAM cache	Run one of the following commands: • `scli --set_rmcache_usage --protection_domain_name <domain NAME> --storage_pool_name <pool NAME> --use_rmcache [--dont_use_rmcache]` • `scli --enable_sds_rmcache [--disable_sds_rmcache] --sds_name <NAME>` **Note** Using this command would be required for every SDS in the cluster.

To increase the amount of Read RAM cache	Run the following command:
	`Usage: scli --set_rmcache_size --sds_name <NAME> --protection_domain_name <NAME> --rmcache_size_mb <SIZE>`
	Where --rmcache_size_mb is the size of rmcache in MB, and the range is between 128MB-300GB.
	It is important to ensure that Read RAM cache is enabled at all levels (PD, SP, and SDS). When rmcache is properly enabled, query output will look like this:
	`[root@116BT-18 ~]# scli --query_sds --sds_name SDS48`
	`SDS 76e6c1bb00000002 Name: SDS48 Version: 2.0.906`
	`Protection Domain: dc2fe5d500000000, Name: domain1`
	`DRL mode: Volatile`
	`Authentication error: None`
	`IP information (total 1 IPs):`
	`1: 172.16.2.48 Role: All (SDS and SDC)`
	`Port: 7072`
	`RAM Read Cache information:`
	`128.0 MB (131072 KB) total size`
	`Cache is enabled`
	`RAM Read Cache memory allocation state is SUCCESSFUL`

Read Flash Cache settings for SDS

Read Flash Cache is available in version 2.x and above. This feature is a Read Cache used to increase read performance and buffers writes to increase the performance of Read-after-Write I/Os. It allows users to create and configure an "RFcache" device for any SSD or Flash card. ScaleIO will cache user data depending on the mode selected. This feature can greatly improve performance for specific workloads. The RFcache device is also referred to as an accelerated device.

To create an RFcache device and configure it, use the following steps:

1. Create an RFcache device (Recommendation: create 1 device per SDS).

```
scli --add_sds_rfcache_device --sds_name <NAME> --
rfcache_device_path <device_path> --rfcache_device_name
<RFcache device NAME>
```

2. Set the RFcache parameters (Recommendation: these parameters have a great impact on performance, therefore use the defaults).

```
scli --set_rfcache_parameters --protection_domain_name <domain
NAME> --rfcache_pass_through_mode pass_through_write_miss
```

Note

The default settings are; Passthrough mode = Write_Miss, Page Size 64 KB, Max IO size 128 KB.

3. Enable acceleration of a Storage Pool—accelerate all SDS devices that are in the pool:

```
scli  --set_rfcache_usage  --protection_domain_name <domain
NAME>  --storage_pool_name <pool NAME> --use_rfcache
```

For Read Flash Cache, the available modes are as follows:

- pass_through_none
- pass_through_read
- pass_through_write
- pass_through_read_and_write
- pass_through_write_miss

The default caching mode is “write-miss”. In this mode, it is essentially a write-through option where only reads and updates are cached. This mode buffers writes to the data that was already in cache.

For more information related to using and configuring Read Flash Cache, refer to *ScaleIO Deployment Guide* and *ScaleIO User Guide* at https://support.emc.com.

Manual SDC configuration/verification

If there are problems connecting to a ScaleIO system after a reboot, users can modify the `drv_cfg.txt` file or use the drv_cfg utility to ensure that the SDC reconnects to the MDM automatically upon node restart/reboot.

To do this, perform one of the following procedures:

- Edit `/bin/emc/scaleio/drv_cfg.txt` and change the #MDM line to include each IP address for any node(s) containing an MDM in your cluster.
  For example, change the line:

  From:

  #mdm 10.20.30.40

  To:

mdm 10.108.158.48,10.108.158.49

where .48 is the master node, and .49 is the slave node. The Tie-Breaker node should not be included.

- Use the drv_cfg binary or drv_cfg.exe on Windows to re-attach the MDM.
- Rescan all volumes using the drv_cfg utility.

Jumbo Frames and the potential impact on performance

When enabling jumbo frames, one can expect approximately 10% improvement in performance if all network components fully support jumbo frames. If some network components do not fully support jumbo frames, it is recommended to use the default setting; mtu 1,500.

Prior to activating mtu settings on the logical level, set Jumbo frames = mtu 9000 on the physical switch ports that are connected to the server. Failure to do so may lead to network "disconnects" and packet drops.

