Sun StorEdge™ Availability Suite 3.2 Remote Mirror Software Configuration Guide

Sun Microsystems, Inc.
www.sun.com

Part No. 817-3753-10
December 2003, Revision 51

Submit comments about this document at: `http://www.sun.com/hwdocs/feedback`

Contents

Preface

The *Sun StorEdge™ Availability Suite 3.2 Software Configuration Guide* provides information for the efficient set-up and use of the software.

Using UNIX Commands

This document might not contain information on basic UNIX® commands and procedures such as shutting down the system, booting the system, and configuring devices. See the following for this information:

- Software documentation that you received with your system
- Solaris™ operating environment documentation, which is at

  `http://docs.sun.com`

Shell Prompts

Shell	Prompt
C shell	*machine-name*%
C shell superuser	*machine-name*#
Bourne shell and Korn shell	`$`
Bourne shell and Korn shell superuser	#

Typographic Conventions

Typeface[*]	Meaning	Examples
`AaBbCc123`	The names of commands, files, and directories; on-screen computer output	Edit your `.login` file. Use `ls -a` to list all files. `% You have mail`.

`AaBbCc123`	What you type, when contrasted with on-screen computer output	`%` **`su`**
		`Password:`

AaBbCc123	Book titles, new words or terms, words to be emphasized. Replace command-line variables with real names or values.	Read Chapter 6 in the *User's Guide*. These are called *class* options. You *must* be superuser to do this. To delete a file, type `rm` *filename*.

* The settings on your browser might differ from these settings.

Related Documentation

Application	Title	Part Number

Man pages	`sndradm`
	`iiadm`
	`dsstat`
	`kstat`
	`svadm`
		N/A

Latest release information	*Sun StorEdge Availability Suite 3.2 Software Release Notes*	817-2782
	Sun Cluster 3.0 and Sun StorEdge Software Release Note Supplement	816-5128
Installation and User	*Sun StorEdge Availability Suite 3.2 Software Installation Guide*	817-2783
System administration	*Sun StorEdge Availability Suite 3.2 Point-In-Time Copy Software Administration and Operations Guide*	817-2781
	Sun StorEdge Availability Suite 3.2 Remote Mirror Software Administration and Operations Guide	817-2784

Accessing Sun Documentation

You can view, print, or purchase a broad selection of Sun documentation, including localized versions, at:

`http://www.sun.com/documentation`

Contacting Sun Technical Support

If you have technical questions about this product that are not answered in this document, go to:

`http://www.sun.com/service/contacting`

Sun Welcomes Your Comments

Sun is interested in improving its documentation and welcomes your comments and suggestions. You can submit your comments by going to:

`http://www.sun.com/hwdocs/feedback`

Please include the title and part number of your document with your feedback:

Sun StorEdge Availability Suite 3.2 Remote Mirror Software Configuration Guide, part number 817-3753-10

Configuring the Remote Mirror Software

The Sun StorEdge™ Availability Suite 3.2 Remote Mirror software is a volume-level *replication* facility for the Solaris™ 8 and 9 (Update 3 and higher) Operating Systems. The Remote Mirror software replicates disk volume write operations between physically separate *primary* and *secondary* sites in real time. The Remote Mirror software can be used with any Sun™ network adapter and network link that supports TCP/IP.

Since the software is volume-based, it is storage-independent and supports raw volumes or any volume manager, for both Sun and third-party products. Additionally, the product supports any application or database that has a single host running the Solaris system that writes data. Databases, applications or file systems that are configured to allow multiple hosts running the Solaris system to write data to a shared volume are not supported (for example: Oracle 9iRAC, Oracle Parallel Server).

As part of a disaster recovery and business continuance plan, the Remote Mirror software keeps up-to-date copies of critical data at remote sites. The Remote Mirror software enables you to rehearse and test business continuance plans. For a highly available solution, the Sun StorEdge Availability Suite software can be configured to failover within Sun Cluster 3.x environments.

The Remote Mirror software is active while your applications are accessing the data volumes, replicating the data continually to the remote sites or scoreboarding changes, which allow for a fast resynchronization at a later time.

The Remote Mirror software enables you to initiate resynchronization manually from either the primary site to the secondary site (typically called *forward synchronization*), or from the secondary site to the primary site (typically called *reverse synchronization*).

