SPC Benchmark 1™

Full Disclosure Report

MacroSAN Technologies Co ., Ltd
MacroSAN MS3000G2

SPC-1 V3.4.0

Submission Identifier: A31005

Submitted For Review: May 15, 2017

Second Edition – February 2018

Trademarks

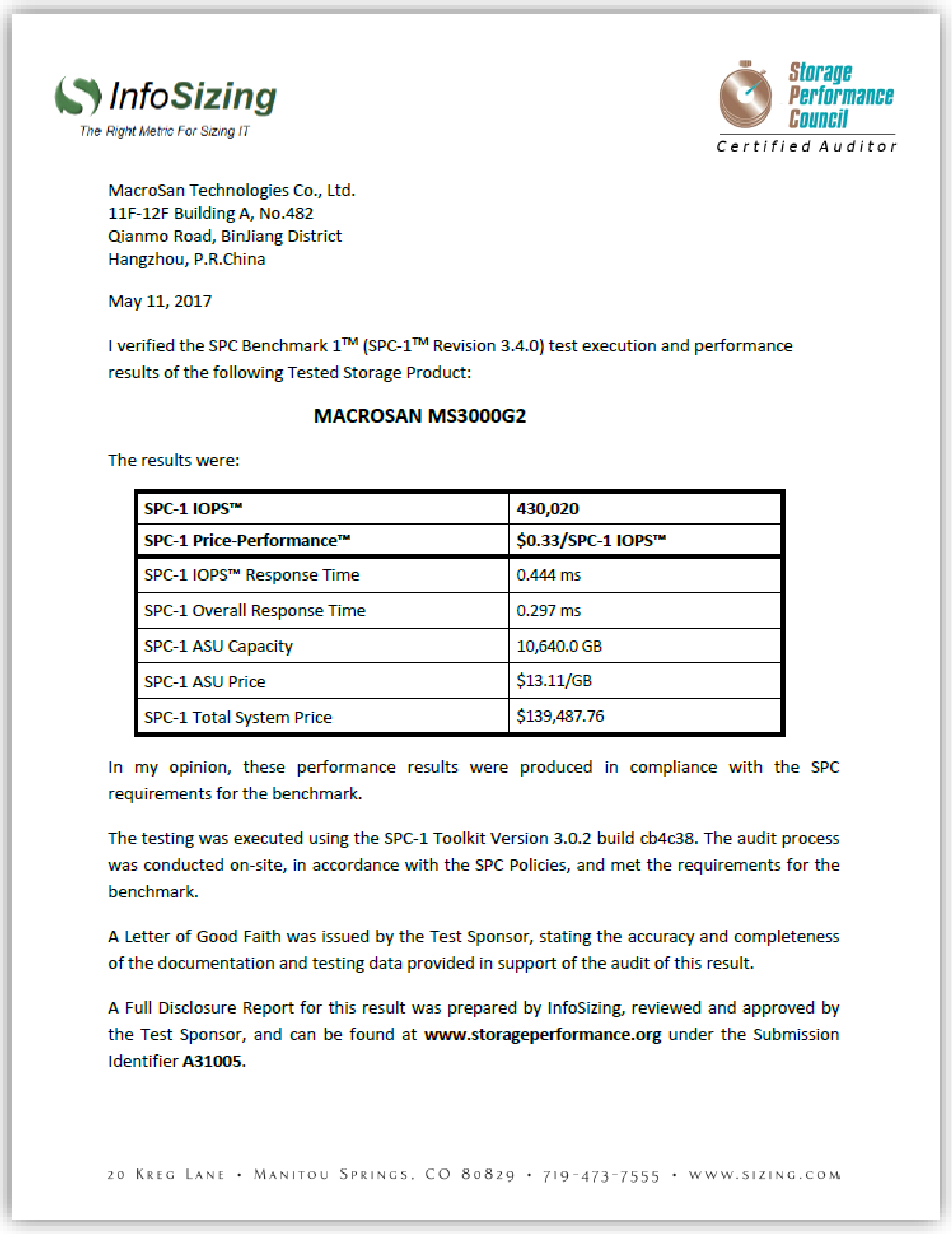
Benchmark Specification and Glossary

The official SPC Benchmark 1™ (SPC-1™) specification is available on the website of the Storage Performance Council (SPC) at www.storageperformance.org.

The SPC-1™ specification contains a glossary of the SPC-1™ terms used in this publication.

Table of Contents

AUDIT CERTIFICATION



InfoSizing
The Right Metric For Sizing IT

Storage Performance Council
Certified Auditor

MacroSan Technologies Co., Ltd.
11F-12F Building A, No.482
Qianmo Road, BinJiang District
Hangzhou, P.R.China

May 11, 2017

I verified the SPC Benchmark 1™ (SPC-1™ Revision 3.4.0) test execution and performance results of the following Tested Storage Product:

MACROSAN MS3000G2

The results were:

SPC-1 IOPS™	**430,020**
SPC-1 Price-Performance™	**$0.33/SPC-1 IOPS™**
SPC-1 IOPS™ Response Time	0.444 ms
SPC-1 Overall Response Time	0.297 ms
SPC-1 ASU Capacity	10,640.0 GB
SPC-1 ASU Price	$13.11/GB
SPC-1 Total System Price	$139,487.76

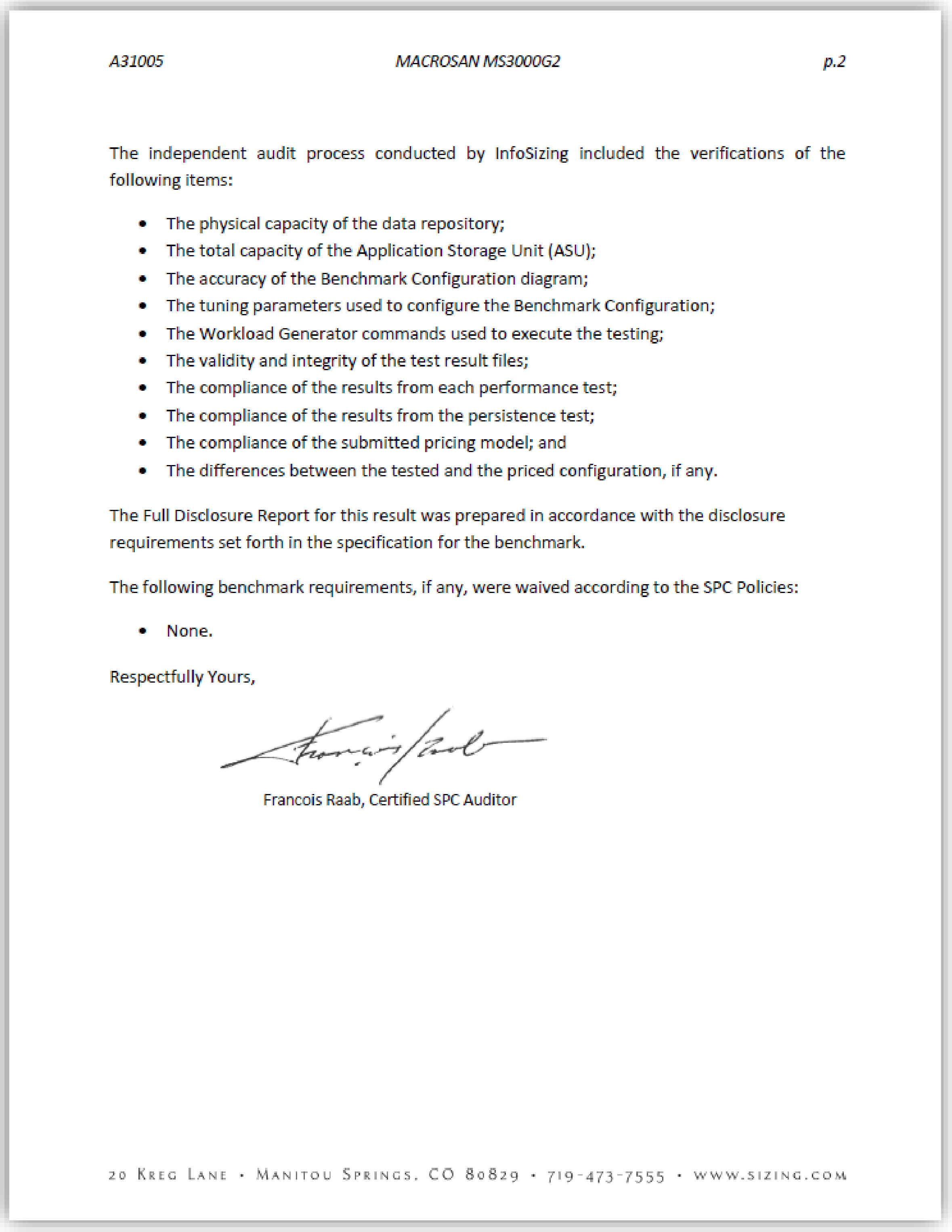
In my opinion, these performance results were produced in compliance with the SPC requirements for the benchmark.

The testing was executed using the SPC-1 Toolkit Version 3.0.2 build cb4c38. The audit process was conducted on-site, in accordance with the SPC Policies, and met the requirements for the benchmark.

A Letter of Good Faith was issued by the Test Sponsor, stating the accuracy and completeness of the documentation and testing data provided in support of the audit of this result.

A Full Disclosure Report for this result was prepared by InfoSizing, reviewed and approved by the Test Sponsor, and can be found at **www.storageperformance.org** under the Submission Identifier **A31005**.

20 Kreg Lane • Manitou Springs, CO 80829 • 719-473-7555 • www.sizing.com



The independent audit process conducted by InfoSizing included the verifications of the following items:

- The physical capacity of the data repository;
- The total capacity of the Application Storage Unit (ASU);
- The accuracy of the Benchmark Configuration diagram;
- The tuning parameters used to configure the Benchmark Configuration;
- The Workload Generator commands used to execute the testing;
- The validity and integrity of the test result files;
- The compliance of the results from each performance test;
- The compliance of the results from the persistence test;
- The compliance of the submitted pricing model; and
- The differences between the tested and the priced configuration, if any.

The Full Disclosure Report for this result was prepared in accordance with the disclosure requirements set forth in the specification for the benchmark.

The following benchmark requirements, if any, were waived according to the SPC Policies:

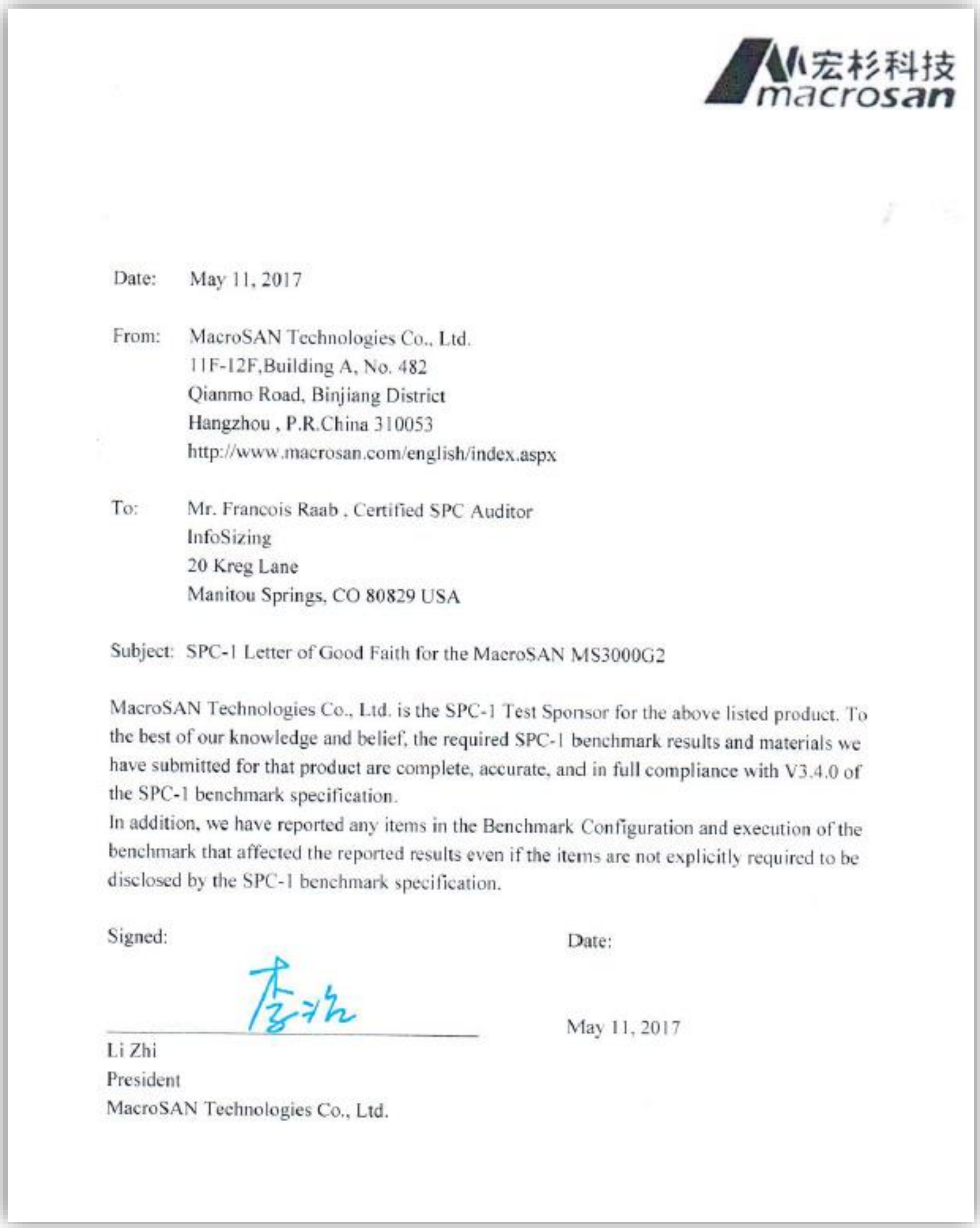
- None.

Respectfully Yours,

[illegible]

Francois Raab, Certified SPC Auditor

LETTER OF GOOD FAITH



宏杉科技 macrosan

Date: May 11, 2017

From: MacroSAN Technologies Co., Ltd.
11F-12F,Building A, No. 482
Qianmo Road, Binjiang District
Hangzhou , P.R.China 310053
http://www.macrosan.com/english/index.aspx

To: Mr. Francois Raab , Certified SPC Auditor
InfoSizing
20 Kreg Lane
Manitou Springs, CO 80829 USA

Subject: SPC-1 Letter of Good Faith for the MacroSAN MS3000G2

MacroSAN Technologies Co., Ltd. is the SPC-1 Test Sponsor for the above listed product. To the best of our knowledge and belief, the required SPC-1 benchmark results and materials we have submitted for that product are complete, accurate, and in full compliance with V3.4.0 of the SPC-1 benchmark specification.
In addition, we have reported any items in the Benchmark Configuration and execution of the benchmark that affected the reported results even if the items are not explicitly required to be disclosed by the SPC-1 benchmark specification.

Signed: [illegible]

Date: May 11, 2017

Li Zhi
President
MacroSAN Technologies Co., Ltd.

