



Reuters Market Data System

RMDS 6.0 with InfiniBand on Solaris IPoIB Performance Test Results on Sun Fire™ X4150 Server with Cisco InfiniBand Infra

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 internal use only and can 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 might yield different results. Additionally, test results at the component level might 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: September 21, 2007

Amjad.Khan@sun.com

Contents

1 General Information	3
1.1 Objective.....	3
1.1.1 Results Summary.....	3
1.2 Testing Methodology.....	3
1.3 Software Versions.....	5
1.3.1 RMDS.....	5
1.3.2 RMDS Test Tool.....	5
1.3.3 Operating Systems.....	5
1.4 Hardware.....	5
2 Preparation for Performance Test.....	5
2.1 Network.....	5
2.2 Hardware.....	5
2.3 Operating System Configuration.....	5
2.3.1 Solaris™ Kernel Parameters.....	6
2.3.2 TCP and UDP Buffers.....	6
2.3.3 Additional Solaris Kernel and ibd driver Parameters for Network Latency.....	6
2.4 RMDS Configuration.....	6
2.5 Miscellaneous Notes.....	7
3 Detailed Results.....	9
3.1 RSSL/RWF Update Throughput.....	9
3.1.1 Standalone Source Distributor.....	9
3.1.2 P2PS/LAN.....	10
3.2 Update Throughput via P2PS/POP.....	11
3.2.1 RSSL/RWF.....	11
3.3 End-to-End RSSL/RWF Latency.....	12
3.3.1 RRCP Backbone Results.....	12
3.3.2 Rendezvous Backbone Results.....	13

© Sun Microsystems, Inc. 2007. Sun, Sun Microsystems, the Sun logo, Solaris, and Sun Fire are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries. Inc. AMD Opteron logo is a trademark or registered trademarks of Advanced Micro Devices. Intel Xeon is a trademark or registered trademark of Intel Corporation or its subsidiaries in the United States and other countries. Information subject to change without notice. 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 Reuters RMDS 6.0 with the **InfiniBand**(IB) technology, for a particular hardware and software platform. The Internet Protocol over InfiniBand (IPoIB) were used on the Solaris™ 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 with the InfiniBand(IB) network infrastructure using Solaris IPoIB, specifically the Point-to-Point 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)

1.1.1 Results Summary

RMDS Components Configuration	RMDS Throughput & End-to-End Latency Using Solaris IPoIB / Cisco IB Infrastructure
Maximum Throughput at <=1ms RMDS end-to-end Latency	400,000 updates/sec.
P2PS-LAN, RRCP, Producer 50/50 Fanout (Cache Disabled)	2,363,400 updates/sec.
P2PS-LAN, RVD, Producer 50/50 Fanout (Cache Disabled)	2,262,400 updates/sec.
P2PS-POP, Producer 50/50 Fanout (Cache Enabled)	2,232,200 updates/sec.
Src_dist, RRCP (Cache Disabled)	866,000 updates/sec.
Src_dist, RVD (Cache Disabled)	765,000 updates/sec.
P2PS-LAN, RRCP No Fanout (Cache Disabled)	850,000 updates/sec.
P2PS-LAN, RVD No Fanout (Cache Disabled)	755,000 updates/sec.
P2PS-POP, No Fanout (Cache Enabled)	430,000 updates/sec.

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, as illustrated in Figure 1. Level 1 data was used, with a 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.