Refer to your relevant vendor guidelines on how to configure jumbo frame support.

Jumbo Frame configuration for Linux

Configure Jumbo Frames for NIC cards in the Linux-based ScaleIO servers.

Perform the following steps, for all the NIC cards in the ScaleIO system:

Procedure

1. Run the `ifconfig` command to get the NIC information.
2. Depending on the OS, run the command:

Operating System	Command
RHEL/CentOS	Edit `/etc/sysconfig/network-scripts/ifcfg-`*<NIC_NAME>*
SLES	Edit `/etc/sysconfig/network/ifcfg-`*<NIC_NAME>*

3. Add parameter `mtu=9000` to the file.
4. To apply the changes, type: **`service network restart`**
5. Execute `ifconfig` again to verify that the settings have been changed.
6. To test the command, type: **`ping -M do -s`** `8972`*<DESTINATION_IP_ADDRESS>*

   Output should look similar to this:

Figure 1 Test ping to ensure mtu setting

```
[root@116BT-18 network-scripts]# ping -M do -s 8972 10.108.158.48
PING 10.108.158.48 (10.108.158.48) 8972(9000) bytes of data.
8980 bytes from 10.108.158.48: icmp_seq=1 ttl=64 time=0.031 ms
8980 bytes from 10.108.158.48: icmp_seq=2 ttl=64 time=0.027 ms
8980 bytes from 10.108.158.48: icmp_seq=3 ttl=64 time=0.026 ms
8980 bytes from 10.108.158.48: icmp_seq=4 ttl=64 time=0.026 ms
8980 bytes from 10.108.158.48: icmp_seq=5 ttl=64 time=0.027 ms
8980 bytes from 10.108.158.48: icmp_seq=6 ttl=64 time=0.026 ms
8980 bytes from 10.108.158.48: icmp_seq=7 ttl=64 time=0.026 ms
8980 bytes from 10.108.158.48: icmp_seq=8 ttl=64 time=0.029 ms
8980 bytes from 10.108.158.48: icmp_seq=9 ttl=64 time=0.027 ms
8980 bytes from 10.108.158.48: icmp_seq=10 ttl=64 time=0.026 ms
8980 bytes from 10.108.158.48: icmp_seq=11 ttl=64 time=0.012 ms
8980 bytes from 10.108.158.48: icmp_seq=12 ttl=64 time=0.026 ms
8980 bytes from 10.108.158.48: icmp_seq=13 ttl=64 time=0.026 ms
8980 bytes from 10.108.158.48: icmp_seq=14 ttl=64 time=0.027 ms
^C
--- 10.108.158.48 ping statistics ---
14 packets transmitted, 14 received, 0% packet loss, time 13581ms
rtt min/avg/max/mdev = 0.012/0.025/0.031/0.007 ms
```

Jumbo Frame configuration for Windows

Configure Jumbo Frames for NIC cards in the Windows-based ScaleIO servers.

Perform the following steps, for all the NIC cards in the ScaleIO system:

Procedure

1. Change the `Maximum Transmission Unit (mtu)` setting to `9,000`, or the highest value that is supported by the switch and the connected nodes.
2. Determine the appropriate NIC name by typing the command:

   ```
   netsh interface ipv4 show interface
   ```

   Output similar to the following appears:

   Figure 2 Show interface output

   ```
   Idx     Met         MTU          State                Name
   ---  ----------  ----------  ------------  ---------------------------
     1          50  4294967295  connected     Loopback Pseudo-Interface 1
    17           5        1500  connected     10G_Data
    18           5        1500  connected     10G_Mgmt
   ```

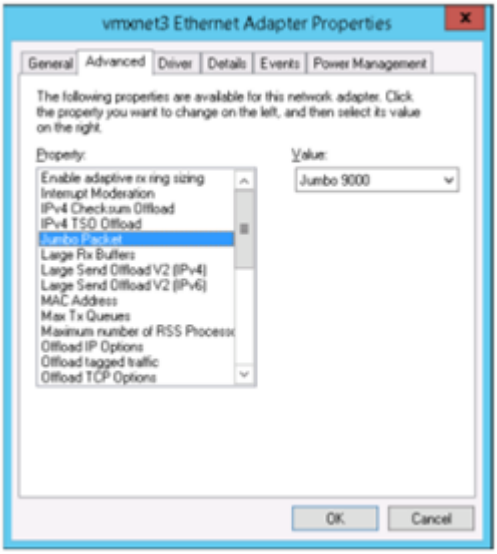
   In this example, index 17 is the appropriate network.
3. Type the command:

   ```
   netsh interface ipv4 set subinterface <network_ID> mtu=9000
   store=persistent
   ```

   where, *network_ID* is the ID from the output in the previous step; in this case, the ID is 17.