SPC Benchmark 1™

Executive Summary

MacroSAN Technologies Co., Ltd
MacroSAN MS3000G2

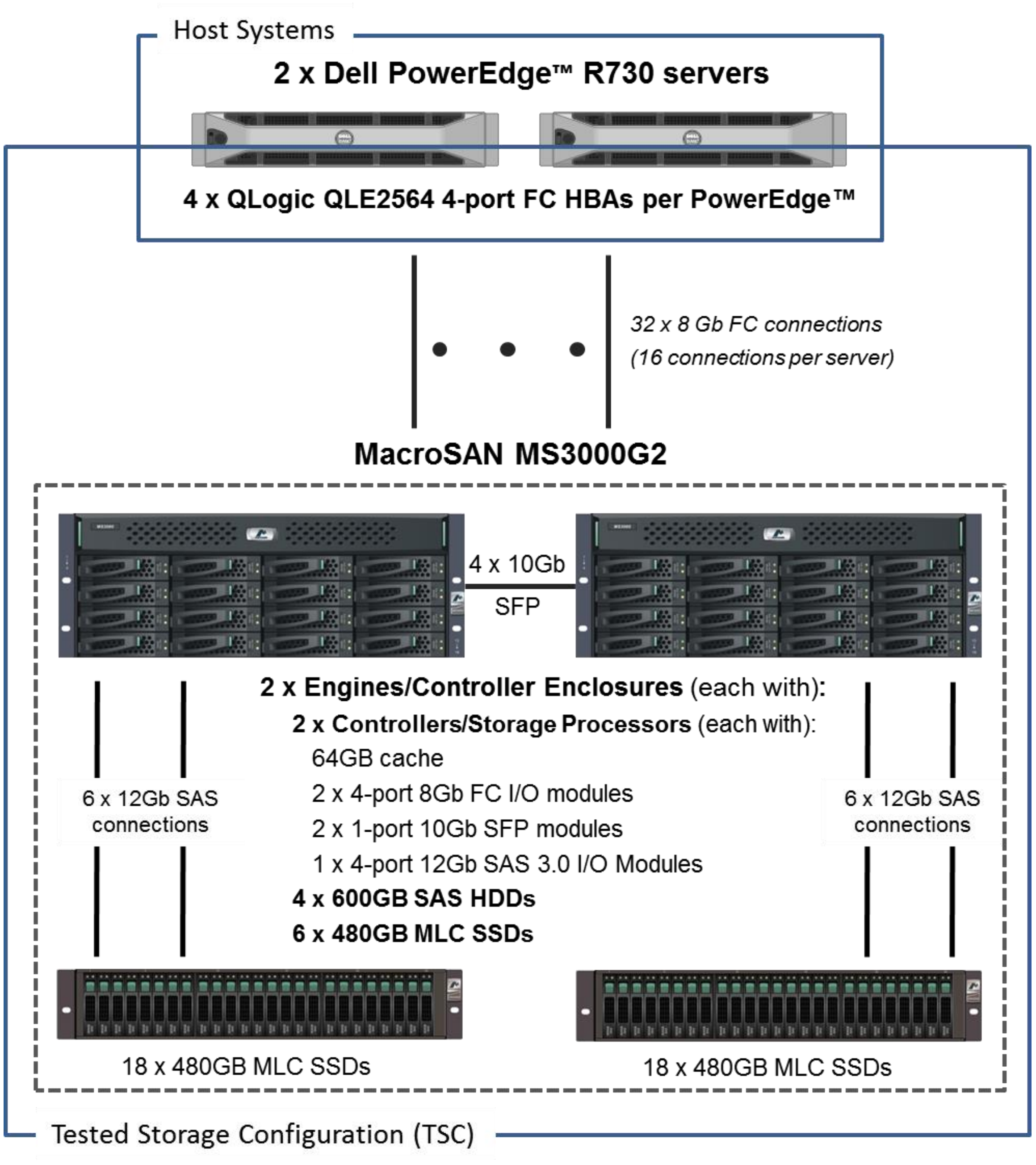
SPC-1 IOPS™	**430,020**
SPC-1 Price-Performance™	**$324.38/SPC-1 KIOPS™**
SPC-1 IOPS™ Response Time	0.444 ms
SPC-1 Overall Response Time	0.297 ms
SPC-1 ASU Capacity	10,640.0 GB
SPC-1 ASU Price	$13.11/GB
SPC-1 Total System Price	$139,487.76
Data Protection Level	Protected 2 (RAID-10 and full redundancy)
Physical Storage Capacity	25,872 GB
Pricing Currency / Target Country	U.S. Dollars / People's Republic of China

SPC-1 V3.4.0

Submission Identifier: A31005

Submitted For Review: May 15, 2017

Benchmark Configuration Diagram


Host Systems
2 x Dell PowerEdge™ R730 servers
4 x QLogic QLE2564 4-port FC HBAs per PowerEdge™
32 x 8 Gb FC connections
(16 connections per server)
MacroSAN MS3000G2
4 x 10Gb
SFP
2 x Engines/Controller Enclosures (each with):
2 x Controllers/Storage Processors (each with):
64GB cache
2 x 4-port 8Gb FC I/O modules
2 x 1-port 10Gb SFP modules
1 x 4-port 12Gb SAS 3.0 I/O Modules
4 x 600GB SAS HDDs
6 x 480GB MLC SSDs
6 x 12Gb SAS connections
6 x 12Gb SAS connections
18 x 480GB MLC SSDs
18 x 480GB MLC SSDs
Tested Storage Configuration (TSC)

Tested Storage Product Description

MacroSAN MS3000G2 is a new generation of MacroSAN mid-range storage products which is oriented to medium size data centers. It integrates lots of advanced design concept and architecture technology to provide safe, reliable, flexible storage platform.

MS3000G2 can compose multi-engine architecture, and scale to at most 16 controllers. Each engine, with two controllers, support at most 256GB cache and default configuration of six 1 Gb/s Ethernet interfaces and four 10 Gb/s Ethernet interfaces. Furthermore, based on ODSP software platform, MS3000G2 can realize interface development, customized function development and function migration. MS3000G2 reaches the leading level in performance, reliability, function and management.

For more details, visit:

http://www.macrosan.com/english/pro.aspx?id=55

Priced Storage Configuration Components

8 x QLogic QLE2564 4-port FC HBAs

2 x MacroSAN MS3000G2, each with:
- **2 x Controllers/Storage Processors, each with:**
  - **64GB cache**
  - **2 x 4-port 8Gb FC I/O modules**
  - **2 x 1-port 10Gb SFP modules**
  - **1 x 4-port 12Gb SAS 3.0 I/O Modules**
- **4 x 600GB SAS HDDs**
- **6 x 480GB MLC SSDs**

2 x External storage enclosures, each with:

18 x 480GB MLC SSDs

Storage Configuration Pricing

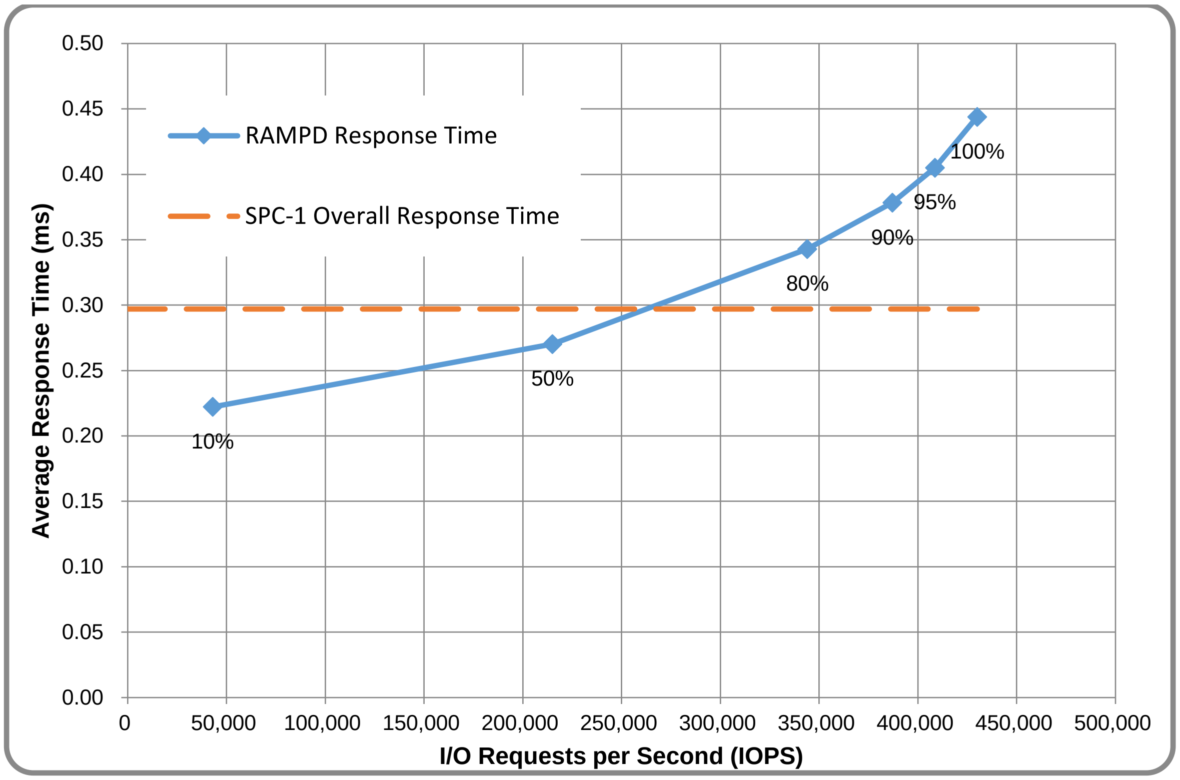
	Description	Qty	Unit Price	Ext. Price	Disc.	Disc. Price
	Hardware & Software					
MS3000G2-128GB-Enhanced	MS3000G2 Enhanced Storage Controllers Unit (Dual Controllers, 4U, 16*3.5/2.5 disks, 128GB cache, 6*GE ports+4*10GE port-no SFPs, 4*IO module slots, 3Y 7x9xND Basic Svc&Warranty）	2	35,007.35	70,014.71	50%	35,007.35
SSU2225	SSU2225 SAS Sw itch Unit(support 25 2.5inch disks, SAS 3.0)	2	7,948.88	15,897.75	50%	7,948.88
MS3M2HD600G0C2AEA	600GB 15KRPM 2.5inch SAS disk/3.5 inch carrier	8	1,606.62	12,852.94	50%	6,426.47
MS2M1SD480G0M2AEB	480GB MLC SSD Drive(2.5") for SSU2225	36	2,367.65	85,235.29	50%	42,617.65
MS3M1SD480G0M2AEA	480GB MLC SSD Drive(2.5") for MS3000G2	12	2,367.65	28,411.76	50%	14,205.88
IOA1014A	4*8Gb FC IO Module	8	2,029.41	16,235.29	50%	8,117.65
MS1MMF010G	10Gb MultiMode Datacom SFP+ Optical Transceiver	8	321.32	2,570.59	50%	1,285.29
LIS_MS3000-M_BASE_STD	MacroSAN ODSP for Multi control - MS3000G2 Storage Management Softw are (Basic Storage Management, CRAID, sopport FC&iSCSI, System monitoring and Warning） - English version	1	7,058.82	7,058.82	50%	3,529.41
LC05-LC05-MI2-L5	5M fiber（multimode, LC-LC）	36	8.00	288.00		288.00
05020012	SAS cable (1M, miniSASHD-miniSASHD)	12	0.00	0.00		0.00
02090044	Qlogic QLE2564 HBA Card, PCIE, 8Gbps 4-Ports, Fiber Channel Multimode LC Optic Interface	8	1,500.00	12,000.00		12,000.00
				Hardware & Software Subtotal		**131,426.58**
	Support & Maintenance					
MS-3000-SV-BS-SPU	MS3000G2 Installation Service - Per SPU	2	1,088.24	2,176.47	20%	1,741.18
MS-3000-SV-BS-DSU	M3000G2 Installation Service - Per DSU	2	588.24	1,176.47	20%	941.18
MS-3000-SV-GS-1Y-SPU	MS3000G2 Svc&Warranty Upgrade - 7x24x4 Premium - Per Year Per SPU	2	1,941.18	3,882.35	20%	3,105.88
MS-3000-SV-GS-1Y-DSU	MS3000G2 Svc&Warranty Upgrade - 7x24x4 Premium - Per Year Per DSU	2	1,420.59	2,841.18	20%	2,272.94
				Support & Maintenance Subtotal		**8,061.18**
	SPC-1 Total System Price					**139,487.76**
	SPC-1 IOPS™					430,020
	SPC-1 Price-Performance™ ($/SPC-1 KIOPS™)					**324.38**
	SPC-1 ASU Capacity (GB)					10,640
	SPC-1 ASU Price ($/GB)					**13.11**

Discount Details: The discounts shown are generally available, and based on the capacity and total price of the storage configuration purchased.

Warranty: Pricing includes Gold-Level Service with: 24x7 online support, unlimited software upgrades and bug fixes, and on-site presence of a qualified maintenance engineer within 4 hours of a problem acknowledgement, inside the Target Market.

Availability Date: Currently available.

Response Time and Throughput Graph



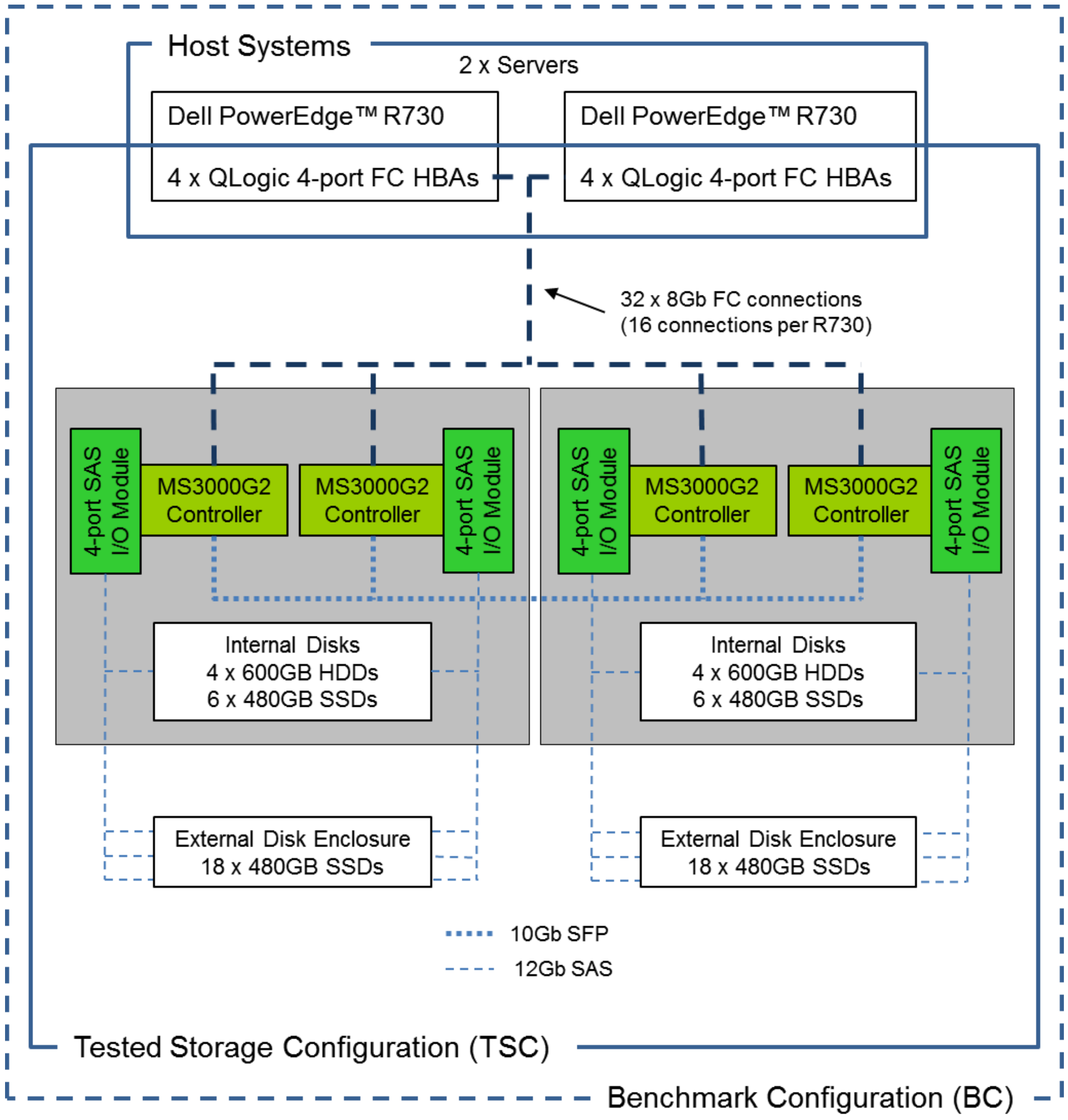
Contact Information	
Test Sponsor Primary Contact	MacroSAN Technologies Co ., Ltd.– http://www.macrosan.com.cn Yi Shen – shenyi@macrosan.com
SPC Auditor	InfoSizing – www.sizing.com Francois Raab – francois@sizing.com

Revision Information	
SPC Benchmark 1™ Revision	V3.4.0
SPC-1 Workload Generator Revision	V3.0.2 build cb4c38
Publication Revision History	• First Edition: May 15, 2017 • Second Edition: February 15, 2018 - Updated SPC-1 Price-Performance™ metric based on SPC-1 v3.6.0 definition.

CONFIGURATION INFORMATION

Benchmark Configuration and Tested Storage Configuration

The following diagram illustrates the Benchmark Configuration (BC), including the Tested Storage Configuration (TSC) and the Host System(s).


Host Systems
2 x Servers
Dell PowerEdge™ R730
4 x QLogic 4-port FC HBAs
Dell PowerEdge™ R730
4 x QLogic 4-port FC HBAs
32 x 8Gb FC connections (16 connections per R730)
4-port SAS I/O Module
MS3000G2 Controller
MS3000G2 Controller
4-port SAS I/O Module
4-port SAS I/O Module
MS3000G2 Controller
MS3000G2 Controller
4-port SAS I/O Module
Internal Disks 4 x 600GB HDDs 6 x 480GB SSDs
Internal Disks 4 x 600GB HDDs 6 x 480GB SSDs
External Disk Enclosure 18 x 480GB SSDs
External Disk Enclosure 18 x 480GB SSDs
10Gb SFP
12Gb SAS
Tested Storage Configuration (TSC)
Benchmark Configuration (BC)

Storage Network Configuration

The Tested Storage Configuration (TSC) involved an external storage subsystem made of 2 MacroSAN MS3000G2 Engines, driven by 2 host systems (Dell PowerEdge R730). Each R730 connected one-to-one to each MS3000G2. Each one-to-one connection was established using two ports from of each of the four 4-port HBAs on the R730; and two ports from each of the two 4-port I/O Modules on each of the two Storage Processors in each MS3000G2. These paths used Fibre Chanel operating at 8Gbps.

