

Reuters Market Data System

RMDS 6.0 (Solaris SPARC) Performance Test Results

DISCLAIMER: The test results and recommendations contained in this report are made available for informational purposes only. By issuing this report, Sun Microsystems, Inc. does not guarantee similar performance results. This report is provided for your internal use only and may not be redistributed or published in any form without the prior written consent of Sun Microsystems, Inc. All information contained herein is provided on an "AS-IS" BASIS WITHOUT WARRANTY OF ANY KIND. Obtaining repeatable, measurable performance results requires a controlled environment with specific hardware, software, network, and configuration in an isolated system. Adjusting any single element may yield different results. Additionally, test results at the component level may not be indicative of system level performance, or vice versa. Sun Microsystems, Inc. is pleased to provide this report for our customers. We appreciate that each organization has unique requirements, and therefore may find this information insufficient for its needs. Customers wishing to obtain custom analysis for their systems are encouraged to contact their local Sun Microsystems, Inc. representative.

Issue 1.0

Date of Issue: June 8, 2006

Contents

1 General Information	3
1.1 Objective.....	3
1.2 Testing Methodology.....	3
1.3 Software Versions.....	3
1.4 Hardware.....	3
2 Preparation for Performance Test.....	4
2.1 Network.....	4
2.2 Hardware.....	4
2.3 Operating System Configuration.....	4
2.4 RMDS Configuration.....	5
2.5 Miscellaneous Notes.....	5
3 Detailed Results.....	6
3.1 RSSL/RWF Update Throughput.....	6
3.2 Update Throughput via P2PS/POP.....	8
3.3 End-to-End RSSL/RWF Latency.....	9

© Sun Microsystems, Inc. 2006. All Rights Reserved.

Sun Microsystems, Inc. , by publishing this document, does not guarantee that any information contained herein is and will remain accurate or that use of the information will ensure correct and faultless operation of the relevant service or equipment. Sun Microsystems, Inc., its agents and employees, shall not be held liable to or through any user for any loss or damage whatsoever resulting from reliance on the information contained herein.

This document contains information proprietary to Sun Microsystems, Inc. and may not be reproduced, disclosed, or used in whole or part without the express written permission of Sun Microsystems, Inc.

Any Software, including but not limited to, the code, screen, structure, sequence, and organization thereof, and Documentation are protected by national copyright laws and international treaty provisions. This manual is subject to U.S. and other national export regulations.

1 General Information

1.1 Objective

The objective of this document is to report the performance test results for RMDS 6.0, for a particular hardware and software platform. The test procedures are described in Reuters RMDS 6.0 Performance Test Procedures and Results document.

The goal of these tests is to measure throughput and latency through RMDS 6.0 infrastructure components, specifically the Point-to-Pont Server (P2PS) and Source Distributor. The tests are grouped into three categories:

- Update throughput using RSSL/RWF data (see 3.1)
- Update throughput via P2PS/POP (see 3.2)
- End-to-end RSSL/RWF latency using embedded timestamp (see 3.3)

1.2 Testing Methodology

For throughput testing, the *sink_driven_src* utility was used to generate update traffic, and the *rmdstestclient* utility was used to consume the updates. Level 1 data was used, with a Marketfeed (MF) update size of 140 bytes, and an equivalent Reuters Wire Format (RWF) update size of 74 bytes. Tests with no fanout of updates used a 100,000 item watchlist. The infrastructure is tuned for maximum throughput, and the update rate was increased until the CPU limit was reached with no errors reported. Where needed, and as noted, multiple Source Distributors or multiple P2PSs were used to create the load necessary to measure the component under test.

The embedded timestamp approach was used to calculate end-to-end latency for Level 1 (Quotes and Trades) data. RMDS 6.0 end-to-end update latency is measured by using *sink_driven_src* as the publisher and *rmdstestclient* as the subscriber.

In the embedded timestamp approach, the publisher embeds timestamps into selected updates which the subscriber uses for latency calculations. In this scenario, the publisher and subscriber must be running on the same node for accurate timestamps.