4. In the **Advanced** tab of the **Adapter Properties** dialog for your vendor and driver, change the value of `Jumbo Packet` to *9000*, as illustrated in the following figure:

Figure 3 Adapter properties



5. Click **OK**.

   The network connection may disconnect briefly during this phase.

6. Verify that the configuration is working, by typing the command:

```
ping -f -l 8972 <Destination_IP_Address>
```

   Output similar to the following should be displayed:

Figure 4 Ping output

```
C:\>ping -f -l 8972 9.99.101.12

Pinging 9.99.101.12 with 8972 bytes of data:
Reply from 9.99.101.12: bytes=8972 time<1ms TTL=64
Reply from 9.99.101.12: bytes=8972 time<1ms TTL=64
Reply from 9.99.101.12: bytes=8972 time<1ms TTL=64
Reply from 9.99.101.12: bytes=8972 time<1ms TTL=64

Ping statistics for 9.99.101.12:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms
```

Note

Ensure that the switch supports 10 GB Ethernet.

Jumbo Frame configuration for ESXi

Configure Jumbo Frames for NIC cards in the ESXi-based ScaleIO servers.

Perform the following steps, for all the NIC cards in the ScaleIO system:

Procedure

1. Change the `Maximum Transmission Unit (mtu)` setting to `9,000` on the vSwitches and on the SVM (be sure to make the change in `/etc/sysconfig/network/ifcfg-ethX)`:

   a. Type the command:

   ```
   esxcfg-vswitch -m 9000 <vSwitch>
   ```

   b. Create VMKernel with jumbo frames support by typing the following commands:

   a.
   ```
   esxcfg-vswitch -d
   ```

   b.
   ```
   esxcfg-vswitch -A vmkernel# vSwitch#
   ```

   c.
   ```
   esxcfg-vmknic -a -i <ip address> -n <netmask> -m 9000
   <portgroup name>
   ```

Note

Changing Jumbo Frames on a vSwitch will not change the VMKernel MTU size. For older vCenter versions, check to ensure the MTU setting has been changed. If not successful, users may need to delete and recreate the VMKernel. For newer vCenter versions, modify the MTU of VMKernel by using the vsphere web client.

Jumbo Frame configuration for ScaleIO Virtual Machine (SVM)

Configure Jumbo Frames for NIC cards in the ScaleIO Virtual Machine (SVM)-based ScaleIO servers.

Perform the following steps, for all the NIC cards in the ScaleIO system:

Procedure

1. Edit the `/etc/sysconfig/network/ifcfg-<NIC_NAME>`.
2. Add parameter `mtu=9000` to the file.
3. To apply the changes type:

   ```
   service network restart
   ```

4. Execute `ifconfig` command again to confirm that the settings have been changed.

5. To test the command type:

```
ping -M do -s 8972 <DESTINATION_IP_ADDRESS>
```

Output should look similar to the following:

Figure 5 Test ping to ensure mtu setting

```
[root@116BT-18 network-scripts]# ping -M do -s 8972 10.108.158.48
PING 10.108.158.48 (10.108.158.48) 8972(9000) bytes of data.
8980 bytes from 10.108.158.48: icmp_seq=1 ttl=64 time=0.031 ms
8980 bytes from 10.108.158.48: icmp_seq=2 ttl=64 time=0.027 ms
8980 bytes from 10.108.158.48: icmp_seq=3 ttl=64 time=0.026 ms
8980 bytes from 10.108.158.48: icmp_seq=4 ttl=64 time=0.026 ms
8980 bytes from 10.108.158.48: icmp_seq=5 ttl=64 time=0.027 ms
8980 bytes from 10.108.158.48: icmp_seq=6 ttl=64 time=0.026 ms
8980 bytes from 10.108.158.48: icmp_seq=7 ttl=64 time=0.026 ms
8980 bytes from 10.108.158.48: icmp_seq=8 ttl=64 time=0.029 ms
8980 bytes from 10.108.158.48: icmp_seq=9 ttl=64 time=0.027 ms
8980 bytes from 10.108.158.48: icmp_seq=10 ttl=64 time=0.026 ms
8980 bytes from 10.108.158.48: icmp_seq=11 ttl=64 time=0.012 ms
8980 bytes from 10.108.158.48: icmp_seq=12 ttl=64 time=0.026 ms
8980 bytes from 10.108.158.48: icmp_seq=13 ttl=64 time=0.026 ms
8980 bytes from 10.108.158.48: icmp_seq=14 ttl=64 time=0.027 ms
^C
--- 10.108.158.48 ping statistics ---
14 packets transmitted, 14 received, 0% packet loss, time 13581ms
rtt min/avg/max/mdev = 0.012/0.025/0.031/0.007 ms
```