Host System and Tested Storage Configuration Components

The following table lists the components of the Host System(s) and the Tested Storage Configuration (TSC).

Host Systems

2 x Dell PowerEdge R730 servers, each with:
    2 x Intel Xeon E5-2620 V4 (2.1 GHz, 8-Core, 20MB L3)
    256GB Main Memory
    Red Hat Enterprise Linux 7.1

Priced Storage Configuration
8 x QLogic QLE2564 4-port FC HBAs

2 x MacroSAN MS3000G2, each with:
    2 x Controllers/Storage Processors, each with:
        64GB cache
        2 x 4-port 8Gb FC I/O modules
        2 x 1-port 10Gb SFP modules
        1 x 4-port 12Gb SAS 3.0 I/O Modules
    4 x 600GB SAS HDDs
    6 x 480GB MLC SSDs
2 x External storage enclosures, each with:
    18 x 480GB MLC SSDs

Differences Between Tested and Priced Storage Configurations

There were no differences between the Tested Storage Configuration and the Priced Storage Configuration.

Component Changes in Revised Full Disclosure Report

The following table outlines component changes that were made in revisions to this Full Disclosure Report.

Original Component	Revised Component	Description of Change
n/a	n/a	Initial submission

Benchmark Configuration Creation Process

Customer Tuning Parameters and Options

All the customer tuning parameters and options that have been altered from their default values for this benchmark are included in Appendix C and in the Supporting Files (see Appendix A).

Tested Storage Configuration Creation

A detailed description of how the logical representation of the TSC was created is included in Appendix D and in the Supporting Files (see Appendix A).

Tested Storage Configuration Inventory

An inventory of the components in the TSC, as seen by the Benchmark Configuration, is included in Appendix E and in the Supporting Files (see Appendix A).

Workload Generator Storage Configuration

The SPC-1 Workload Generator storage configuration commands and parameters used to invoke the execution of the tests are included in Appendix F and in the Supporting Files (see Appendix A).

Logical Volume Capacity and ASU Mapping

The following table details the capacity of each ASU and how they are mapped to logical volumes (LV).

	LV per ASU	LV Capacity	Used per LV	Total per ASU	% ASU Capacity
ASU-1	18	266.0	266.0	4,788.0	45%
ASU-2	18	266.0	266.0	4,788.0	45%
ASU-3	2	532.0	532.0	1,064.0	10%
			SPC-1 ASU Capacity	**10,640.0**	

Physical Storage Capacity and Utilization

The following table details the Physical Capacity of the storage devices and the Physical Capacity Utilization (percentage of Total Physical Capacity used) in support of hosting the ASUs.

Devices	Count	Physical Capacity	Total Capacity
480GB SSD	48	446.0	21,408.0
600GB HDD	8	558.0	4,464.0
		Total Physical Capacity	**25,872.0**
		Physical Capacity Utilization	**41.13%**

Data Protection

The data protection level used for all logical volumes was **Protected 2**, which was accomplished by configuring 2 pools, each 24 drives, into 8 RAID-10 arrays. All components and access paths from the Host Systems to the Storage Devices were redundant.

BENCHMARK EXECUTION RESULTS

This portion of the Full Disclosure Report documents the results of the various SPC-1 Tests, Test Phases, and Test Runs.

Benchmark Execution Overview

Workload Generator Input Parameters

The SPC-1 Workload Generator commands and input parameters for the Test Phases are presented in the Supporting Files (see Appendix A).

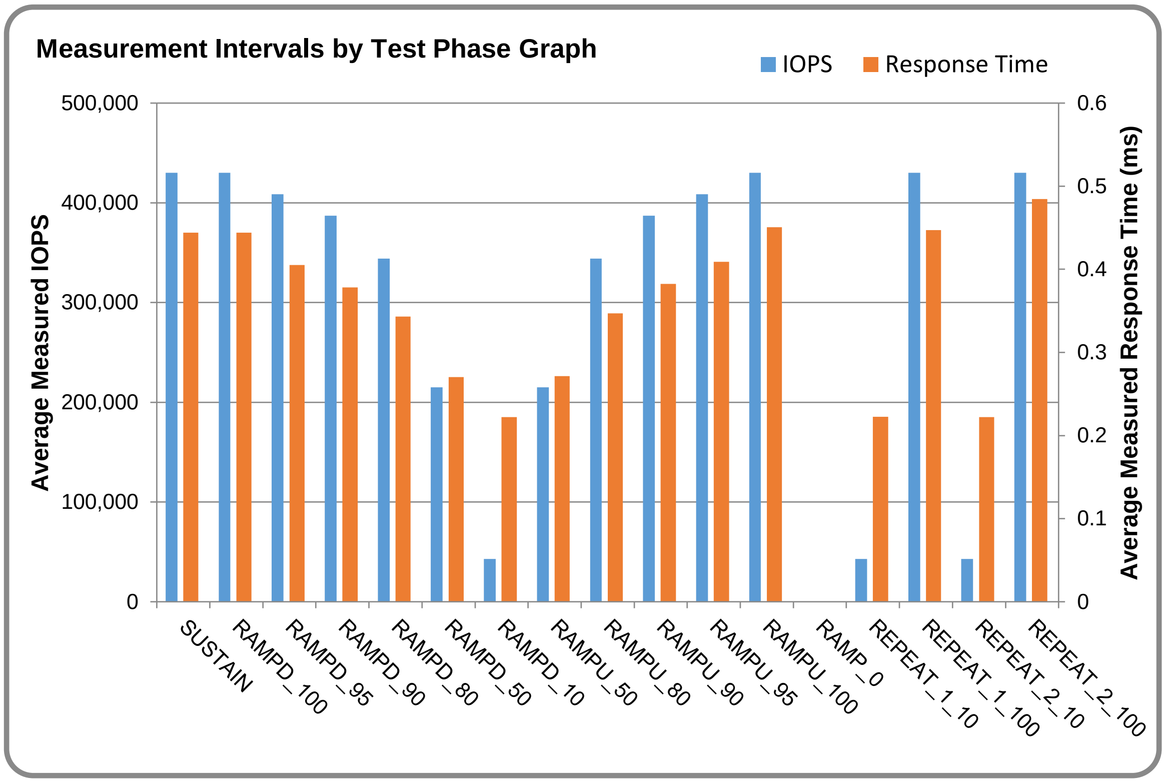
Primary Metrics Test Phases

The benchmark execution consists of the Primary Metrics Test Phases, including the Test Phases SUSTAIN, RAMPD_100 to RAMPD_10, RAMPU_50 to RAMPU_100, RAMP_0, REPEAT_1 and REPEAT_2.

Each Test Phase starts with a transition period followed by a Measurement Interval.

Measurement Intervals by Test Phase Graph

The following graph presents the average IOPS and the average Response Times measured over the Measurement Interval (MI) of each Test Phase.



Exception and Waiver

The audit of this SPC-1 result was conducted on-site. During the course of the audit, no exceptions were encountered and no benchmark requirements were waived.

SUSTAIN Test Phase

SUSTAIN – Results File

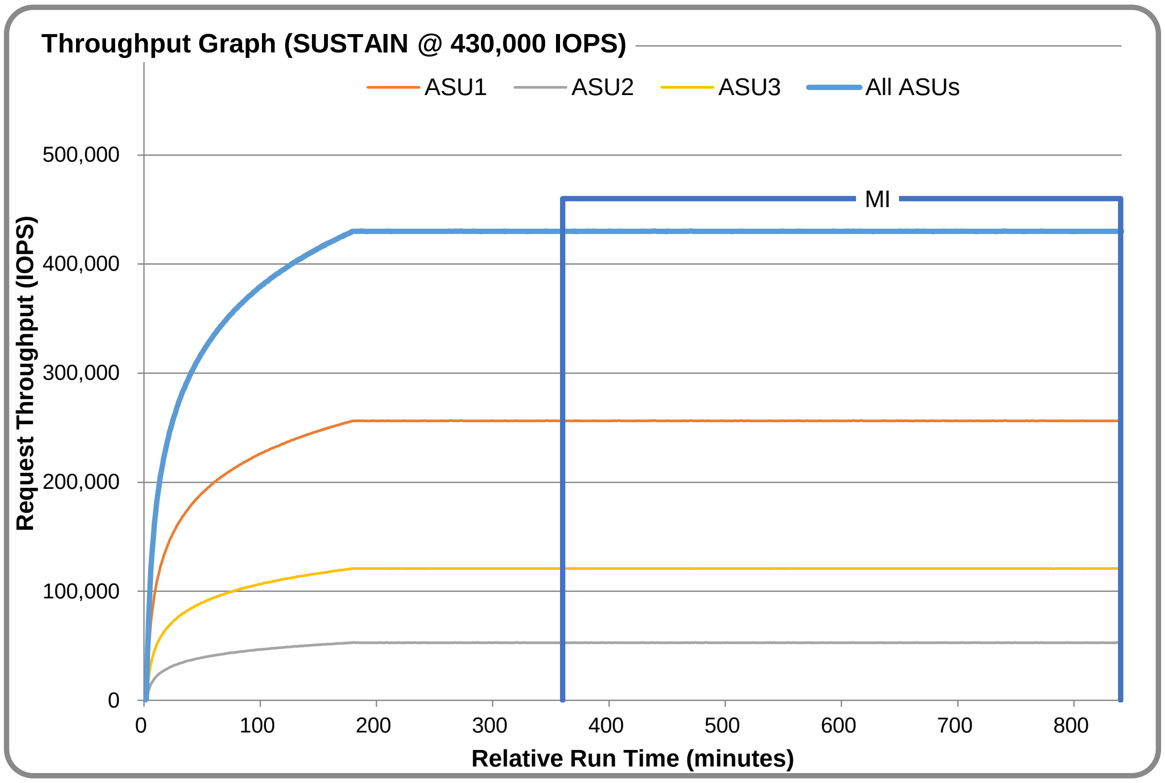
The results file generated during the execution of the SUSTAIN Test Phase is included in the Supporting Files (see Appendix A) as follows:

- **SPC1_METRICS_0_Raw_Results.xlsx**

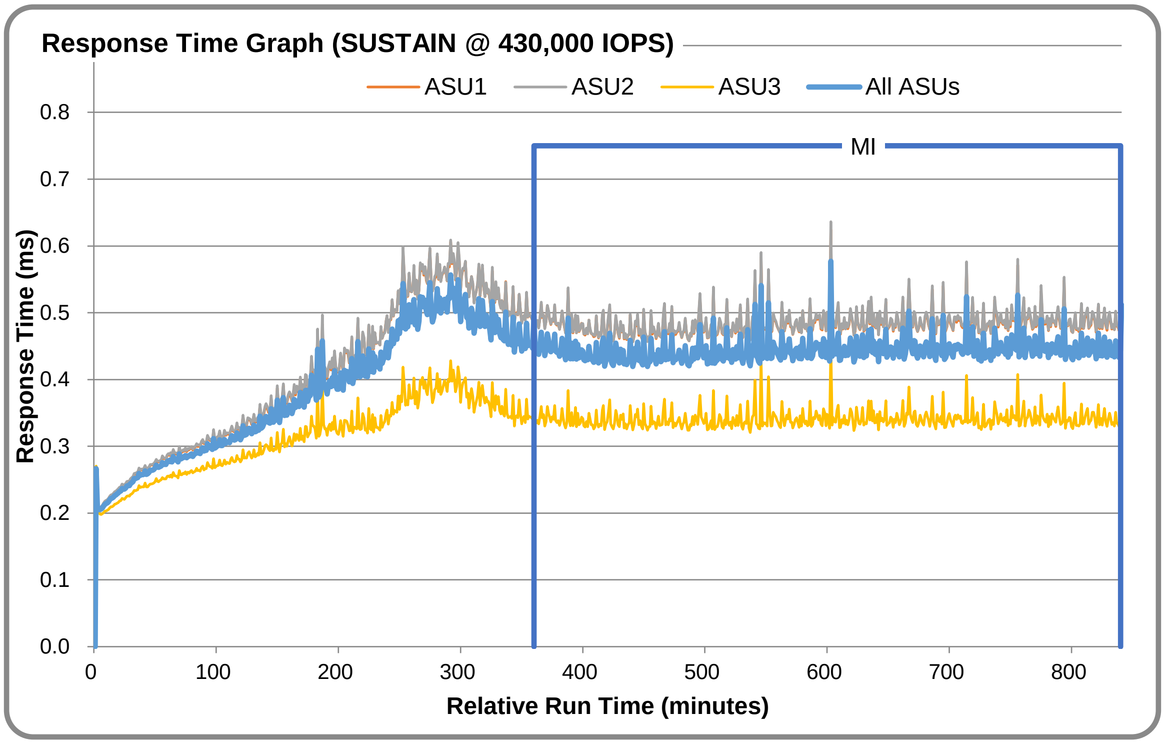
SUSTAIN – Execution Times

Interval	Start Date & Time	End Date &Time	Duration
Transition Period	3-May-17 12:45:21	3-May-17 18:45:17	5:59:56
Measurement Interval	3-May-17 18:45:17	4-May-17 02:45:18	8:00:01

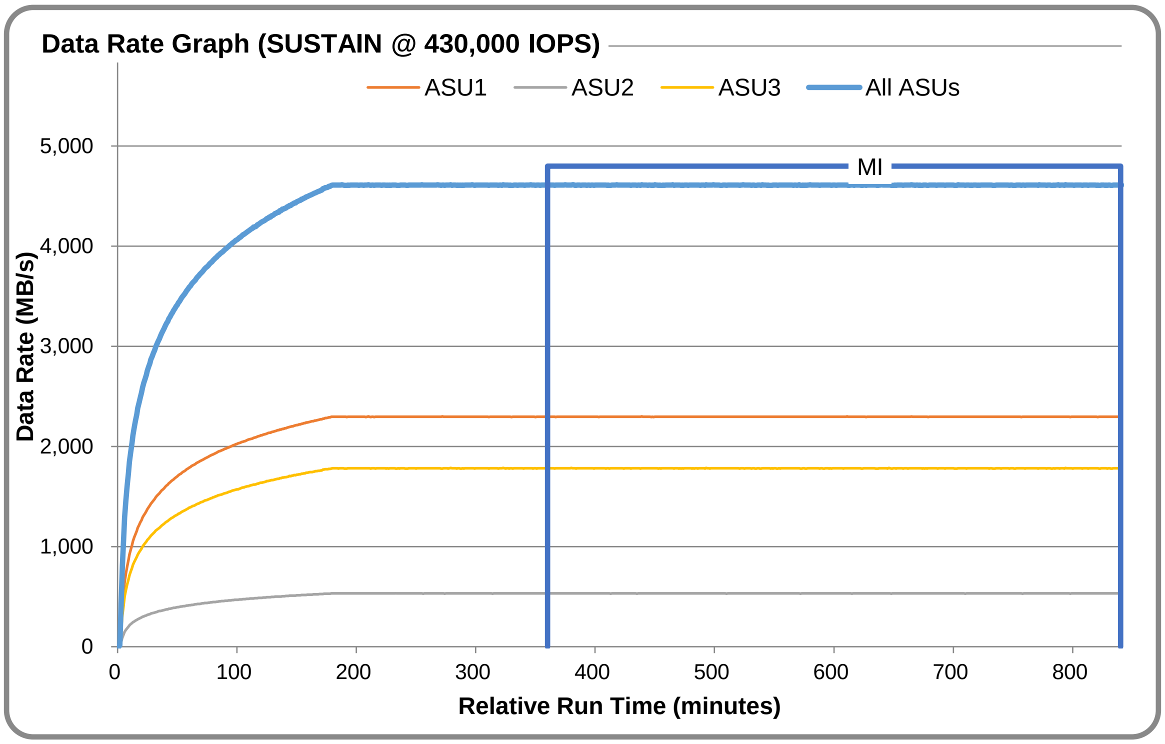
SUSTAIN – Throughput Graph



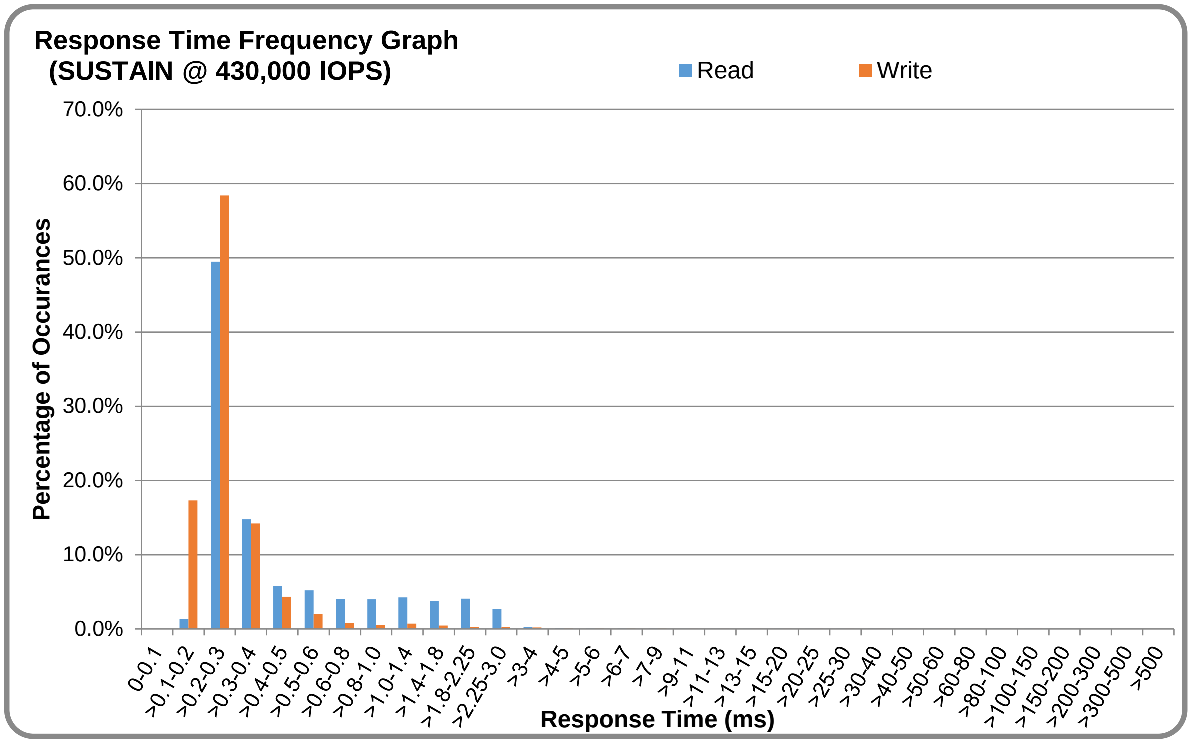
SUSTAIN – Response Time Graph



SUSTAIN – Data Rate Graph



SUSTAIN – Response Time Frequency Graph



SUSTAIN – Intensity Multiplier

The following table lists the targeted intensity multiplier (Defined), the measured intensity multiplier (Measured) for each I/O STREAM, its coefficient of variation (Variation) and the percentage of difference (Difference) between Target and Measured.