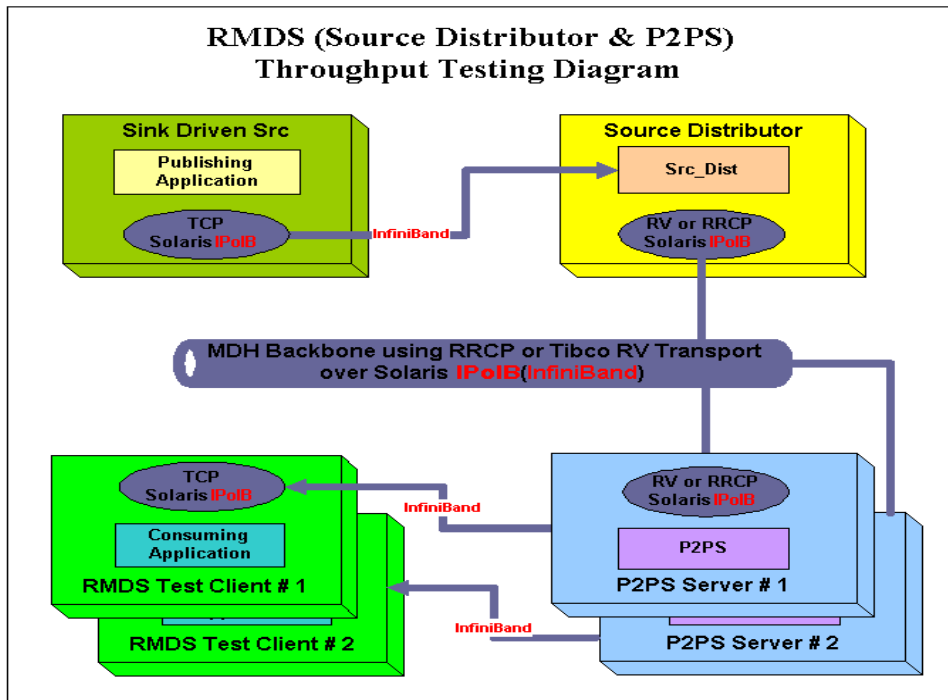


Figure 1: Throughput testing diagram

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, as shown in Figure 2.

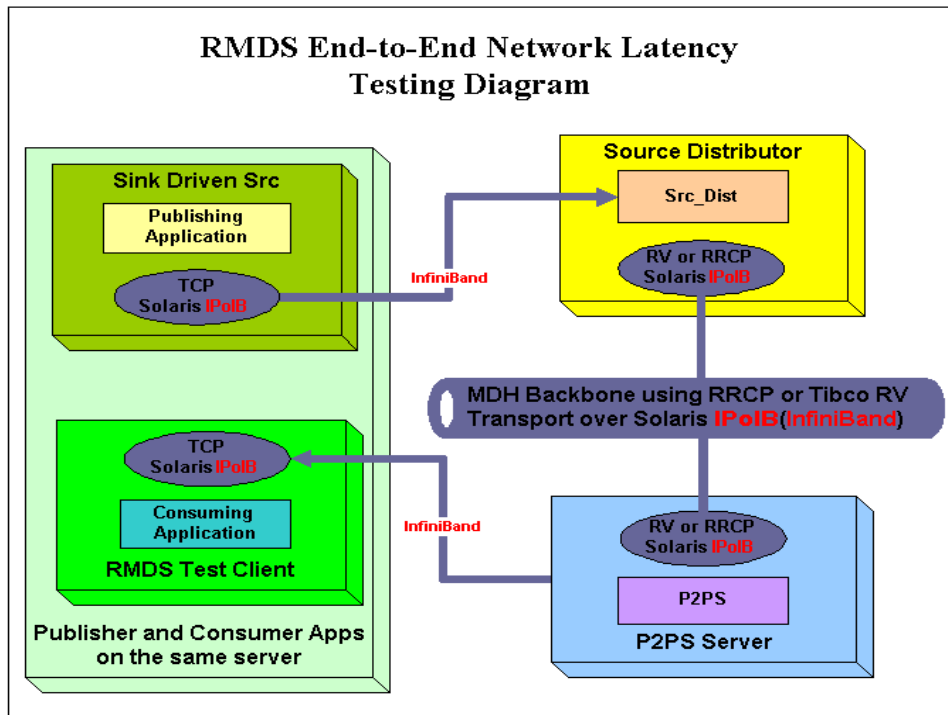


Figure 2: End-to-end latency testing diagram

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. mdh6.0.2.L2
p2ps ver. p2ps6.0.2.L2
rrcp as included in p2ps6.0.2.L2
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 11/06 (update 3)
- Patch level 118855-36