CHAPTER 4

Operating System Tuning Recommendations

This chapter includes:

- Optimizing ESXi 26
- Optimizing Linux 26
- Optimizing the ScaleIO Virtual Machine (SVM) 26
- Optimizing VM guests 27
- Optimizing Windows 29

Optimizing ESXi

To improve I/O concurrency, users may increase the per device queue length value on a per data store basis. Per device queue length is referred to as “No of outstanding IOs with competing worlds” in the display output.

Use the following command to increase the queue length;

```
esxcli storage core device set -d <DEVICE_ID> -O <Outstanding IOs>
```

where, *<Outstanding IOs>* can be a number ranging from 32-256 (the default=32).

Example:

```
esxcli storage core device set -d eui.
16bb852c56d3b93e3888003b00000000 -O 256
```

Optimizing Linux

When using the SSD devices, it is recommended that the I/O scheduler of the devices be modified.

Type the following on each server, for each SDS device:

```
echo noop > /sys/block/<device_name>/queue/scheduler
```

For example:

```
echo noop > /sys/block/sdb/queue/scheduler
```

Note

To make these changes persistent after reboot, either create a script that runs on boot, or change the kernel default scheduler via kernel command line.

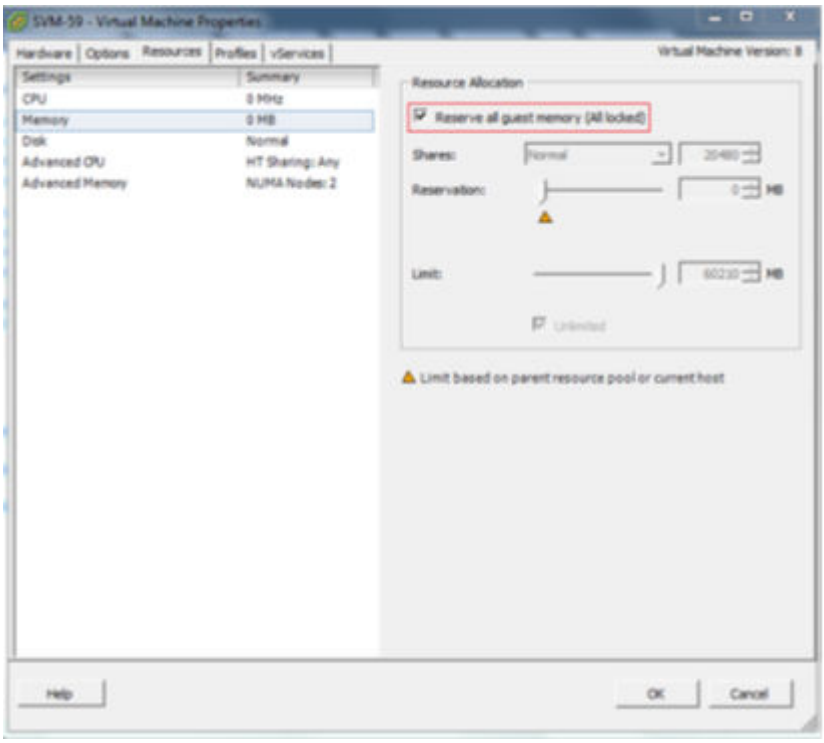
Optimizing the ScaleIO Virtual Machine (SVM)

Procedure

1. Double the CPU and memory assigned to the SVM.
   - In general, 8 vCPUs and 4 GB of memory are sufficient, but this may vary in your environment.
   - 8 vCPUs should use 1 virtual socket and 8 cores per socket.
2. From the **Resources** tab of the **Virtual Machine Properties** window, select **Reserve all guest memory (All locked)**.