	ASU1-1	ASU1-2	ASU1-3	ASU1-4	ASU2-1	ASU2-2	ASU2-3	ASU3-1
Defined	0.0350	0.2810	0.0700	0.2100	0.0180	0.0700	0.0350	0.2810
Measured	0.0350	0.2810	0.0700	0.2100	0.0180	0.0700	0.0350	0.2810
Variation	0.0010	0.0003	0.0007	0.0004	0.0015	0.0007	0.0010	0.0003
Difference	0.010%	0.001%	0.002%	0.001%	0.010%	0.000%	0.003%	0.002%

RAMPD_100 Test Phase

RAMPD_100 – Results File

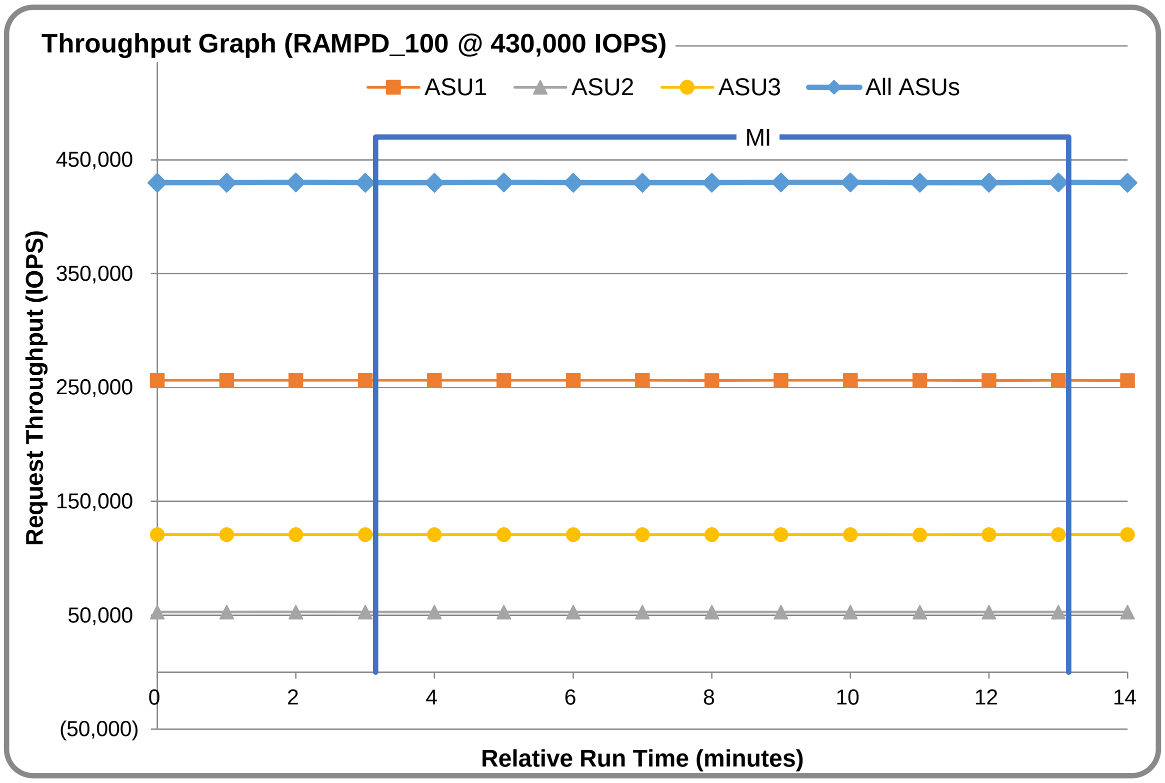
The results file generated during the execution of the RAMPD_100 Test Phase is included in the Supporting Files (see Appendix A) as follows:

- **SPC1_METRICS_0_Raw_Results.xlsx**

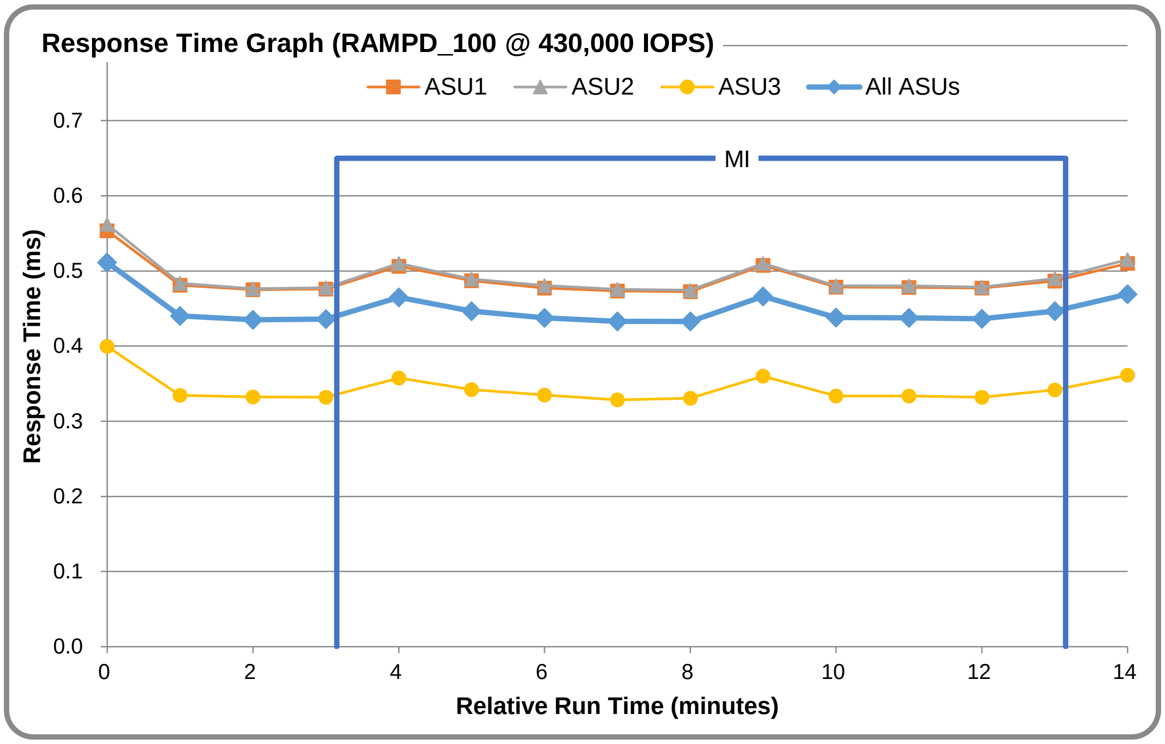
RAMPD_100 – Execution Times

Interval	Start Date & Time	End Date &Time	Duration
Transition Period	4-May-17 02:46:18	4-May-17 02:49:18	0:03:00
Measurement Interval	4-May-17 02:49:18	4-May-17 02:59:19	0:10:01

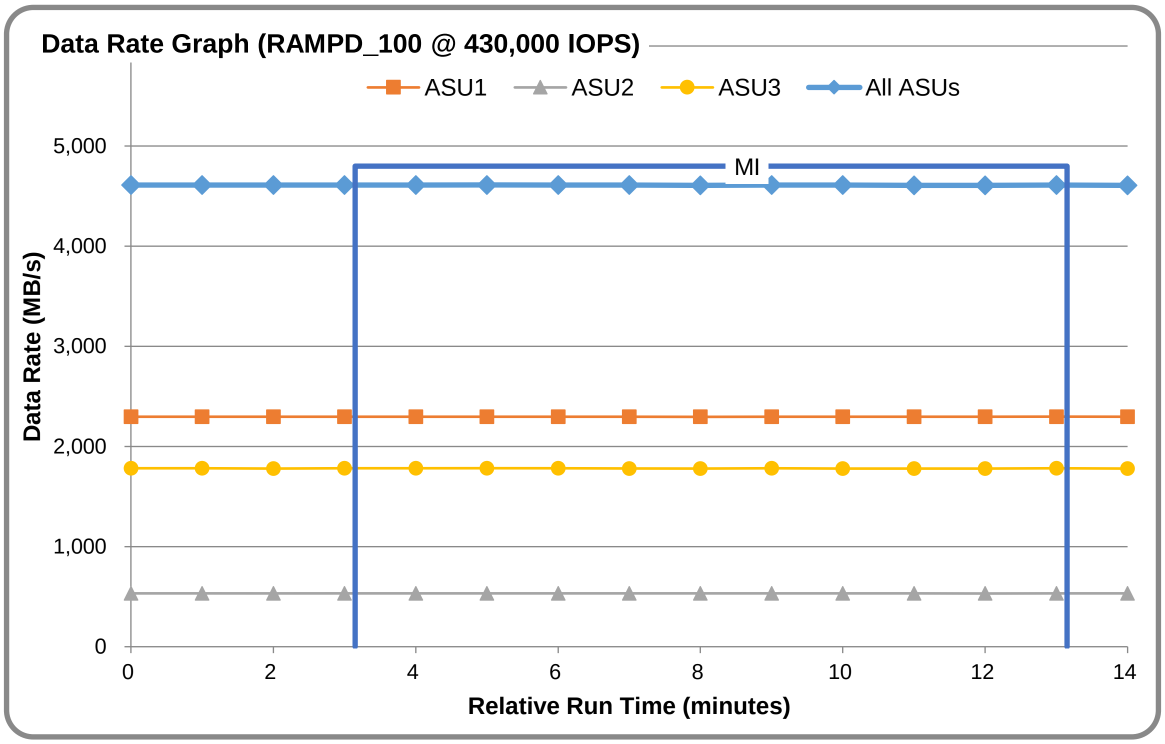
RAMPD_100 – Throughput Graph



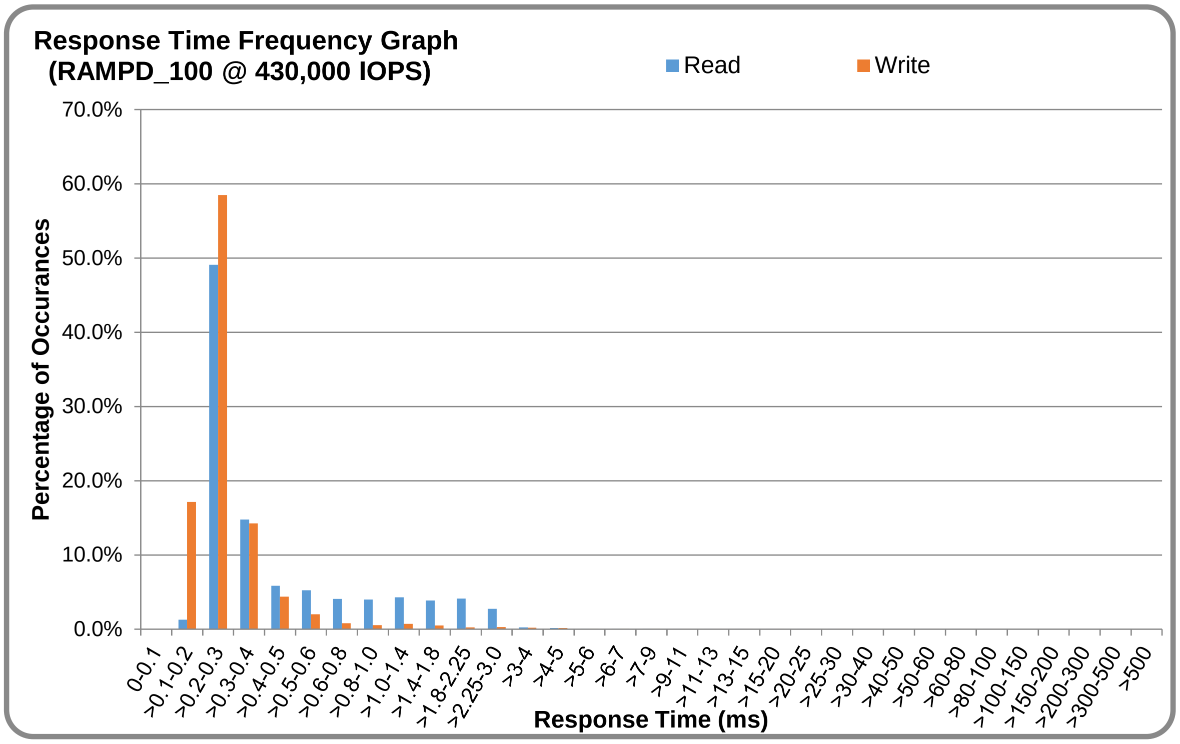
RAMPD_100 – Response Time Graph



RAMPD_100 – Data Rate Graph



RAMPD_100 – Response Time Frequency Graph



RAMPD_100 – Intensity Multiplier

The following table lists the targeted intensity multiplier (Defined), the measured intensity multiplier (Measured) for each I/O STREAM, its coefficient of variation (Variation) and the percentage of difference (Difference) between Target and Measured.