1.3 Software Versions

1.3.1 RMDS

src_dist ver. mdh6.0.0.L4
p2ps ver. p2ps6.0.0.L4
rrcp as included in p2ps6.0.0.L4
rvd 7.4.19

1.3.2 RMDS Test Tool

sink_driven_src (from MDH load above)
rmdstestclient (from P2PS load above)

1.3.3 Operating Systems

- Solaris 10 1/06 (update 1) - SPARC

1.4 Hardware

- Sun V240 – 2 X 1.5GHz UltraSPARC IIIi processors with 2GB RAM

2 Preparation for Performance Test

2.1 Network

All the performance tests were run where the machines were connected to a private network via 1 Gbps switches. All the network cards and switch ports were set to Auto Negotiate.

2.2 Hardware

All RMDS components were run on the same class of machine.

2.3 Operating System Configuration

Earlier tests have shown that the value chosen for ticks per second (tps) on the test application machine has a significant impact on latency measurement. Accordingly, a tps value of 500 was used in these tests.

2.3.1 TCP and UDP Buffers

Any settings changed from the defaults are noted below:

Step	Procedure		
1	OS	Enter the following lines in system file noted	System File
		set shmsys:shminfo_shmmax=6000256	/etc/system
		set hires_tick=1	
		/usr/sbin/ndd -set /dev/tcp tcp_max_buf 4194304	/etc/rc2.d/S69in et
		/usr/sbin/ndd -set /dev/tcp tcp_recv_hiwat 1048576	
		/usr/sbin/ndd -set /dev/tcp tcp_xmit_hiwat 1048576	
		/usr/sbin/ndd -set /dev/udp udp_max_buf 4194304	
		/usr/sbin/ndd -set /dev/udp udp_xmit_hiwat 1048576	
/usr/sbin/ndd -set /dev/udp udp_recv_hiwat 1048576			

2.4 RMDS Configuration

The configuration templates—*rmds.cnf.template* and *pop.cnf.template*—were customized for the tests.

Config File	Description	Path
<i>rmds.cnf.template</i>	Configuration file	<i>\$RMDS_SW/config</i> <i>on mdh and p2ps/LAN systems</i>
<i>pop.cnf.template</i>	Configuration file to do P2PS/POP test	<i>\$RMDS_SW/config</i> <i>on p2ps/POP system only</i>

Any non-required changes (i.e., IP addresses, hostnames, etc). are noted below:

Config File	Parameter	Value
<i>rmds.cnf</i>	*RRCP*udpRecvBufSize	8192
<i>rmds.cnf</i>	*RRCP*udpSendBufSize	8192

2.5 Miscellaneous Notes

Any other significant deviations from the standard test procedures, or clarifications, are noted below (such as number/type of machines used, CPU binding policy, etc.):

Test	Deviation	Comments
src_dist/p2ps	CPU binding and interrupts	Except where noted, the primary component being tested (src_dist or p2ps) was bound to CPU 0. The respective transport daemon (rrcpd or rvd) was bound to CPU 1 in some cases. Interrupts were enabled on both CPUs.

3 Detailed Results

3.1 RSSL/RWF Update Throughput

- All the throughput numbers quoted here are for Level 1 data.
- The data file used in these tests has 1 update, with an update (data, not including header) size of 74 bytes in RWF.
- All of the tests with no fan-out used 100,000 item watchlist.
- In most of the throughput tests the individual processes were bound to particular CPU (s).
- ***sink_driven_src*** and ***rmdstestclient*** were used as the publisher and consumer of data.
- In some Source Distributor tests, two P2PSs were used to create sufficient load.

3.1.1 Standalone Source Distributor

Configuration Option	Transport	Max Throughput	Comments
Cache Disabled	RRCP	176,000	Single processor CPU utilization for src_dist was 98%.
Cache Enabled	RRCP	106,000	Single processor CPU utilization for src_dist was 98%.
Cache Disabled	Rendezvous	160,000	Single processor CPU utilization for src_dist was 100%.
Cache Enabled	Rendezvous	95,000	Single processor CPU utilization for src_dist was 100%.