   The **Virtual Machine Properties** window is displayed:

Figure 6 Virtual Machine Properties



Optimizing VM guests

I/O scheduler

When using SSD devices, it is recommended that you modify the devices' I/O scheduler.

Type the following on each server, for each SDS device:

```
echo noop > /sys/block/<device_name>/queue/scheduler
```

Example:

```
echo noop > /sys/block/sdb/queue/scheduler
```

Note

For most Linux distributions, NOOP is not the default. Different Linux versions have different default values. For RHEL7 and SLES 11/12, the default value is deadline, but for older versions, the default is CFQ.

Paravirtual SCSI controller

The Paravirtual SCSI (PVSCSI) controller should be used on guest VMs for high performance. It is important that users choose the correct PVSCSI controller, because choosing the wrong controller can adversely affect performance.

Current PVSCSI queue depth default values are 64 for the device and 254 for the adapter. Users can increase the PVSCSI queue depth to 256 for the device and 1024 for the adapter inside a Windows virtual machine.

Windows virtual machine:

1. From the command line run:

```
REG ADD HKLM\SYSTEM\CurrentControlSet\services\pvscsi\Parameters
\Device /v DriverParameter /t REG_SZ /d
"RequestRingPages=32,MaxQueueDepth=254"
```

2. Reboot the virtual machine.
3. Verify the successful creation of a registry entry:
   a. Open the registry editor by running the REGEDIT command from the command line.
   b. Browse to `HKLM\SYSTEM\CurrentControlSet\services\pvscsi \Parameters\Device`.
   c. Verify that the DriverParameter key exists with a value of RequestRingPages=32, MaxQueueDepth=254.

Linux guests:

1. Edit the `vmw_pvscsi.conf` file:

```
echo "options vmw_pvscsi cmd_per_lun=254 ring_pages=32" > /etc/
modprobe.d/vmw_pvscsi.conf
```

2. Re-run `vmware-config-tools.pl`:

```
vmware-config-tools.pl
```

You can add up to 4 PVSCSI controllers per guest. Allocating different ScaleIO volumes to different PVSCSI controllers can help realize the maximum potential of guest performance.

It is strongly recommended that you review this VMware Knowledge Base article (article 2053145) so that you can make educated decisions regarding PVSCSI values.

Queue length

To improve I/O concurrency, users may increase the per-device queue length value on a per-data-store basis. Per-device queue length is referred to as "Number of outstanding I/Os with competing worlds" in the display output.

Procedure

1. Increase the queue length:

```
esxcli storage core device set -d <DEVICE_ID> -O <Outstanding
IOs>
```

Where *<Outstanding IOs>* can be a number ranging from 32-256 (default=32).

Example:

```
esxcli storage core device set -d eui.
16bb852c56d3b93e3888003b00000000 -O 256
```

2. Ensure that the settings remain persistent after a reboot:

```
#localcli --plugin-dir /usr/lib/vmware/esxcli/int/ boot
storage restore -paths
```

Note

If you do not run this command, the queue depth setting will revert back to 32 upon reboot.

Optimizing Windows

The delayed acknowledgment feature causes very high latencies for low IO rates. It is highly recommended that the “delayed ack” feature be disabled on every network interface.

To change each SDS interface, excluding the management IP address, edit the following registry entries:

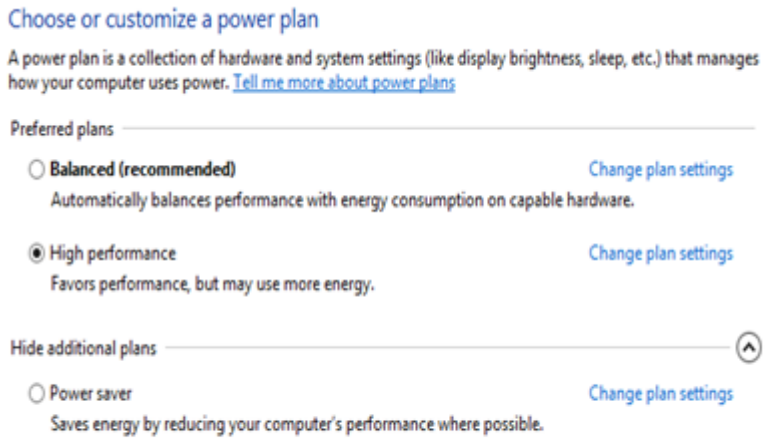
- HKEY_LOCAL_MACHINE \ SYSTEM \ CurrentControlSet \ Services \ Tcpip \ Parameters \ Interfaces *<SAN interface GUID>*
- Entries: TcpAckFrequency TcpNoDelay
- Value type: REG_DWORD, number
- Value to disable: 1