	ASU1-1	ASU1-2	ASU1-3	ASU1-4	ASU2-1	ASU2-2	ASU2-3	ASU3-1
Defined	0.0350	0.2810	0.0700	0.2100	0.0180	0.0700	0.0350	0.2810
Measured	0.0350	0.2810	0.0700	0.2100	0.0180	0.0700	0.0350	0.2809
Variation	0.0008	0.0003	0.0006	0.0003	0.0012	0.0008	0.0005	0.0003
Difference	0.012%	0.017%	0.004%	0.009%	0.089%	0.005%	0.017%	0.029%

RAMPD_100 – I/O Request Summary

I/O Requests Completed in the Measurement Interval	258,007,251
I/O Requests Completed with Response Time <= 30 ms	257,995,428
I/O Requests Completed with Response Time > 30 ms	11,823

Response Time Ramp Test

Response Time Ramp Test – Results File

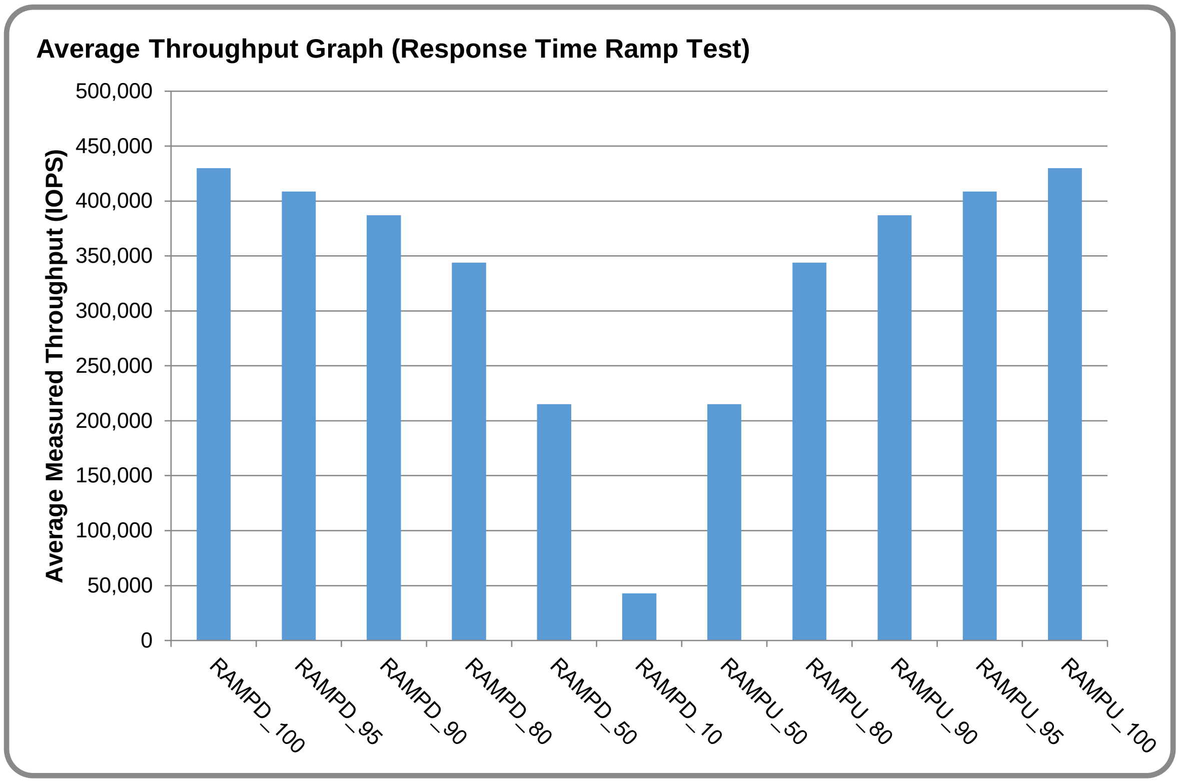
The results file generated during the execution of the Response Time Ramp Test is included in the Supporting Files (see Appendix A) as follows:

- **SPC1_METRICS_0_Raw_Results.xlsx**

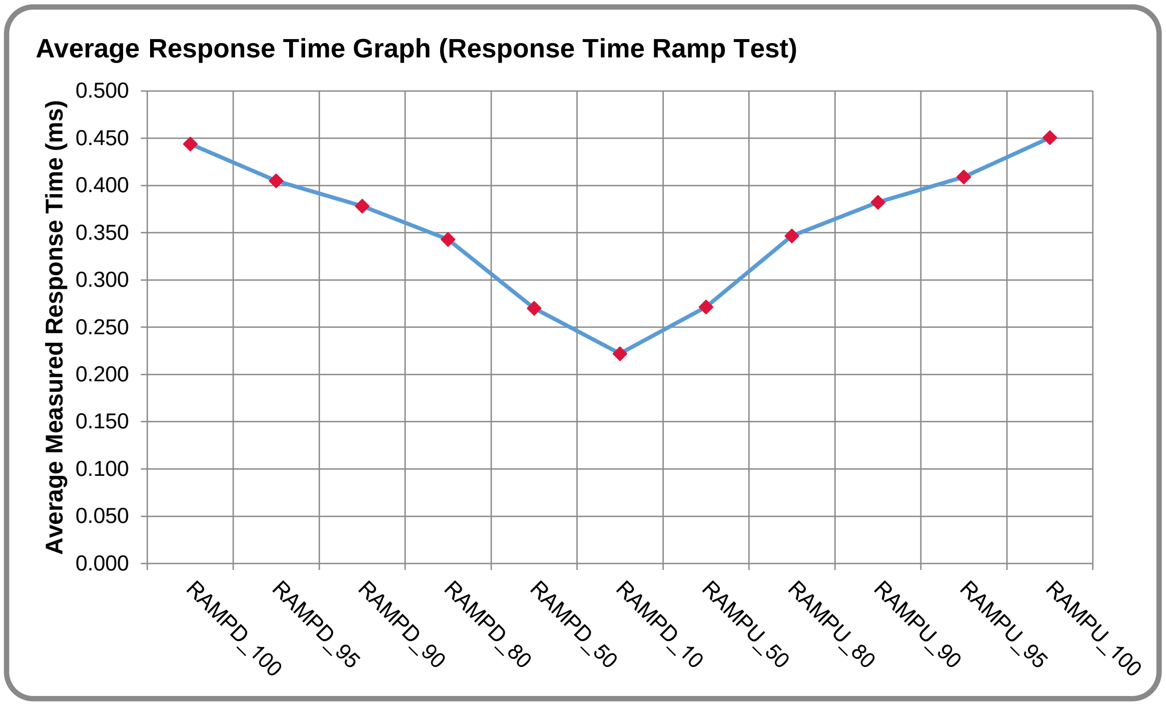
Response Time Ramp Test – Phases

The Response Time Ramp Test is comprised of 11 Test Phases, including six Ramp-Down Phases (executed at 100%, 95%, 90%, 80%, 50%, and 10% of the Business Scaling Unit) and five Ramp-Up Phases (executed at 50%, 80%, 90%, 95%, and 100% of the Business Scaling Unit).

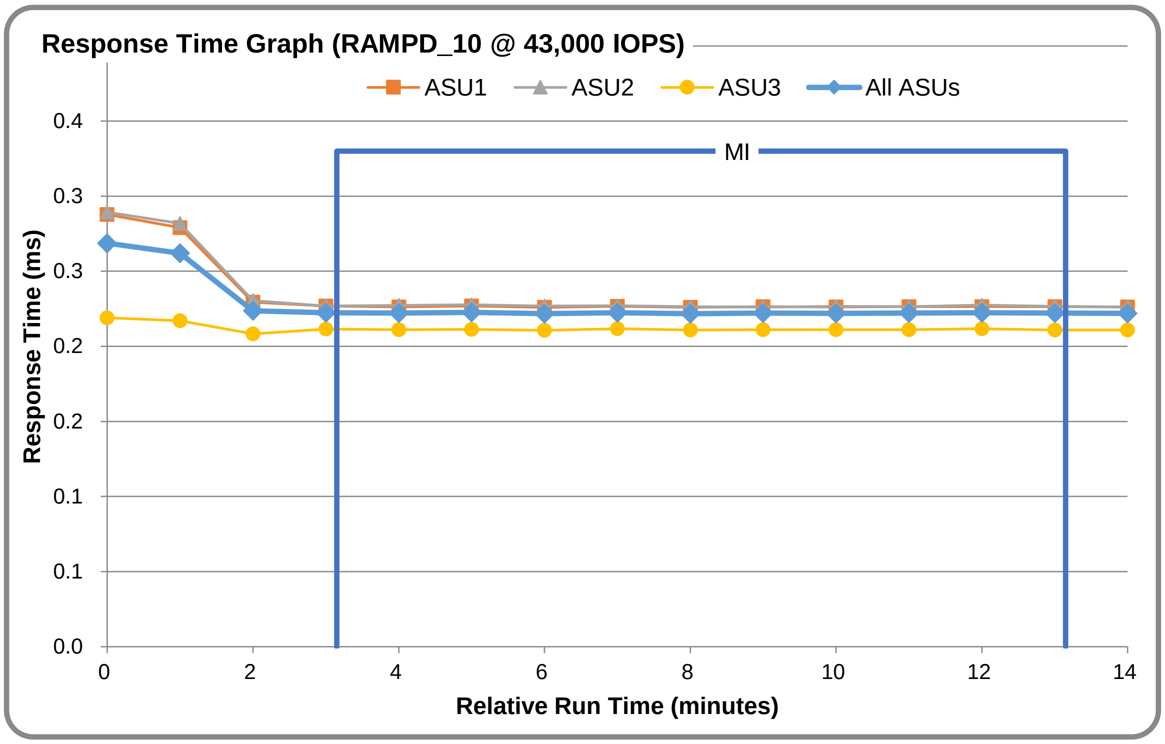
Response Time Ramp Test – Average Throughput Graph



Response Time Ramp Test – Average Response Time Graph



Response Time Ramp Test – RAMPD_10 Response Time Graph



Repeatability Test

Repeatability Test Results File

The results file generated during the execution of the Repeatability Test is included in the Supporting Files (see Appendix A) as follows:

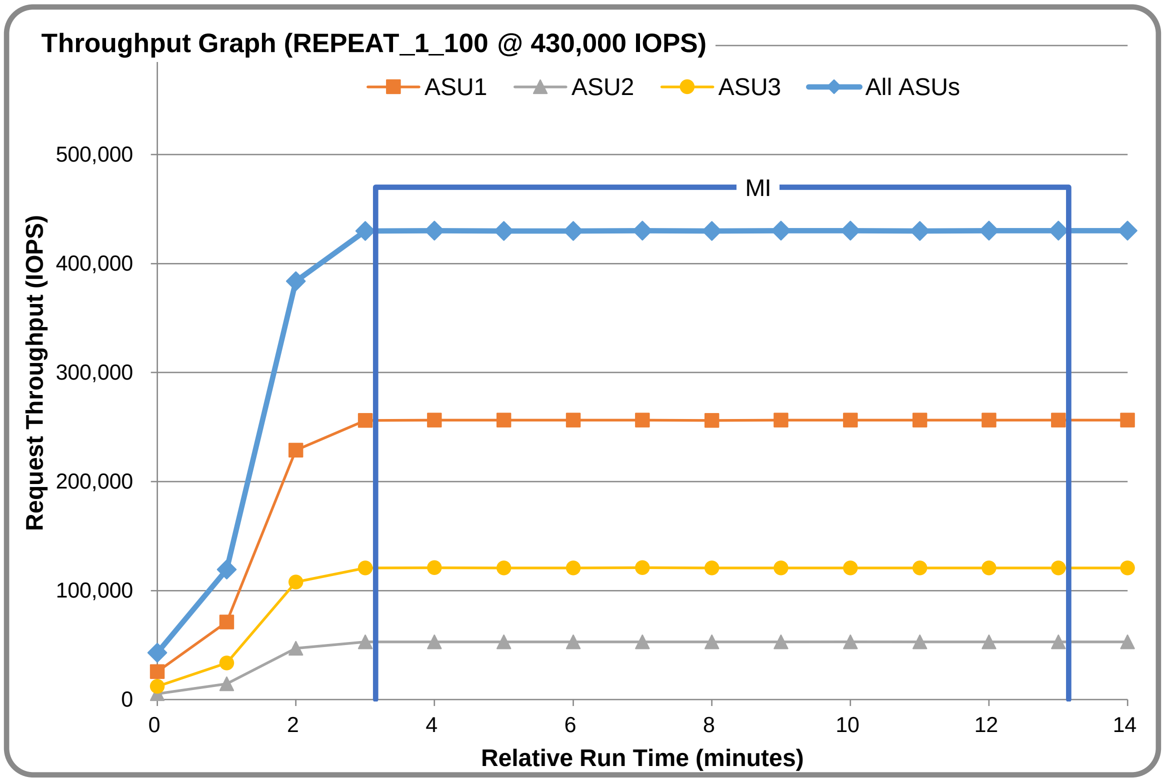
- **SPC1_METRICS_0_Raw_Results.xlsx**

Repeatability Test Results

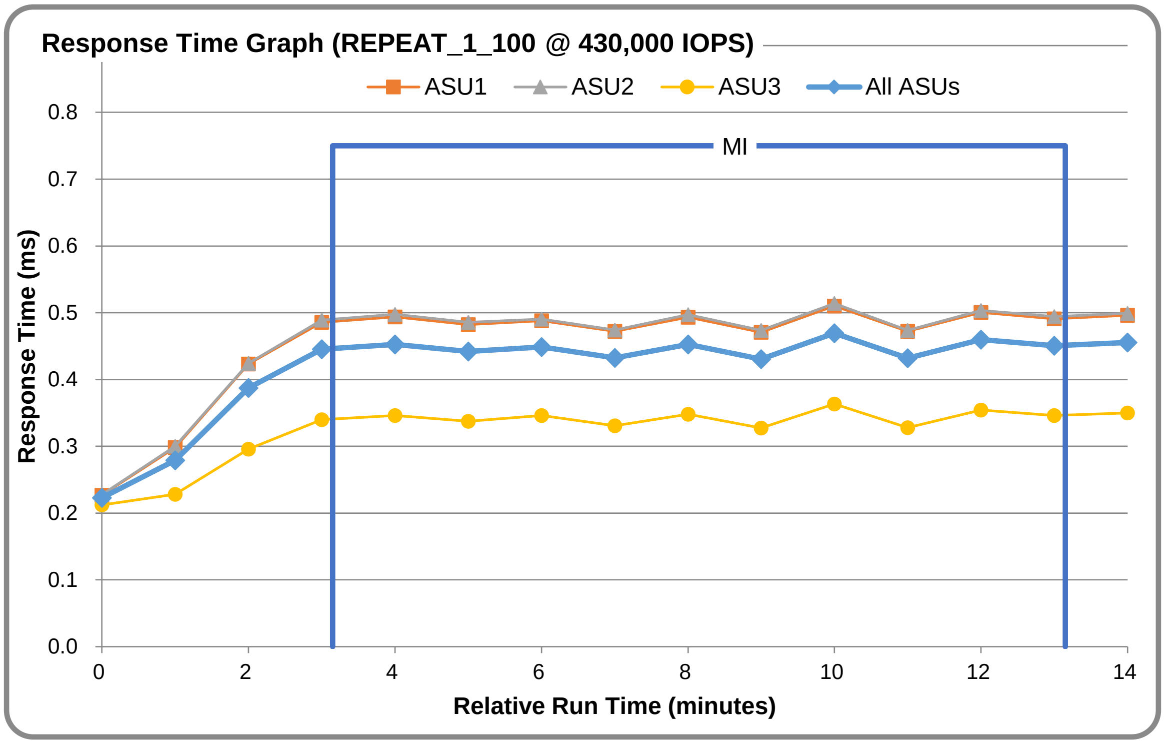
The throughput measurements for the Response Time Ramp Test (RAMPD) and the Repeatability Test Phases (REPEAT_1 and REPEAT_2) are listed in the tables below.

Test Phase	100% IOPS	10% IOPS
RAMPD	430,020	43,008
REPEAT_1	430,005	43,004
REPEAT_2	430,025	43,002