Replication and configuration in the Remote Mirror software is done on a set basis. A remote mirror set consists of a primary volume, a secondary volume, a bitmap volume on both the primary and secondary sites (used to track and scoreboard changes for fast resynchronization), and an optional *asynchronous queue* volume for *asynchronous replication* mode. The primary and secondary volumes are recommended to be the same size. You can use the `dsbitmap` tool to determine the required size of the bitmap volumes. For more information on configuring remote mirror sets or the `dsbitmap` tool see the *Sun StorEdge Availability Suite 3.2 Remote Mirror Software Administration and Operations Guide*.

Theory of Operation

Replication can occur either synchronously or asynchronously. In synchronous mode, an application write operation is not acknowledged until the write operation is committed on both the primary and secondary hosts. In asynchronous mode, the application write operation is acknowledged when it is committed to storage locally and written to an asynchronous queue. This queue drives write operations to the secondary site asynchronously.

Synchronous Replication

The data flow for synchronous operation is as follows:

1. Scoreboard bit is set in the bitmap volume.
2. Local write operation and network write operation are initiated in parallel
3. When both write operations are complete, the scoreboard bit is cleared (*lazy clear*).
4. Write operation is acknowledged to application.

The advantage of *synchronous replication* is that both primary and secondary sites are always in sync. This type of replication is practical only if the latency of the link is low and the bandwidth requirements of the application can be met by the link. These constraints usually confine a synchronous solution to a campus or metropolitan location.

In this case, the average service time for a write operatin is:

bitmap write + MAX (local data write, network round trip + remote data write)

In the campus and metropolitan location, the network round trip is negligible and the service time is approximately twice what is observed when the Remote Mirror software is not installed.

Assuming 5 milliseconds for a write, then:

5ms + MAX (5ms, 1ms + 5ms) = 11ms

Note – This value of 5 milliseconds is a reasonable assumption on a lightly loaded system. On a more realistically loaded system, queuing backlog increases the value.

However, if the network round trip is approximately 50 milliseconds (typical for long distance replication), the network latency renders the synchronous solution impractical as shown in the following example:

5ms + MAX (5ms, 50ms + 5ms) = 60ms

Asynchronous Replication

Asynchronous replication separates the remote write operation from the application write operation. In this mode, the acknowledgment occurs when the network write operation is added to the asynchronous queue. This means that the secondary site can get out of synchronization with the primary site until all write operations are delivered to the secondary site. In this mode, data flows in this manner:

1. Scoreboard bit is set.
2. Local write, asynchronous queue write operations are done in parallel.
3. Write is acknowledged to application.
4. Flusher threads read asynchronous queue entry and perform network write.
5. Scoreboard bit is cleared (lazy clear).

The service time is the time required for the following:

bitmap write + MAX (local write, asynchronous queue entry data)

Using the value of 5 milliseconds service time for a write operation, the estimated service time for an asynchronous write operation is:

5ms + MAX (5ms, 5ms) = 10ms

If the network drain rate for the volume or *consistency group* is exceeded by the write rate for an extended period of time, the asynchronous queue fills up. Proper sizing is important so a method for estimating the appropriate volume size is discussed later in this document.

There are two modes that govern how the Remote Mirror software behaves in the event of the asynchronous disk queue filling;

- *Blocking*

  In blocking mode, which is the default setting, the Remote Mirror software blocks, and waits for the asynchronous disk queue to drain to a certain point before adding the write to the asynchronous queue. This impacts application write operations, but maintains write ordering across the link.

- Non-blocking mode

  In non-blocking mode (not available with memory-based queues), the Remote Mirror software does not block when the disk asynchronous queue fills, but drops into *logging* mode and scoreboards the write. On a subsequent *update synchronization* these are read from bit 0 forward, and there is no preservation of write ordering. If this mode is used and if the asynchronous disk queue fills and write ordering is lost, the associated volume or consistency group is inconsistent. It is strongly advised that a point-in-time copy be taken on the secondary site prior to starting the update synchronization, for example, using the autosync daemon.

Consistency Groups

In synchronous mode, write ordering for an application that spans many volumes is assured because the application waits for completion before issuing another I/O operatin when ordering is required, and the Remote Mirror software does not signal completion until the write operatin is on both primary and secondary sites.