Note

Depending on the Windows version, the method for changing delayed ack varies. Refer to your relevant vendor guidelines for specific details.

Ensure that the `customize power plan` is set to High Performance, as shown in the following figure:

Figure 7 Customize Power Plan



CHAPTER 5

Appendix

This chapter includes:

- ScaleIO Performance Parameters 32

ScaleIO Performance Parameters

The following table describes all values for the v2.x “default” and “high_performance” profiles and is applicable to installations on all of the platforms discussed in this document.

Component	New Name (CLI)	Min Value	Max Value	Default	High Performance
MDM	mdm_net_alloc_rcv_buffer_wait_ms	100	10000	500	500
MDM	mdm_net_break_do_io_loop	0	100	0	5
MDM	mdm_number_sdc_receive_umt	1	100	5	5
MDM	mdm_number_sds_receive_umt	1	100	10	10
MDM	mdm_number_sds_send_umt	1	100	10	10
MDM	mdm_number_sds_keepalive_receive_umt	1	100	10	10
MDM	mdm_sds_capacity_counters_update_interval	1	120	1	1
MDM	mdm_sds_capacity_counters_polling_interval	1	120	5	5
MDM	mdm_sds_volume_size_polling_interval	1	120	15	15
MDM	mdm_sds_volume_size_polling_retry_interval	1	120	5	5
MDM	mdm_number_sds_tasks_umt	1	2048	1024	1024
MDM	mdm_initial_sds_snapshot_capacity	1	10*1024	1024	1024
MDM	mdm_sds_snapshot_capacity_chunk_size	1	50*1024	5*1024	5*1024
MDM	mdm_sds_snapshot_used_capacity_threshold	1	99	50	50
MDM	mdm_sds_snapshot_free_capacity_threshold	101	1000	200	200
MDM	mdm_number_sockets_per_sds_ip	1	8	1	4
MDM	mdm_sds_keepalive_time	2000	3600000	5000	5000
SDS	sds_number_network_umt	2	16	4	8
SDS	sds_tcp_send_buffer_size	4	128*1024	0 (dynamic)	4*1024
SDS	sds_tcp_receive_buffer_size	4	128*1024	0 (dynamic)	4*1024

Component	New Name (CLI)	Min Value	Max Value	Default	High Performance
SDS	sds_max_number_asyncronous_io_per_device	1	2000	64	128
SDS	sds_number_sdc_io_umt	100	500	100	500
SDS	sds_number_sds_io_umt	100	500	100	500
SDS	sds_number_sds_copy_io_umt	100	164	164	164
SDS	sds_number_copy_umt	100	164	164	164
SDS	sds_number_os_threads	1	32	4	8
SDS	sds_number_sockets_per_sds_ip	1	8	1	4
SDS	sds_net_break_do_io_loop	0	100	0	5
SDS	sds_number_io_buffers	1	8	2	4
SDS	sds_net_alloc_rcv_buffer_wait_ms	100	10000	1000	1000
SDS	sdc_tcp_send_buffer_size	4	128*1024	512	4*1024
SDS	sdc_tcp_receive_buffer_size	4	128*1024	512	4*1024
SDS	sdc_number_sockets_per_sds_ip	1	8	1	2
SDS	sdc_number_network_os_threads	1	10	2	8
SDS	sdc_max_inflight_requests	1	10000	100	100
SDS	sdc_max_in_flight_data	1	10000	10	10
SDS	sdc_number_io_retries	1	100	12	12
SDS	sdc_volume_statistics_interval	1000	600000	5000	5000
SDS	sdc_optimize_zero_buffers	0 (FALSE)	1 (TRUE)	0 (FALSE)	0 (FALSE)
SDS	sdc_number_unmap_blocks	1	200	100	100
SDS	sdc_number_non_io_os_threads	1	16	3	3