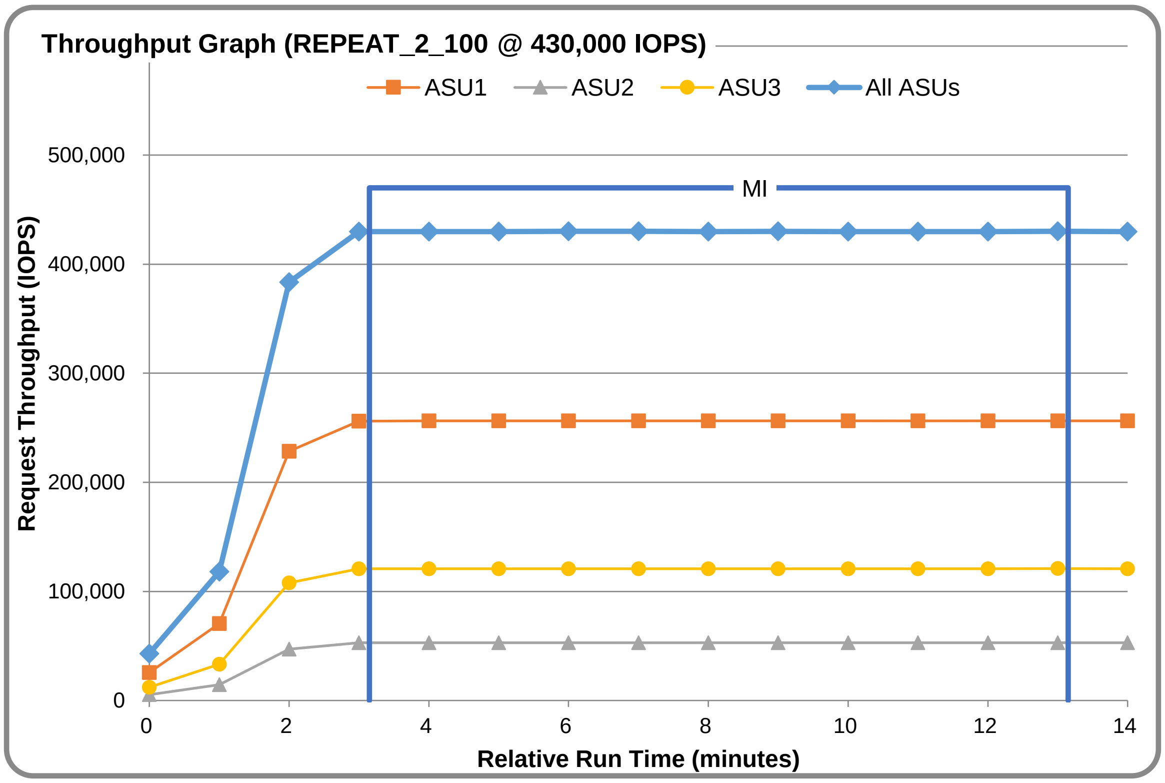
REPEAT_1_100 – Throughput Graph



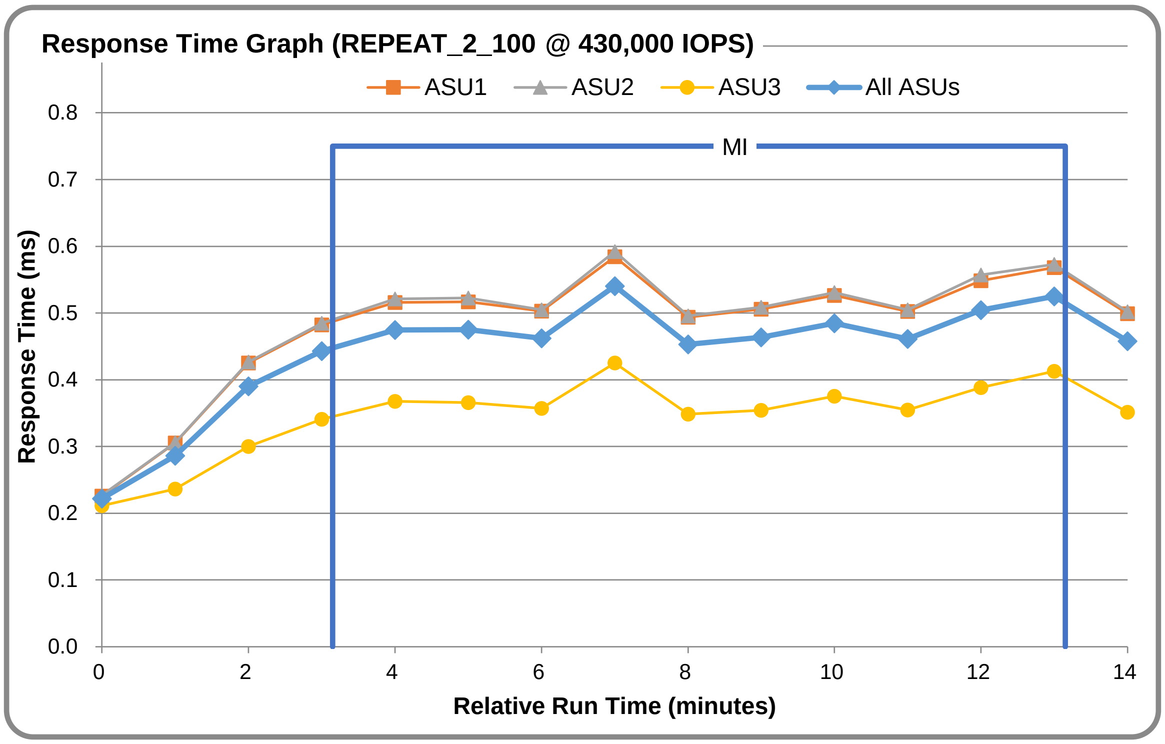
REPEAT_1_100 – Response Time Graph



REPEAT_2_100 – Throughput Graph



REPEAT_2_100 – Response Time Graph



Repeatability Test – Intensity Multiplier

The following tables lists the targeted intensity multiplier (Defined), the measured intensity multiplier (Measured) for each I/O STREAM, its coefficient of variation (Variation) and the percent of difference (Difference) between Target and Measured.

REPEAT_1_100 Test Phase

	ASU1-1	ASU1-2	ASU1-3	ASU1-4	ASU2-1	ASU2-2	ASU2-3	ASU3-1
Defined	0.0350	0.2810	0.0700	0.2100	0.0180	0.0700	0.0350	0.2810
Measured	0.0350	0.2810	0.0700	0.2100	0.0180	0.0700	0.0350	0.2810
Variation	0.0012	0.0004	0.0006	0.0003	0.0012	0.0009	0.0007	0.0004
Difference	0.003%	0.010%	0.010%	0.003%	0.007%	0.007%	0.047%	0.007%

REPEAT_2_100 Test Phase

	ASU1-1	ASU1-2	ASU1-3	ASU1-4	ASU2-1	ASU2-2	ASU2-3	ASU3-1
Defined	0.0350	0.2810	0.0700	0.2100	0.0180	0.0700	0.0350	0.2810
Measured	0.0350	0.2810	0.0700	0.2100	0.0180	0.0700	0.0350	0.2810
Variation	0.0009	0.0003	0.0003	0.0004	0.0022	0.0006	0.0009	0.0003
Difference	0.027%	0.004%	0.002%	0.011%	0.139%	0.017%	0.014%	0.003%

Data Persistence Test

Data Persistence Test Result files

The results files generated during the execution of the Data Persistence Test is included in the Supporting Files (see Appendix A) as follows:

- **SPC1_PERSIST_1_0_Raw_Results.xlsx**
- **SPC1_PERSIST_2_0_Raw_Results.xlsx**

Data Persistence Test Execution

The Data Persistence Test was executed using the following sequence of steps:

- The PERSIST_1_0 Test Phase was executed to completion.
- The Benchmark Configuration was taken through an orderly shutdown process and powered off.
- The Benchmark Configuration was powered on and taken through an orderly startup process.
- The PERSIST_2_0 Test Phase was executed to completion.

Data Persistence Test Results

Data Persistence Test Phase: Persist1	
Total Number of Logical Blocks Written	53,179,955
Total Number of Logical Blocks Verified	30,910,135
Total Number of Logical Blocks Overwritten	22,269,820
Total Number of Logical Blocks that Failed Verification	0
Time Duration for Writing Test Logical Blocks (sec.)	301
Size in Bytes of each Logical Block	8,192
Number of Failed I/O Requests During the Test	0

Committed Data Persistence Implementation

The TSC uses cache protection technology. Each MS3000G2 Engine has two batteries and four system disks. When an accidental power loss occurs, the controllers continue to be powered by the built-in batteries and are able to flush the data in their cache to permanent storage on their system disks. When power is restored and the Engine is restarted, the data on the system disks is used for automatic recovery.

APPENDIX A: SUPPORTING FILES

The following table details the content of the Supporting Files provided as part of this Full Disclosure Report.

File Name	Description	Location
/SPC1_RESULTS	**Data reduction worksheets**	**root**
SPC1_INIT_0_Raw_Results.xlsx	Raw results for INIT Test Phase	/SPC1_RESULTS
SPC1_METRICS_0_Quick_Look.xlsx	Quick Look Test Run Overview	/SPC1_RESULTS
SPC1_METRICS_0_Raw_Results.xlsx	Raw results for Primary Metrics Test	/SPC1_RESULTS
SPC1_METRICS_0_Summary_Results.xlsx	Primary Metrics Summary	/SPC1_RESULTS
SPC1_PERSIST_1_0_Raw_Results.xlsx	Raw results for PERSIST1 Test Phase	/SPC1_RESULTS
SPC1_PERSIST_2_0_Raw_Results.xlsx	Raw results for PERSIST2 Test Phase	/SPC1_RESULTS
SPC1_Run_Set_Overview.xlsx	Run Set Overview Worksheet	/SPC1_RESULTS
SPC1_VERIFY_0_Raw_Results.xlsx	Raw results for first VERIFY Test Phase	/SPC1_RESULTS
SPC1_VERIFY_1_Raw_Results.xlsx	Raw results for second VERIFY Test Phase	/SPC1_RESULTS
/C_Tuning	**Tuning parameters and options**	**root**
aio-max-nr.sh	Set maximum asynchronous I/O	/C_Tuning
nr_request.sh	Increase disk queue depth	/C_Tuning
scheduler.sh	Change the I/O scheduler	/C_Tuning
/D_Creation	**Storage configuration creation**	**root**
init_ms3000g2.sh	Create the storage environment	/D_Creation
mklvm_ms3000g2.sh	Create the Logical Volumes	/D_Creation
lv_scan.sh	Scan the Logical Volumes	/D_Creation
/E_Inventory	**Configuration inventory**	**root**
volume_list.sh	Captures list of Logical Volumes	/E_Inventory
profile_ms3000g2.sh	Captures storage devices profiles	/E_Inventory
volume_listing_start.txt	List of logical volumes before INIT	/E_Inventory
volume_listing_end.txt	List of logical volumes after restart	/E_Inventory
profile_end_ms3000g2.txt	List of storage devices after restart	/E_Inventory
/F_Generator	**Workload generator**	**root**
slave_asu.asu	Defining LUNs hosting the ASUs	/F_generator
2host.HST	Host configuration file	/F_generator
ms3000g2_test_phase1.sh	Executing test phases up to VERIFY_1	/F_generator
ms3000g2_test_persist1.sh	Executing test phase PERSIST_1	/F_generator
ms3000g2_test_phase2.sh	Executing test phase PERSIST_2	/F_generator

APPENDIX B: THIRD PARTY QUOTATION

All components are directly available through the Test Sponsor.

APPENDIX C: TUNING PARAMETERS AND OPTIONS

The following scripts, listed below, were used to set tuning parameters and options:

- ***aio-max-nr.sh*** to set the maximum asynchronous I/O
- ***nr_request.sh*** to change the I/O scheduler
- ***scheduler.sh*** to increase the disk queue depth

The scripts described above are included in the Supporting Files (see Appendix A) and listed below.

aio-max-nr.sh

```
echo 1048576 > /proc/sys/fs/aio-max-nr
cat   /proc/sys/fs/aio-max-nr
```

nr_request.sh

```
#!/bin/bash

#default:
for i in sdc sdd sdb sde sdg sdf sdh sdj sdi sdk sdm sdl sdn sdp sdo sdq
do
    echo 1024 > /sys/block/$i/queue/nr_requests
done

#show
cat -n /sys/block/sd[b-q]/queue/nr_requests
```

scheduler.sh

```
#!/bin/bash

#default: noop anticipatory deadline [cfq]
for i in sdc sdd sdb sde sdg sdf sdh sdj sdi sdk sdm sdl sdn sdp sdo sdq
do
    echo noop > /sys/block/$i/queue/scheduler
done

#show
cat -n /sys/block/sd[b-q]/queue/scheduler
```

APPENDIX D: STORAGE CONFIGURATION CREATION

Step 1 - Create Storage Pool, RAIDs, LUNs, Clients, Mapping

The ***init_ms3000g2.sh*** script, listed below, is executed on either of the Storage Controllers via a CLI session to perform the following actions:

1. Create a storage pool for each Engine: Pool-A-1 and Pool-B-1
2. Create 8 RAID10
3. Create 16 LUNs (2 LUNs per RAID10, 667GB per LUN)
4. Create 16 Clients in each Engine
5. Add Host System FC Initiators for the Clients (1 Initiator per Client)
6. Create Storage Targets for the Clients (1 Target per Client and each Target is associated with 1 FC port)
7. Add a LUN to the Targets (1 LUN per Target)

The script file described above is included in the Supporting Files (see Appendix A) and listed below.