3.1.2 P2PS/LAN

Configuration Option	Mounts : Commonality	Transport	Max Throughput	Comments
Cache Disabled	No fan-out	RRCP	135,000	The p2ps process was not bound to a CPU. The rrcpd process was bound to CPU 0.
Cache Enabled	No fan-out	RRCP	86,000	The p2ps process was not bound to a CPU. The rrcpd process was bound to CPU 0.
Cache Disabled	100 mounts; Producer 50/50	RRCP	8,000 input 404,000 output	The p2ps process was bound to CPU 1. The rrcpd process was bound to CPU 0.
Cache Disabled	No fan-out	Rendezvous	147,000	The p2ps process was not bound to a CPU. The rvd process was bound to CPU 1.
Cache Enabled	No fan-out	Rendezvous	90,000	The p2ps process was not bound to a CPU. The rvd process was bound to CPU 1.
Cache Disabled	100 mounts; Producer 50/50	Rendezvous	8,000 input 404,000 output	The p2ps process was bound to CPU 1. The rvd process was bound to CPU 0.

3.2 Update Throughput via P2PS/POP

These tests were performed using a P2PS/POP with data provided by an upstream P2PS/LAN, with an RRCP transport.

3.2.1 RSSL/RWF

Configuration Option	Mounts : Commonality	Max Throughput	Comments
Cache Enabled	No fan-out	95,000	The p2ps/POP process was bound to CPU 0. The throughput number was obtained with single processor CPU utilization of 96% for p2ps/POP.
Cache Enabled	100 mounts; Producer 50/50	7,400 input 373,700 output	The p2ps/POP process was bound to CPU 1. The throughput number was obtained with single processor CPU utilization of 98% for p2ps/POP.

3.3 End-to-End RSSL/RWF Latency

Latency is defined as the time for a data item to propagate through one or more RMDS components. “End to end” latency is defined as the delta between the time an update is posted by the publisher application to its API and the time the same update is received by the consuming application from its API, i.e. it includes both the latency contribution from the API and the core infrastructure components.

NOTES:

- Caching was disabled in both the Source Distributor and the P2PS during these tests.
- Optimized binaries of the RMDS infrastructure components were used.
- NTP was disabled on the tools node, as any drifts in time will affect the reported latency.
- Tests were run with 100,000 item watchlist and RWF data update size of 74 bytes [Data file (*sample.xml*) was used]. The update size is equivalent to a 140-byte IDN update.
- Latency tests were run at each update rate for at least 5 minutes, up to the maximum sustainable update rate for a given setup.
- Decode of data was turned on in these tests.

3.3.1 RRCP Backbone Results

Update Rate [74-byte RWF messages]	Mean Latency (millisec)	Std Deviation (millisec)	Maximum Latency (millisec)	Minimum Latency (millisec)	Number of Latency Points
1,000	0.514	0.126	4.054	0.323	3470
5,000	0.681	0.167	5.321	0.488	4010
10,000	0.869	0.178	5.001	0.682	3570
20,000	0.914	0.479	17.570	0.669	3580
30,000	1.058	0.243	5.776	0.695	3580
40,000	1.213	0.310	6.439	0.403	3760
50,000	1.275	0.333	6.209	0.753	3930
60,000	1.381	0.413	6.229	0.751	3580
70,000	1.485	0.459	6.670	0.718	3050
80,000	1.605	0.542	7.220	0.760	3490
90,000	1.688	0.716	11.786	0.703	3580
100,000	1.856	1.036	23.199	0.738	3740
110,000	1.977	0.915	8.296	0.778	3400
120,000	2.329	1.279	11.856	0.816	3510
130,000	2.881	1.566	11.236	0.842	4230
140,000	4.511	2.553	17.175	0.355	3880

3.3.2 Rendezvous Backbone Results

Update Rate [74-byte RWF messages]	Mean Latency (millisec)	Std Deviation (millisec)	Maximum Latency (millisec)	Minimum Latency (millisec)	Number of Latency Points
1,000	0.535	0.136	5.036	0.372	3890
5,000	0.715	0.414	11.033	0.541	3570
10,000	0.999	0.957	20.971	0.726	3580
20,000	1.241	1.170	17.650	0.777	3580
30,000	1.448	1.201	17.919	0.820	3620
40,000	1.608	1.278	15.384	0.810	3560
50,000	1.876	1.605	15.919	0.874	3580
60,000	2.223	2.046	16.520	0.835	3600
70,000	2.449	2.204	17.646	0.951	3570
80,000	3.110	3.313	32.223	0.971	3580
90,000	4.167	3.771	19.901	0.926	3561
100,000	7.302	5.165	36.541	1.416	3581