1.4 Hardware

- Sun's next generation two-socket 1U rack-mount Sun Fire X4150 server powered by Dual-Core Intel® Xeon® 5100series — 2 X 3.0 GHz Dual-Core Intel Xeon 5160 processors (total four cores)
- Cisco HCA 400EX X8 PCI-E installed on the system
- Cisco SFS-7000D InfiniBand Server Switch using system version SFS-7000D TopspinOS 2.9.0 releng #170 06/26/2007 16:40:42

2 Preparation for Performance Test

2.1 Network

All the performance tests were run where the machines were connected to a private network via Cisco SFS-7000D InfiniBand Server Switch. Two partitions (VLANs) were created on the IB switch to handle UDP and TCP traffic on each. For testing purposes the **ibd** interfaces were used with the IPoIB protocol.

2.2 Hardware

All RMDS components were run on the same class of machine. Some test support systems in the test bed had 3.0 GHz AMD Opteron™ processors. On all of the systems, the Cisco HCA (ibd) interfaces were used for MDH backbone and TCP network traffic.

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 Solaris Kernel Parameters

Any settings changed from the defaults are noted below:

Step	Procedure		
1	OS	Enter the following lines in system file noted	System File
	Solaris	set shmsys:shminfo_shmmax=6000256 set hires_tick=1 (Only on the sink_driven_src and rmdstestclient systems)	/etc/system

2.3.2 TCP and UDP Buffers

Any settings changed from the defaults are noted below:

Step	Procedure		
2	OS	Enter the following lines in system file noted	System File
	Solaris	/usr/sbin/ndd -set /dev/tcp tcp_max_buf 8388608	/lib/svc/met hod/net-init
		/usr/sbin/ndd -set /dev/tcp tcp_recv_hiwat 2097152	
		/usr/sbin/ndd -set /dev/tcp tcp_xmit_hiwat 2097152	
		/usr/sbin/ndd -set /dev/udp udp_max_buf 8388608	
		/usr/sbin/ndd -set /dev/udp udp_xmit_hiwat 1048576	
		/usr/sbin/ndd -set /dev/udp udp_recv_hiwat 1048576	
/usr/sbin/ndd -set /dev/udp udp_do_checksum 0			

2.3.3 Additional Solaris Kernel and ibd driver Parameters for Network Latency

Any settings changed from the defaults are noted below:

Step	Procedure		
3	OS	Enter the following lines in the file noted	System File
	Solaris	set autoup=300	/etc/system
	ibd driver	set dld:dld_opt=2 (for throughput and latency) set ibd:ibd_srv_fifos=0	

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

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
<i>rmds.cnf</i>	*RDFD*readBias	60 (add this entry)
<i>rmds.cnf</i>	*p2ps*maxOutputBuffers	1000
<i>rmds.cnf</i>	*p2ps*guaranteedOutputBuffers	800

Values modified for **network latency** testing are noted below:

Config File	Parameter	Value
<i>rmds.cnf</i>	*IDN_RDF*rrmpFlushInterval	0
<i>rmds.cnf</i>	*RDFD*rrmpFlushInterval	0
<i>rmds.cnf</i>	*p2ps*tcpNoDelay	True
<i>rmds.cnf</i>	*p2ps*timedWrites	False
<i>rmds.cnf</i>	*p2ps*flushInterval	0