init_ms3000g2.sh

```
#!/bin/bash

#Create 2*Pool:Pool-A-1/Pool-B-1

ms-cli EngineA "pool mgt create -n Pool-A-1 -t t"
ms-cli EngineB "pool mgt create -n Pool-B-1 -t t"

#Create RAID: 8 * RAID10

echo " waiting for create EngineA RAID "
ms-cli  EngineA  "raid  mgt  create  -n  RAID-1  -p  Pool-A-1  -l  RAID10  -d
    1:1:1:5,1:1:1:6,1:1:1:7,1:1:1:8,1:1:1:9,1:1:1:10"
ms-cli  EngineA  "raid  mgt  create  -n  RAID-2  -p  Pool-A-1  -l  RAID10  -d
    1:2:1:1,1:2:1:2,1:2:1:3,1:2:1:4,1:2:1:5,1:2:1:6"
ms-cli  EngineA  "raid  mgt  create  -n  RAID-3  -p  Pool-A-1  -l  RAID10  -d
    1:2:1:9,1:2:1:10,1:2:1:11,1:2:1:12,1:2:1:13,1:2:1:14"
ms-cli  EngineA  "raid  mgt  create  -n  RAID-4  -p  Pool-A-1  -l  RAID10  -d
    1:2:1:17,1:2:1:18,1:2:1:19,1:2:1:20,1:2:1:21,1:2:1:22"

echo " waiting for create EngineB RAID "
ms-cli  EngineB  "raid  mgt  create  -n  RAID10-1  -p  Pool-B-1  -l  RAID10  -d
    1:1:1:5,1:1:1:6,1:1:1:7,1:1:1:8,1:1:1:9,1:1:1:10"
ms-cli  EngineB  "raid  mgt  create  -n  RAID10-2  -p  Pool-B-1  -l  RAID10  -d
    1:2:1:1,1:2:1:2,1:2:1:3,1:2:1:4,1:2:1:5,1:2:1:6"
ms-cli  EngineB  "raid  mgt  create  -n  RAID10-3  -p  Pool-B-1  -l  RAID10  -d
    1:2:1:9,1:2:1:10,1:2:1:11,1:2:1:12,1:2:1:13,1:2:1:14"
ms-cli  EngineB  "raid  mgt  create  -n  RAID10-4  -p  Pool-B-1  -l  RAID10  -d
    1:2:1:17,1:2:1:18,1:2:1:19,1:2:1:20,1:2:1:21,1:2:1:22"

#Create LUN: 16
echo "create LUN"
```

```
ms-cli EngineA "lun mgt createthicklun -n LUN-A-1-1 -o SP1 -p Pool-A-1 -r disable
    -w disable -m no-sync -s 667 -R RAID-1"
ms-cli EngineA "lun mgt createthicklun -n LUN-A-1-2 -o SP2 -p Pool-A-1 -r disable
    -w disable -m no-sync -s 667 -R RAID-1"
ms-cli EngineA "lun mgt createthicklun -n LUN-A-2-1 -o SP1 -p Pool-A-1 -r disable
    -w disable -m no-sync -s 667 -R RAID-2"
ms-cli EngineA "lun mgt createthicklun -n LUN-A-2-2 -o SP2 -p Pool-A-1 -r disable
    -w disable -m no-sync -s 667 -R RAID-2"
ms-cli EngineA "lun mgt createthicklun -n LUN-A-3-1 -o SP1 -p Pool-A-1 -r disable
    -w disable -m no-sync -s 667 -R RAID-3"
ms-cli EngineA "lun mgt createthicklun -n LUN-A-3-2 -o SP2 -p Pool-A-1 -r disable
    -w disable -m no-sync -s 667 -R RAID-3"
ms-cli EngineA "lun mgt createthicklun -n LUN-A-4-1 -o SP1 -p Pool-A-1 -r disable
    -w disable -m no-sync -s 667 -R RAID-4"
ms-cli EngineA "lun mgt createthicklun -n LUN-A-4-2 -o SP2 -p Pool-A-1 -r disable
    -w disable -m no-sync -s 667 -R RAID-4"

ms-cli EngineB "lun mgt createthicklun -n LUN-B-1-1 -o SP1 -p Pool-B-1 -r disable
    -w disable -m no-sync -s 667 -R RAID10-1"
ms-cli EngineB "lun mgt createthicklun -n LUN-B-1-2 -o SP2 -p Pool-B-1 -r disable
    -w disable -m no-sync -s 667 -R RAID10-1"
ms-cli EngineB "lun mgt createthicklun -n LUN-B-2-1 -o SP1 -p Pool-B-1 -r disable
    -w disable -m no-sync -s 667 -R RAID10-2"
ms-cli EngineB "lun mgt createthicklun -n LUN-B-2-2 -o SP2 -p Pool-B-1 -r disable
    -w disable -m no-sync -s 667 -R RAID10-2"
ms-cli EngineB "lun mgt createthicklun -n LUN-B-3-1 -o SP1 -p Pool-B-1 -r disable
    -w disable -m no-sync -s 667 -R RAID10-3"
ms-cli EngineB "lun mgt createthicklun -n LUN-B-3-2 -o SP2 -p Pool-B-1 -r disable
    -w disable -m no-sync -s 667 -R RAID10-3"
ms-cli EngineB "lun mgt createthicklun -n LUN-B-4-2 -o SP2 -p Pool-B-1 -r disable
    -w disable -m no-sync -s 667 -R RAID10-4"
ms-cli EngineB "lun mgt createthicklun -n LUN-B-4-1 -o SP1 -p Pool-B-1 -r disable
    -w disable -m no-sync -s 667 -R RAID10-4"

#Add Initiator with Client:32

ms-cli EngineA "client initiator modifyalias -w 31:61:00:24:ff:af:04:07 -a FC-
    1:1:1"
ms-cli EngineA "client initiator modifyalias -w 21:00:00:24:ff:71:64:e5 -a FC-
    1:1:2"
ms-cli EngineA "client initiator modifyalias -w 21:00:00:24:ff:69:d1:c8 -a FC-
    1:1:3"
ms-cli EngineA "client initiator modifyalias -w 21:00:00:24:ff:69:b6:7e -a FC-
    1:1:4"
ms-cli EngineA "client initiator modifyalias -w 11:c9:00:24:ff:0a:96:0f -a FC-
    1:2:1"
ms-cli EngineA "client initiator modifyalias -w 11:61:00:24:ff:af:04:07 -a FC-
    1:2:2"
ms-cli EngineA "client initiator modifyalias -w 21:00:00:24:ff:69:d1:cb -a FC-
    1:2:3"
ms-cli EngineA "client initiator modifyalias -w 21:00:00:24:ff:66:f2:38 -a FC-
    1:2:4"
ms-cli EngineA "client initiator modifyalias -w 31:c9:00:24:ff:0a:96:0f -a FC-
    2:1:1"
ms-cli EngineA "client initiator modifyalias -w 11:c7:00:24:ff:7b:d3:3f -a FC-
    2:1:2"
ms-cli EngineA "client initiator modifyalias -w 21:00:00:24:ff:67:37:04 -a FC-
    2:1:3"
ms-cli EngineA "client initiator modifyalias -w 21:00:00:24:ff:66:f2:3b -a FC-
    2:1:4"
```

```
ms-cli EngineA "client initiator modifyalias -w 01:61:00:24:ff:af:04:07 -a FC-
    2:2:1"
ms-cli EngineA "client initiator modifyalias -w 01:c9:00:24:ff:0a:96:0f -a FC-
    2:2:2"
ms-cli EngineA "client initiator modifyalias -w 21:00:00:24:ff:69:b6:7d -a FC-
    2:2:3"
ms-cli EngineA "client initiator modifyalias -w 21:00:00:24:ff:66:f2:39 -a FC-
    2:2:4"

ms-cli EngineB "client initiator modifyalias -w 01:c7:00:24:ff:7b:d3:3f -a FC-
    1:1:1"
ms-cli EngineB "client initiator modifyalias -w 21:c7:00:24:ff:7b:d3:3f -a FC-
    1:1:2"
ms-cli EngineB "client initiator modifyalias -w 21:00:00:24:ff:69:d1:c9 -a FC-
    1:1:3"
ms-cli EngineB "client initiator modifyalias -w 21:00:00:24:ff:67:37:06 -a FC-
    1:1:4"
ms-cli EngineB "client initiator modifyalias -w 31:c7:00:24:ff:7b:d3:3f -a FC-
    1:2:1"
ms-cli EngineB "client initiator modifyalias -w 21:00:00:24:ff:71:64:e7 -a FC-
    1:2:2"
ms-cli EngineB "client initiator modifyalias -w 21:00:00:24:ff:66:f2:3a -a FC-
    1:2:3"
ms-cli EngineB "client initiator modifyalias -w 21:00:00:24:ff:69:b6:7f -a FC-
    1:2:4"
ms-cli EngineB "client initiator modifyalias -w 21:00:00:24:ff:71:64:e4 -a FC-
    2:1:1"
ms-cli EngineB "client initiator modifyalias -w 21:00:00:24:ff:71:64:e6 -a FC-
    2:1:2"
ms-cli EngineB "client initiator modifyalias -w 21:00:00:24:ff:67:37:05 -a FC-
    2:1:3"
ms-cli EngineB "client initiator modifyalias -w 21:00:00:24:ff:67:37:07 -a FC-
    2:1:4"
ms-cli EngineB "client initiator modifyalias -w 21:c9:00:24:ff:0a:96:0f -a FC-
    2:2:1"
ms-cli EngineB "client initiator modifyalias -w 21:61:00:24:ff:af:04:07 -a FC-
    2:2:2"
ms-cli EngineB "client initiator modifyalias -w 21:00:00:24:ff:69:b6:7c -a FC-
    2:2:3"
ms-cli EngineB "client initiator modifyalias -w 21:00:00:24:ff:69:d1:ca -a FC-
    2:2:4"

#Change Initiator OS modul:32

ms-cli EngineA "client initiator modifyos -w 31:61:00:24:ff:af:04:07 -o Linux"
ms-cli EngineA "client initiator modifyos -w 21:00:00:24:ff:71:64:e5 -o Linux"
ms-cli EngineA "client initiator modifyos -w 21:00:00:24:ff:69:d1:c8 -o Linux"
ms-cli EngineA "client initiator modifyos -w 21:00:00:24:ff:69:b6:7e -o Linux"
ms-cli EngineA "client initiator modifyos -w 11:c9:00:24:ff:0a:96:0f -o Linux"
ms-cli EngineA "client initiator modifyos -w 11:61:00:24:ff:af:04:07 -o Linux"
ms-cli EngineA "client initiator modifyos -w 21:00:00:24:ff:69:d1:cb -o Linux"
ms-cli EngineA "client initiator modifyos -w 21:00:00:24:ff:66:f2:38 -o Linux"
ms-cli EngineA "client initiator modifyos -w 31:c9:00:24:ff:0a:96:0f -o Linux"
ms-cli EngineA "client initiator modifyos -w 11:c7:00:24:ff:7b:d3:3f -o Linux"
ms-cli EngineA "client initiator modifyos -w 21:00:00:24:ff:67:37:04 -o Linux"
ms-cli EngineA "client initiator modifyos -w 21:00:00:24:ff:66:f2:3b -o Linux"
ms-cli EngineA "client initiator modifyos -w 01:61:00:24:ff:af:04:07 -o Linux"
ms-cli EngineA "client initiator modifyos -w 01:c9:00:24:ff:0a:96:0f -o Linux"
ms-cli EngineA "client initiator modifyos -w 21:00:00:24:ff:69:b6:7d -o Linux"
ms-cli EngineA "client initiator modifyos -w 21:00:00:24:ff:66:f2:39 -o Linux"
```

```
ms-cli EngineB "client initiator modifyos -w 01:c7:00:24:ff:7b:d3:3f -o Linux"
ms-cli EngineB "client initiator modifyos -w 21:c7:00:24:ff:7b:d3:3f -o Linux"
ms-cli EngineB "client initiator modifyos -w 21:00:00:24:ff:69:d1:c9 -o Linux"
ms-cli EngineB "client initiator modifyos -w 21:00:00:24:ff:67:37:06 -o Linux"
ms-cli EngineB "client initiator modifyos -w 31:c7:00:24:ff:7b:d3:3f -o Linux"
ms-cli EngineB "client initiator modifyos -w 21:00:00:24:ff:71:64:e7 -o Linux"
ms-cli EngineB "client initiator modifyos -w 21:00:00:24:ff:66:f2:3a -o Linux"
ms-cli EngineB "client initiator modifyos -w 21:00:00:24:ff:69:b6:7f -o Linux"
ms-cli EngineB "client initiator modifyos -w 21:00:00:24:ff:71:64:e4 -o Linux"
ms-cli EngineB "client initiator modifyos -w 21:00:00:24:ff:71:64:e6 -o Linux"
ms-cli EngineB "client initiator modifyos -w 21:00:00:24:ff:67:37:05 -o Linux"
ms-cli EngineB "client initiator modifyos -w 21:00:00:24:ff:67:37:07 -o Linux"
ms-cli EngineB "client initiator modifyos -w 21:c9:00:24:ff:0a:96:0f -o Linux"
ms-cli EngineB "client initiator modifyos -w 21:61:00:24:ff:af:04:07 -o Linux"
ms-cli EngineB "client initiator modifyos -w 21:00:00:24:ff:69:b6:7c -o Linux"
ms-cli EngineB "client initiator modifyos -w 21:00:00:24:ff:69:d1:ca -o Linux"

#Add Target: 32
ms-cli EngineA "client target create -t fc -p FC-1:1:1"
ms-cli EngineA "client target create -t fc -p FC-1:1:2"
ms-cli EngineA "client target create -t fc -p FC-1:1:3"
ms-cli EngineA "client target create -t fc -p FC-1:1:4"
ms-cli EngineA "client target create -t fc -p FC-1:2:1"
ms-cli EngineA "client target create -t fc -p FC-1:2:2"
ms-cli EngineA "client target create -t fc -p FC-1:2:3"
ms-cli EngineA "client target create -t fc -p FC-1:2:4"
ms-cli EngineA "client target create -t fc -p FC-2:1:1"
ms-cli EngineA "client target create -t fc -p FC-2:1:2"
ms-cli EngineA "client target create -t fc -p FC-2:1:3"
ms-cli EngineA "client target create -t fc -p FC-2:1:4"
ms-cli EngineA "client target create -t fc -p FC-2:2:1"
ms-cli EngineA "client target create -t fc -p FC-2:2:2"
ms-cli EngineA "client target create -t fc -p FC-2:2:3"
ms-cli EngineA "client target create -t fc -p FC-2:2:4"

ms-cli EngineB "client target create -t fc -p FC-1:1:1"
ms-cli EngineB "client target create -t fc -p FC-1:1:2"
ms-cli EngineB "client target create -t fc -p FC-1:1:3"
ms-cli EngineB "client target create -t fc -p FC-1:1:4"
ms-cli EngineB "client target create -t fc -p FC-1:2:1"
ms-cli EngineB "client target create -t fc -p FC-1:2:2"
ms-cli EngineB "client target create -t fc -p FC-1:2:3"
ms-cli EngineB "client target create -t fc -p FC-1:2:4"
ms-cli EngineB "client target create -t fc -p FC-2:1:1"
ms-cli EngineB "client target create -t fc -p FC-2:1:2"
ms-cli EngineB "client target create -t fc -p FC-2:1:3"
ms-cli EngineB "client target create -t fc -p FC-2:1:4"
ms-cli EngineB "client target create -t fc -p FC-2:2:1"
ms-cli EngineB "client target create -t fc -p FC-2:2:2"
ms-cli EngineB "client target create -t fc -p FC-2:2:3"
ms-cli EngineB "client target create -t fc -p FC-2:2:4"

#map-target

ms-cli EngineA "client itl maptarget  -i 31:61:00:24:ff:af:04:07 -t FC-Target-
    1:1:1"
ms-cli EngineA "client itl maptarget  -i 21:00:00:24:ff:71:64:e5 -t FC-Target-
    1:1:2"
ms-cli EngineA "client itl maptarget  -i 21:00:00:24:ff:69:d1:c8 -t FC-Target-
    1:1:3"
```

```
ms-cli EngineA "client itl maptarget  -i 21:00:00:24:ff:69:b6:7e -t FC-Target-
    1:1:4"
ms-cli EngineA "client itl maptarget  -i 11:c9:00:24:ff:0a:96:0f -t FC-Target-
    1:2:1"
ms-cli EngineA "client itl maptarget  -i 11:61:00:24:ff:af:04:07 -t FC-Target-
    1:2:2"
ms-cli EngineA "client itl maptarget  -i 21:00:00:24:ff:69:d1:cb -t FC-Target-
    1:2:3"
ms-cli EngineA "client itl maptarget  -i 21:00:00:24:ff:66:f2:38 -t FC-Target-
    1:2:4"
ms-cli EngineA "client itl maptarget  -i 31:c9:00:24:ff:0a:96:0f -t FC-Target-
    2:1:1"
ms-cli EngineA "client itl maptarget  -i 11:c7:00:24:ff:7b:d3:3f -t FC-Target-
    2:1:2"
ms-cli EngineA "client itl maptarget  -i 21:00:00:24:ff:67:37:04 -t FC-Target-
    2:1:3"
ms-cli EngineA "client itl maptarget  -i 21:00:00:24:ff:66:f2:3b -t FC-Target-
    2:1:4"
ms-cli EngineA "client itl maptarget  -i 01:61:00:24:ff:af:04:07 -t FC-Target-
    2:2:1"
ms-cli EngineA "client itl maptarget  -i 01:c9:00:24:ff:0a:96:0f -t FC-Target-
    2:2:2"
ms-cli EngineA "client itl maptarget  -i 21:00:00:24:ff:69:b6:7d -t FC-Target-
    2:2:3"
ms-cli EngineA "client itl maptarget  -i 21:00:00:24:ff:66:f2:39 -t FC-Target-
    2:2:4"

ms-cli EngineB "client itl maptarget  -i 01:c7:00:24:ff:7b:d3:3f -t FC-Target-
    1:1:1"
ms-cli EngineB "client itl maptarget  -i 21:c7:00:24:ff:7b:d3:3f -t FC-Target-
    1:1:2"
ms-cli EngineB "client itl maptarget  -i 21:00:00:24:ff:69:d1:c9 -t FC-Target-
    1:1:3"
ms-cli EngineB "client itl maptarget  -i 21:00:00:24:ff:67:37:06 -t FC-Target-
    1:1:4"
ms-cli EngineB "client itl maptarget  -i 31:c7:00:24:ff:7b:d3:3f -t FC-Target-
    1:2:1"
ms-cli EngineB "client itl maptarget  -i 21:00:00:24:ff:71:64:e7 -t FC-Target-
    1:2:2"
ms-cli EngineB "client itl maptarget  -i 21:00:00:24:ff:66:f2:3a -t FC-Target-
    1:2:3"
ms-cli EngineB "client itl maptarget  -i 21:00:00:24:ff:69:b6:7f -t FC-Target-
    1:2:4"
ms-cli EngineB "client itl maptarget  -i 21:00:00:24:ff:71:64:e4 -t FC-Target-
    2:1:1"
ms-cli EngineB "client itl maptarget  -i 21:00:00:24:ff:71:64:e6 -t FC-Target-
    2:1:2"
ms-cli EngineB "client itl maptarget  -i 21:00:00:24:ff:67:37:05 -t FC-Target-
    2:1:3"
ms-cli EngineB "client itl maptarget  -i 21:00:00:24:ff:67:37:07 -t FC-Target-
    2:1:4"
ms-cli EngineB "client itl maptarget  -i 21:c9:00:24:ff:0a:96:0f -t FC-Target-
    2:2:1"
ms-cli EngineB "client itl maptarget  -i 21:61:00:24:ff:af:04:07 -t FC-Target-
    2:2:2"
ms-cli EngineB "client itl maptarget  -i 21:00:00:24:ff:69:b6:7c -t FC-Target-
    2:2:3"
ms-cli EngineB "client itl maptarget  -i 21:00:00:24:ff:69:d1:ca -t FC-Target-
    2:2:4"

#map LUN 32
```

```
ms-cli EngineA "client itl maplun -i 31:61:00:24:ff:af:04:07 -t FC-Target-1:1:1
    -l LUN-A-1-1 -I 0 -a rw"
ms-cli EngineA "client itl maplun -i 21:00:00:24:ff:71:64:e5 -t FC-Target-1:1:2
    -l LUN-A-2-1 -I 0 -a rw"
ms-cli EngineA "client itl maplun -i 11:c9:00:24:ff:0a:96:0f -t FC-Target-1:2:1
    -l LUN-A-3-1 -I 0 -a rw"
ms-cli EngineA "client itl maplun -i 11:61:00:24:ff:af:04:07 -t FC-Target-1:2:2
    -l LUN-A-4-1 -I 0 -a rw"
ms-cli EngineA "client itl maplun -i 31:c9:00:24:ff:0a:96:0f -t FC-Target-2:1:1
    -l LUN-A-1-2 -I 0 -a rw"
ms-cli EngineA "client itl maplun -i 11:c7:00:24:ff:7b:d3:3f -t FC-Target-2:1:2
    -l LUN-A-2-2 -I 0 -a rw"
ms-cli EngineA "client itl maplun -i 01:61:00:24:ff:af:04:07 -t FC-Target-2:2:1
    -l LUN-A-3-2 -I 0 -a rw"
ms-cli EngineA "client itl maplun -i 01:c9:00:24:ff:0a:96:0f -t FC-Target-2:2:2
    -l LUN-A-4-2 -I 0 -a rw"
ms-cli EngineA "client itl maplun -i 21:00:00:24:ff:69:d1:c8 -t FC-Target-1:1:3
    -l LUN-A-1-1 -I 0 -a rw"
ms-cli EngineA "client itl maplun -i 21:00:00:24:ff:69:b6:7e -t FC-Target-1:1:4
    -l LUN-A-2-1 -I 0 -a rw"
ms-cli EngineA "client itl maplun -i 21:00:00:24:ff:69:d1:cb -t FC-Target-1:2:3
    -l LUN-A-3-1 -I 0 -a rw"
ms-cli EngineA "client itl maplun -i 21:00:00:24:ff:66:f2:38 -t FC-Target-1:2:4
    -l LUN-A-4-1 -I 0 -a rw"
ms-cli EngineA "client itl maplun -i 21:00:00:24:ff:67:37:04 -t FC-Target-2:1:3
    -l LUN-A-1-2 -I 0 -a rw"
ms-cli EngineA "client itl maplun -i 21:00:00:24:ff:66:f2:3b -t FC-Target-2:1:4
    -l LUN-A-2-2 -I 0 -a rw"
ms-cli EngineA "client itl maplun -i 21:00:00:24:ff:69:b6:7d -t FC-Target-2:2:3
    -l LUN-A-3-2 -I 0 -a rw"
ms-cli EngineA "client itl maplun -i 21:00:00:24:ff:66:f2:39 -t FC-Target-2:2:4
    -l LUN-A-4-2 -I 0 -a rw"

ms-cli EngineB "client itl maplun -i 01:c7:00:24:ff:7b:d3:3f -t FC-Target-1:1:1
    -l LUN-B-2-1 -I 0 -a rw"
ms-cli EngineB "client itl maplun -i 21:c7:00:24:ff:7b:d3:3f -t FC-Target-1:1:2
    -l LUN-B-1-1 -I 0 -a rw"
ms-cli EngineB "client itl maplun -i 31:c7:00:24:ff:7b:d3:3f -t FC-Target-1:2:1
    -l LUN-B-3-1 -I 0 -a rw"
ms-cli EngineB "client itl maplun -i 21:00:00:24:ff:71:64:e7 -t FC-Target-1:2:2
    -l LUN-B-4-1 -I 0 -a rw"
ms-cli EngineB "client itl maplun -i 21:00:00:24:ff:71:64:e4 -t FC-Target-2:1:1
    -l LUN-B-2-2 -I 0 -a rw"
ms-cli EngineB "client itl maplun -i 21:00:00:24:ff:71:64:e6 -t FC-Target-2:1:2
    -l LUN-B-1-2 -I 0 -a rw"
ms-cli EngineB "client itl maplun -i 21:c9:00:24:ff:0a:96:0f -t FC-Target-2:2:1
    -l LUN-B-3-2 -I 0 -a rw"
ms-cli EngineB "client itl maplun -i 21:61:00:24:ff:af:04:07 -t FC-Target-2:2:2
    -l LUN-B-4-2 -I 0 -a rw"
ms-cli EngineB "client itl maplun -i 21:00:00:24:ff:67:37:06 -t FC-Target-1:1:4
    -l LUN-B-1-1 -I 0 -a rw"
ms-cli EngineB "client itl maplun -i 21:00:00:24:ff:69:d1:c9 -t FC-Target-1:1:3
    -l LUN-B-2-1 -I 0 -a rw"
ms-cli EngineB "client itl maplun -i 21:00:00:24:ff:66:f2:3a -t FC-Target-1:2:3
    -l LUN-B-3-1 -I 0 -a rw"
ms-cli EngineB "client itl maplun -i 21:00:00:24:ff:69:b6:7f -t FC-Target-1:2:4
    -l LUN-B-4-1 -I 0 -a rw"
ms-cli EngineB "client itl maplun -i 21:00:00:24:ff:67:37:07 -t FC-Target-2:1:4
    -l LUN-B-1-2 -I 0 -a rw"
ms-cli EngineB "client itl maplun -i 21:00:00:24:ff:67:37:05 -t FC-Target-2:1:3
    -l LUN-B-2-2 -I 0 -a rw"
```

```
ms-cli EngineB "client itl maplun -i 21:00:00:24:ff:69:b6:7c -t FC-Target-2:2:3
    -l LUN-B-3-2 -I 0 -a rw"
ms-cli EngineB "client itl maplun -i 21:00:00:24:ff:69:d1:ca -t FC-Target-2:2:4
    -l LUN-B-4-2 -I 0 -a rw"

echo "complete "
exit 0
```

Step 2 - Create Volumes on the Master Host System

The script ***mklvm_ms3000g2.sh*** is executed on the Master Host System to create 38 logical volumes as follows:

1. Create Physical Volume:

   Create 16 physical volumes using the pvcreate command.