In asynchronous mode, by default, the queue for each volume is drained by one or more independent threads. Because this operation is separated from the application, write ordering is not preserved across write operations to multiple volumes.

If write ordering is required for an application, the Remote Mirror software provides the consistency group feature. Each consistency group has a single network queue, and although multiple write operations are allowed in parallel, write ordering is preserved through the use of sequence numbers.

Planning for Remote Replication

When you are planning for remote replication, consider your business needs, the application write loads, and your network's characteristics.

Business Needs

When you decide to replicate your business data, consider the maximum delay: How long out of date can you allow the data on the secondary site to become? This determines the replication mode and snapshot scheduling. Additionally, it is very important to know if the applications that you are replicating require the write operations to the secondary volume to be replicated in the correct order.

Application Write Load

Understanding the average and peak write loads is critical to determining the type of network connection required between the primary and secondary sites. To make decisions about the configuration, collect the following information:

- The average rate and size of data write operations

  The average rate is the amount of data write operations while the application is under typical load. Application read operations are not important to the provisioning and planning of your remote replication.

- The peak rate and size of data write operations

  The peak rate is the largest amount of data written by the application over a measured duration.

- The duration and frequency of the peak write rate

  The duration is how long the peak write rate lasts and the frequency is how often this condition occurs.

If these application characteristics are not known, you can measure them using tools such as `iostat` or `sar` to measure write traffic while the application is running.

Network Characteristics

When you know the application write load, determine the requirements of the network link. The most important network properties to consider are the network bandwidth and the network latency between the primary and secondary sites. If the network link already exists prior to installing the Sun StorEdge Availability Suite software, you can use tools such as `ping` to help determine the characteristics of the link between the sites.

To use synchronous replication, the network latency must be low enough that your application response time is not affected dramatically by the time of the network round trip of each write operation. Also, the bandwidth of the network must be sufficient to handle the amount of write traffic generated during the application's peak write period. If the network cannot handle the write traffic at any time, the application response time will be impacted.

To use asynchronous replication, the bandwidth of the network link must be able to handle the write traffic generated during the application's average write period. During the application peak write phase, the excess write operations are written to the local asynchronous queue and then written to the secondary site at a later time when the network traffic allows. The application response time can be minimized during bursts of write traffic above the network limit as long as the asynchronous queue is properly sized.

See the "Configuring the Asynchronous Queue" on page 6 section of this document. The remote mirror asynchronous option mode selected (blocking or non-blocking) determines how the software reacts to the queue filling.

Configuring the Asynchronous Queue

If you use asynchronous replication, plan for the configuration settings described in this section. These settings are set on a remote mirror set or consistency group basis.

Disk or Memory Queue

In version 3.2 of the software, the Remote Mirror software added support for disk-based asynchronous queues. For ease of upgrading from previous versions, memory-based queues are still supported, but the new disk-based queues provide the ability to create significantly larger, more efficient queues. Larger queues allow for larger bursts of write activity without affecting application response time. Also, disk-based queues have less impact to system resources than the memory-based queues.

The asynchronous queue must be sufficient in size to handle the bursts of write traffic associated with the application peak write periods. A large queue can handle prolonged bursts of write activity, but also allows the possibility that the secondary site gets further out of sync with the primary. Using the peak write rate, peak write duration, write size, and network link characteristics, you can determine how the queue should be sized. See “Setting the Correct Size for Disk-based Asynchronous Queue” on page 12.

The queue option you select (blocking or non-blocking) determines how the software reacts to a filled disk queue. Use the `dsstat` tool to determine statistics for the asynchronous queue, including the high-water mark (`hwm`), which shows the largest amount of the queue that has been used. To add an asynchronous queue to a remote mirror set or consistency group, use the `sndradm` command with the `-q` option: `sndradm -q a`

Queue Size

Monitor the asynchronous queue using the `dsstat`(1SCM) command to check the *high water mark* (`hwm`). If the `hwm`, caused by the application writes more data than the queue can handle, reaches 80 to 85 percent of the total size of the queue frequently, increase the queue size. This principle applies both to disk-based and memory-based queues. However, the procedure to resize each queue type is different.