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	CPU binding and interrupts disabling and CPU scheduling class assignment	<p>Except where noted, src_dist was bound to CPU 1 and the respective transport daemon (rrcpd or rvd) threads were bound to CPU 2 & 3. Interrupts were disabled on CPU 1, 2, & 3 (no-intr using psradm).</p> <p>For RRCP Transport: Commands Used for process binging: <pre>% psrset -c -F 2 3 % psrset -c -F 1 % psrset -f 1 2 % psrset -e 1 ./rrcpd source % ./src_dist % psrset -b 2 `pgrep src_dist`/1 % pbind -b 2 `pgrep rrcpd`/7 % pbind -b 3 `pgrep rrcpd`/12</pre> FX scheduling class was assigned to src_dist and rrcpd or rvd daemon processes: <pre>% priocntl -s -c FX -m 60 -p 60 `pgrep src_dist` % priocntl -s -c FX -m 60 -p 60 `pgrep rrcpd`</pre> </p> <p>For Tibco RV Transport: Commands Used for process binging: <pre>% psrset -c -F 3 % psrset -c -F 1 % psrset -f 1 2 % ./src_dist % psrset -b 1 `pgrep src_dist`/1 % psrset -b 2 `pgrep rvd`/3</pre> FX scheduling class was assigned to src_dist and rrcpd or rvd daemon processes: <pre>% priocntl -s -c FX -m 60 -p 60 `pgrep src_dist` % priocntl -s -c FX -m 60 -p 60 `pgrep rvd`</pre> </p>

Test	Deviation	Comments
p2ps (LAN/POP)	<p>CPU binding and interrupts disabling and CPU scheduling class assignment</p> <p>For P2PS-POP simply create single processor set out of CPU 2 & 3 and follow the same step mentioned for P2PS-LAN except that there is no transport involved in this testing.</p>	<p>Except where noted, src_dist was bound to CPU 1 and the respective transport daemon (rrcpd or rvd) threads were bound to CPU 2 & 3. Interrupts were disabled on CPU 1, 2, & 3 (no-intr using psradm).</p> <p>For RRCP Transport: Commands Used for process binging: <pre>% psrset -c -F 2 3 % psrset -c -F 1 % psrset -f 1 2 % ./rrcpd sink % psrset -e 1 ./p2ps % pbind -b 2 `pgrep p2ps`/4 % pbind -b 3 `pgrep p2ps`/1-3,5 % pbind -b 1 `pgrep rrcpd`/11 % pbind -b 0 `pgrep rrcpd`/8 % psrset -b 1 `pgrep rrcpd`/9 % pbind -b 3 `pgrep rrcpd`/9</pre> </p> <p>FX scheduling class was assigned to src_dist and rrcpd or rvd daemon processes: <pre>% priocntl -s -c FX -m 60 -p 60 `pgrep p2ps` % priocntl -s -c FX -m 60 -p 60 `pgrep rrcpd`</pre> </p> <p>For Tibco RV Transport: Commands Used for process binging: <pre>% psrset -c -F 2 3 % psrset -c -F 1 % psrset -f 1 2 % psrset -e 1 ./p2ps % pbind -b 2 `pgrep p2ps`/7 % pbind -b 3 `pgrep p2ps`/1-6,8 % psrset -b 2 `pgrep rvd`/14 % pbind -b 0 `pgrep rvd`/3 % pbind -b 1 `pgrep rvd`/13</pre> </p> <p>FX scheduling class was assigned to src_dist and rrcpd or rvd daemon processes: <pre>% priocntl -s -c FX -m 60 -p 60 `pgrep src_dist` % priocntl -s -c FX -m 60 -p 60 `pgrep rvd`</pre> </p>
End-to-End Latency	CPU binding and interrupts disabling and CPU scheduling class assignment	Please see the section 2.3.4 and above notes for setting up the src_dist and P2PS for network latency.

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	800,000 (8 percent CPU still idle)	See section 2.5 for performance tuning details for src_dist with RRCP transport. NOTE: Two systems were used for p2ps in the test bed. Single processor CPU utilization for thread 1 of src_dist was 97 percent.
Cache Enabled	RRCP	385,000	See section 2.5 for performance tuning details for src_dist with RRCP transport.
Cache Disabled	Rendezvous	765,000 (RVD runs out of CPU)	See section 2.5 for performance tuning details for src_dist with Tibco RV transport. NOTE: Two systems were used for p2ps in the test bed. Single processor CPU utilization for thread 1 of src_dist was only 90%.
Cache Enabled	Rendezvous	387,000	See section 2.5 for performance tuning details for src_dist with Tibco RV transport. FX scheduling class was assigned to src_dist and rvd daemon processes.