2. Create Volumes Groups:

   Create 1 volume group (vg11) using the vgcreate command over the 16 physical volumes: /dev/sdb, /dev/sdc, /dev/sdd, /dev/sde, /dev/sdf, /dev/sdg, /dev/sdh, /dev/sdi, /dev/sdj, /dev/sdk, /dev/sdl, /dev/sdm, /dev/sdn, /dev/sdo, /dev/sdp, /dev/sdq.

3. Create Logical Volumes:

   Create 18 logical volumes with a volume capacity of 266GB, on vg11 for ASU-1.

   Create 18 logical volumes with a volume capacity of 266GB, on vg11 for ASU-2.

   Create 2 logical volumes with a volume capacity of 532GB, on vg11 for ASU-3.

4. Scan Physical Volumes, Volume Group, and Logical Volumes; and activate each Logical Volume:

   Execute the ***lv_scan.sh*** on the Slave Host Systems to scan the physical volumes, volume group and logical volumes; and make each logical volume available (activate).

The script files described above are included in the Supporting Files (see Appendix A) and listed below.

mklvm_ms3000g2.sh

```
#!/bin/bash
#create_pv
pvcreate /dev/sdb
pvcreate /dev/sdc
pvcreate /dev/sdd
pvcreate /dev/sde
pvcreate /dev/sdf
pvcreate /dev/sdg
pvcreate /dev/sdh
pvcreate /dev/sdi
pvcreate /dev/sdj
pvcreate /dev/sdk
pvcreate /dev/sdl
pvcreate /dev/sdm
pvcreate /dev/sdn
pvcreate /dev/sdo
pvcreate /dev/sdp
pvcreate /dev/sdq

#create_vg
vgcreate  vg11  /dev/sdb  /dev/sdc  /dev/sdd  /dev/sde  /dev/sdf  /dev/sdg  /dev/sdh
   /dev/sdi  /dev/sdj  /dev/sdk  /dev/sdl  /dev/sdm  /dev/sdn  /dev/sdo  /dev/sdp
   /dev/sdq

#create_lv
lvcreate -n asu11  -i 16 -I 512 -C y -L 266g vg11
lvcreate -n asu12  -i 16 -I 512 -C y -L 266g vg11
lvcreate -n asu13  -i 16 -I 512 -C y -L 266g vg11
lvcreate -n asu14  -i 16 -I 512 -C y -L 266g vg11
lvcreate -n asu15  -i 16 -I 512 -C y -L 266g vg11
lvcreate -n asu16  -i 16 -I 512 -C y -L 266g vg11
lvcreate -n asu17  -i 16 -I 512 -C y -L 266g vg11
lvcreate -n asu18  -i 16 -I 512 -C y -L 266g vg11
lvcreate -n asu19  -i 16 -I 512 -C y -L 266g vg11
lvcreate -n asu110 -i 16 -I 512 -C y -L 266g vg11
lvcreate -n asu111 -i 16 -I 512 -C y -L 266g vg11
lvcreate -n asu112 -i 16 -I 512 -C y -L 266g vg11
lvcreate -n asu113 -i 16 -I 512 -C y -L 266g vg11
lvcreate -n asu114 -i 16 -I 512 -C y -L 266g vg11
lvcreate -n asu115 -i 16 -I 512 -C y -L 266g vg11
lvcreate -n asu116 -i 16 -I 512 -C y -L 266g vg11
lvcreate -n asu117 -i 16 -I 512 -C y -L 266g vg11
lvcreate -n asu118 -i 16 -I 512 -C y -L 266g vg11

lvcreate -n asu21  -i 16 -I 512 -C y -L 266g vg11
lvcreate -n asu22  -i 16 -I 512 -C y -L 266g vg11
lvcreate -n asu23  -i 16 -I 512 -C y -L 266g vg11
lvcreate -n asu24  -i 16 -I 512 -C y -L 266g vg11
lvcreate -n asu25  -i 16 -I 512 -C y -L 266g vg11
lvcreate -n asu26  -i 16 -I 512 -C y -L 266g vg11
lvcreate -n asu27  -i 16 -I 512 -C y -L 266g vg11
lvcreate -n asu28  -i 16 -I 512 -C y -L 266g vg11
lvcreate -n asu29  -i 16 -I 512 -C y -L 266g vg11
lvcreate -n asu210 -i 16 -I 512 -C y -L 266g vg11
lvcreate -n asu211 -i 16 -I 512 -C y -L 266g vg11
lvcreate -n asu212 -i 16 -I 512 -C y -L 266g vg11
lvcreate -n asu213 -i 16 -I 512 -C y -L 266g vg11
lvcreate -n asu214 -i 16 -I 512 -C y -L 266g vg11
lvcreate -n asu215 -i 16 -I 512 -C y -L 266g vg11
lvcreate -n asu216 -i 16 -I 512 -C y -L 266g vg11
lvcreate -n asu217 -i 16 -I 512 -C y -L 266g vg11
lvcreate -n asu218 -i 16 -I 512 -C y -L 266g vg11

lvcreate -n asu31  -i 16 -I 512 -C y -L 532g vg11
lvcreate -n asu32  -i 16 -I 512 -C y -L 532g vg11
```

lv_scan.sh

```
#!/bin/bash

echo '---------------pvscan-----------------'
pvscan
echo '---------------vgscan-----------------'
vgscan
echo '---------------lvscan-----------------'
lvscan
```

```
lvchange -ay /dev/vg11/asu11
lvchange -ay /dev/vg11/asu12
lvchange -ay /dev/vg11/asu13
lvchange -ay /dev/vg11/asu14
lvchange -ay /dev/vg11/asu15
lvchange -ay /dev/vg11/asu16
lvchange -ay /dev/vg11/asu17
lvchange -ay /dev/vg11/asu18
lvchange -ay /dev/vg11/asu19
lvchange -ay /dev/vg11/asu110
lvchange -ay /dev/vg11/asu111
lvchange -ay /dev/vg11/asu112
lvchange -ay /dev/vg11/asu113
lvchange -ay /dev/vg11/asu114
lvchange -ay /dev/vg11/asu115
lvchange -ay /dev/vg11/asu116
lvchange -ay /dev/vg11/asu117
lvchange -ay /dev/vg11/asu118

lvchange -ay /dev/vg11/asu21
lvchange -ay /dev/vg11/asu22
lvchange -ay /dev/vg11/asu23
lvchange -ay /dev/vg11/asu24
lvchange -ay /dev/vg11/asu25
lvchange -ay /dev/vg11/asu26
lvchange -ay /dev/vg11/asu27
lvchange -ay /dev/vg11/asu28
lvchange -ay /dev/vg11/asu29
lvchange -ay /dev/vg11/asu210
lvchange -ay /dev/vg11/asu211
lvchange -ay /dev/vg11/asu212
lvchange -ay /dev/vg11/asu213
lvchange -ay /dev/vg11/asu214
lvchange -ay /dev/vg11/asu215
lvchange -ay /dev/vg11/asu216
lvchange -ay /dev/vg11/asu217
lvchange -ay /dev/vg11/asu218

lvchange -ay /dev/vg11/asu31
lvchange -ay /dev/vg11/asu32
```

APPENDIX E: CONFIGURATION INVENTORY

An inventory of the Tested Storage Configuration was collected during the execution of the scripts ***ms3000g2_test_phase1.sh*** and ***ms3000g2_test_phase2.sh***. The following log files were generated:

• ***volume_listing_start.txt***	List of configured storage before the INIT Phase.
• ***profile_end_ms3000g2.txt***	List of configured volumes after TSC restart.
• ***volume_listing_start.txt***	List of configured storage after TSC restart.

The file ***profile_start_ms3000g2.txt*** was incorrectly generated due to a script error. As an alternative, the auditor, which was on-site during the execution of the benchmark, proceeded with a visual inspection of the TSC and its storage devices.

The above log files are included in the Supporting Files (see Appendix A).

APPENDIX F: WORKLOAD GENERATOR

The host parameters for the SPC-1 workload generator were defined using the script ***2host.HST***.

The ASUs accessed by the SPC-1 workload generator are defined using the script ***slave_asu.asu***.

The initial test phases of the benchmark are executed using the scripts ***ms3000g2_test_phase1.sh*** and ***ms3000g2_test_persist1.sh*** until the Persistence Test shutdown. Once the TSC has been restarted, the PERSIST_2 test phase is executed using the script ***ms3000g2_test_phase2.sh***.

The above scripts are included in the Supporting Files (see Appendix A) and listed below.

2host.HST

```
-- SMALL_HOST definition
LOGIN=root
PASSWORD=passwd
CONFIG=/v302
EXEC=Supported/RHEL/7_1/spc1_optimized_v3.0.2
OUTPUT=/v302/Output
PORT=1001
WINDOWS=N
WEIGHT=1
STORAGE=slave_asu.asu
HOST=localhost
PORT=2001
WINDOWS=N
WEIGHT=1
STORAGE=slave_asu.asu
HOST=localhost
PORT=3001
WINDOWS=N
WEIGHT=1
STORAGE=slave_asu.asu
HOST=localhost
PORT=4001
WINDOWS=N
WEIGHT=1
STORAGE=slave_asu.asu
HOST=localhost
PORT=5001
WINDOWS=N
WEIGHT=1
STORAGE=slave_asu.asu
HOST=localhost
PORT=6001
WINDOWS=N
WEIGHT=1
STORAGE=slave_asu.asu
HOST=localhost

LOGIN=root
```

```
PASSWORD=passwd
CONFIG=/v302
EXEC=Supported/RHEL/7_1/spc1_optimized_v3.0.2
OUTPUT=/v302/Output
PORT=1001
WINDOWS=N
WEIGHT=1
STORAGE=slave_asu.asu
HOST=172.0.65.78
PORT=2001
WINDOWS=N
WEIGHT=1
STORAGE=slave_asu.asu
HOST=172.0.65.78
PORT=3001
WINDOWS=N
WEIGHT=1
STORAGE=slave_asu.asu
HOST=172.0.65.78
PORT=4001
WINDOWS=N
WEIGHT=1
STORAGE=slave_asu.asu
HOST=172.0.65.78
PORT=5001
WINDOWS=N
WEIGHT=1
STORAGE=slave_asu.asu
HOST=172.0.65.78
PORT=6001
WINDOWS=N
WEIGHT=1
STORAGE=slave_asu.asu
HOST=172.0.65.78
```

slave_asu.asu

```
ASU=1
OFFSET=0
SIZE=0
DEVICE=/dev/vg11/asu11
DEVICE=/dev/vg11/asu12
DEVICE=/dev/vg11/asu13
DEVICE=/dev/vg11/asu14
DEVICE=/dev/vg11/asu15
DEVICE=/dev/vg11/asu16
DEVICE=/dev/vg11/asu17
DEVICE=/dev/vg11/asu18
DEVICE=/dev/vg11/asu19
DEVICE=/dev/vg11/asu110
DEVICE=/dev/vg11/asu111
DEVICE=/dev/vg11/asu112
DEVICE=/dev/vg11/asu113
DEVICE=/dev/vg11/asu114
DEVICE=/dev/vg11/asu115
DEVICE=/dev/vg11/asu116
DEVICE=/dev/vg11/asu117
DEVICE=/dev/vg11/asu118
--
ASU=2
```

```
OFFSET=0
SIZE=0
DEVICE=/dev/vg11/asu21
DEVICE=/dev/vg11/asu22
DEVICE=/dev/vg11/asu23
DEVICE=/dev/vg11/asu24
DEVICE=/dev/vg11/asu25
DEVICE=/dev/vg11/asu26
DEVICE=/dev/vg11/asu27
DEVICE=/dev/vg11/asu28
DEVICE=/dev/vg11/asu29
DEVICE=/dev/vg11/asu210
DEVICE=/dev/vg11/asu211
DEVICE=/dev/vg11/asu212
DEVICE=/dev/vg11/asu213
DEVICE=/dev/vg11/asu214
DEVICE=/dev/vg11/asu215
DEVICE=/dev/vg11/asu216
DEVICE=/dev/vg11/asu217
DEVICE=/dev/vg11/asu218
--
ASU=3
OFFSET=0
SIZE=0
DEVICE=/dev/vg11/asu31
DEVICE=/dev/vg11/asu32
```

ms3000g2_test_phase1.sh

```
#!/bin/bash

date

echo "Collect_MS3000G2_info"
ssh root@172.0.64.71 >/dev/null 2>&1 << eeooff
/root/profile_ms3000g2.sh start
exit
eeooff
echo "---------------"

echo "Collect_volume_info"
/v302/volume_list.sh start
echo "---------------"

echo "Init tests"
/v302/Supported/RHEL/7_1/spc1_optimized_v3.0.2  -run  SPC1_INIT    -iops  2500  -
   storage slave_asu.asu -output /v302/Output -master 2host.HST
echo "---------------"

echo "erify"
/v302/Supported/RHEL/7_1/spc1_optimized_v3.0.2  -run  SPC1_VERIFY    -iops  100  -
   storage slave_asu.asu -output /v302/Output
echo "---------------"

echo "Start metrics test"
/v302/Supported/RHEL/7_1/spc1_optimized_v3.0.2 -run  SPC1_METRICS -iops 430000 -
   storage slave_asu.asu -output /v302/Output -master 2host.HST
echo "metrics test over"
echo "---------------"
echo "start test verify"
/v302/Supported/RHEL/7_1/spc1_optimized_v3.0.2  -run  SPC1_VERIFY   -iops  100  -
   storage slave_asu.asu -output /v302/Output
echo "---------------"

#echo "start test persist1"
#/v302/Supported/RHEL/7_1/spc1_optimized_v3.0.2 -run SPC1_PERSIST_1  -iops 107500
   -storage slave_asu.asu -output /v302/Output -master 2host.HST
#echo "---------------"
echo "test_phase1 test over"

date
```

ms3000g2_test_persist1.sh

```
#!/bin/bash

date

echo "start test persist1"
/v302/Supported/RHEL/7_1/spc1_optimized_v3.0.2 -run SPC1_PERSIST_1  -iops 107500
   -storage slave_asu.asu -output /v302/Output -master 2host.HST
echo "---------------"
echo "test_persist1 test over"

date
```

ms3000g2_test_phase2.sh

```
#!/bin/bash

date

echo "Collect MS3000G2 info"
ssh root@172.0.64.71 >/dev/null 2>&1 << eeooff
/root/profile_ms3000g2.sh end
exit
eeooff
echo "---------------"

echo "Collect Volume info"
/v302/volume_list.sh end
echo "---------------"

echo "Start Persist2 phase"
/v302/Supported/RHEL/7_1/spc1_optimized_v3.0.2 -run SPC1_PERSIST_2  -iops 107500
   -storage slave_asu.asu -output /v302/Output -master 2host.HST
echo "---------------"
echo "The phase2 tests over"
date
```