3.1.2 P2PS/LAN

Configuration Option	Mounts : Commonality	Transport	Max Throughput	Comments
Cache Disabled	No fan-out	RRCP	760,000	See section 2.5 for performance tuning details for P2PS with RRCP transport.
Cache Enabled	No fan-out	RRCP	415,000	See section 2.5 for performance tuning details for P2PS with RRCP transport.
Cache Disabled	100 mounts; Producer 50/50	RRCP	46,900 input 2,363,400	See section 2.5 for performance tuning details for P2PS with RRCP transport.
Cache Disabled	No fan-out	Rendezvous	755,000	See section 2.5 for performance tuning details for P2PS with Tibco RV transport.
Cache Enabled	No fan-out	Rendezvous	402,000	See section 2.5 for performance tuning details for P2PS with Tibco RV transport.
Cache Disabled	100 mounts; Producer 50/50	Rendezvous	44,800 input 2,262,400 output	See section 2.5 for performance tuning details for P2PS with Tibco RV transport. The throughput number was obtained with single processor CPU utilization of 100 percent for p2ps.

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	430,000	See section 2.5 for performance tuning details for p2ps/POP without any transport. The throughput number was obtained with single processor CPU utilization of 98 percent for p2ps/POP.
Cache Enabled	100 mounts; Producer 50/50	44,200 input 2,232,200 output	See section 2.5 for performance tuning details for p2ps/POP without any transport. The throughput number was obtained with single processor CPU utilization of 100 percent 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].
- 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 RRCF Backbone Results

Update Rate [74-byte RWF messages]	Mean Latency (millisec.)	Std Deviation (millisec.)	Maximum Latency (millisec.)	Minimum Latency (millisec.)	Number of Latency Points
1000	0.247	0.011	0.329	0.238	3170
5000	0.293	0.008	0.366	0.284	3240
10,000	0.353	0.014	0.483	0.321	3100
20,000	0.405	0.020	0.771	0.370	3175
30,000	0.454	0.040	1.083	0.365	3160
40,000	0.498	0.093	1.253	0.370	3220
50,000	0.509	0.074	1.385	0.357	3090
60,000	0.509	0.083	1.477	0.354	2970
70,000	0.543	0.091	1.654	0.373	3130
80,000	0.550	0.098	1.664	0.385	3100
90,000	0.558	0.098	1.852	0.377	3120
100,000	0.548	0.105	1.688	0.376	3120
150,000	0.631	0.117	1.457	0.355	3120
200,000	0.685	0.157	2.007	0.369	3100
250,000	0.756	0.202	2.238	0.363	3130
300,000	0.845	0.257	2.606	0.392	3110
350,000	0.948	0.340	3.352	0.407	3130
400,000	1.080	0.458	4.051	0.408	3110
450,000	1.237	0.583	4.746	0.425	3135
500,000	1.522	0.887	6.271	0.436	3140

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
1000	0.272	0.117	4.033	0.255	3200
5000	0.312	0.096	4.062	0.291	3130
10,000	0.391	0.225	6.176	0.342	3160
20,000	0.456	0.321	6.176	0.364	3160
30,000	0.493	0.343	7.587	0.370	3120
40,000	0.579	0.508	7.702	0.392	3157
50,000	0.614	0.591	8.121	0.381	3150
60,000	0.611	0.504	7.659	0.385	3120
70,000	0.636	0.555	7.675	0.380	3170
80,000	0.654	0.586	8.003	0.383	3120
90,000	0.735	0.796	8.416	0.390	3140
100,000	0.720	0.703	7.918	0.384	3140
150,000	0.857	0.879	8.012	0.384	3180
200,000	1.058	1.053	8.252	0.389	3130
250,000	1.295	1.362	8.740	0.393	3200
300,000	1.561	1.479	8.785	0.452	3130
350,000	1.837	1.775	9.063	0.393	3240
400,000	2.389	2.124	10.169	0.405	3140
450,000	2.807	2.303	11.567	0.404	3147
500,000	3.947	2.420	12.704	0.414	3140