Memory-Based Queue

- The default maximum number of write operations in the queue (tunable) is 4096. Use the `sndradm -W` command to change this value.
- The default maximum number of 512-byte data blocks (default queue size) (tunable) is 16384, which is about 8 Mbytes of data. Use the `sndradm -F` command to change this value.

Disk-Based Queue

The effective size of the disk queue is the size of the disk queue volume. A disk queue can only be resized by replacing it with a volume of a different size. For example, for a queue size of 16384 blocks, check that the `hwm` does not exceed 13000 to 14000 blocks. If it exceeds this amount, resize the queue using the following procedure.

Note – The maximum size of a disk queue is one block fewer than 1 terabyte, or 2147483647 blocks. Do not use a volume larger than the maximum size.

To Resize a Queue

1. **Place the volume into logging mode (using the** `sndradm -l` **command).**
2. **Resize the queue.**
   - Memory-based: use the `sndradm -F` command.
   - Disk-based: replace the existing disk queue volume with a volume of larger size using the `sndradm -q` command.
3. **Perform an update synchronization by using the** `sndradm -u` **command.**

To Display the Current Queue Size, Length, and `hwm`

1. **Type the following to display the queue size:**
   - Memory-based:

```
# sndradm -P
/dev/vx/rdsk/data_t3_dg/vol0 -> priv-2-
230:/dev/vx/rdsk/data_t3_dg/vol0
autosync: off, max q writes: 4096, max q fbas: 16384, async
threads: 8, mode: async, state: replicating
```

The size of the queue in blocks is given by `max q fbas` (16384 blocks in this example). The maximum number of items allowed in the queue is given by `max q writes` (4096 in this example). In this example, this means that the average size of an item in the queue is 2K.

   - Disk-based:

```
# sndradm -P
/dev/vx/rdsk/data_t3_dg/vol0 -> priv-
230:/dev/vx/rdsk/data_t3_dg/vol0
autosync: off, max q writes: 4096, max q fbas: 16384, async
threads: 1, mode: async, blocking diskqueue:
/dev/vx/rdsk/data_t3_dg/dq_single, state: replicating
```

The diskqueue volume is displayed (`/dev/vx/rdsk/data_t3_dg/dq_single`). The size of the queue can be determined by examining the size of the volume.

2. **Type the following to show the current queue length and its** `hwm`**:**

```
# dsstat -m sndr -d q
name               q role    qi     qk  qhwi   qhwk
data_a5k_dg/vol0   D  net     4     13     5    118
```

Where:

- `qi` is the current number of items in the queue
- `qk` is the current total data size in the queue (in Kbytes)
- `qhwi` is the maximum number of items that have ever been in the queue at any one time
- `qhwk` is the maximum data in Kbytes that have ever been in the queue at any one time

3. **To show streaming summary and disk queue information, type:**

```
# dsstat -m sndr -r bn -d sq 2
```

4. **To show more information, run** `dsstat`**(1SCM) with other display options.**

Sample `dsstat` *Output For a Correctly-Sized Queue*

Note – This example shows only a portion of the command output required for this section; the `dsstat` command actually displays more information.

The following `dsstat`(1SCM) kernel statistics output shows information about the asynchronous queue. In these examples, the queue is sized correctly and is not currently filled. This example shows the following settings and statistics:

Disk Based Example

```
# dsstat -m sndr -r n -d sq -s \ priv-2-230:/dev/vx/rdsk/data_t3_dg/vol67
name              q role    qi     qk  qhwi   qhwk    kps   tps  svt
data_t3_dg/vol67  D  net    48    384   240   1944     10     1   54
```

Where:

- The qi entry means that a total of 48 write transactions have been put into the queue
- The qk entry means that 384 Kbytes have been put into the queue
- The qhwi entry shows that the hwm for queued items is 240 items; not currently being reached
- The qhwk entry shows that the hwm for queued data (Kbytes) is 1944; not currently being reached

Assuming the disk queue volume size is 1 Gbyte, or 2097152 disk blocks, the hwm of 1944 blocks is well below 80% full. The disk queue is sized correctly for the write load.

Sample `dsstat` *Output for an Incorrectly-Sized Disk Queue*

The following `dsstat`(1SCM) kernel statistics output shows information about the asynchronous queue, which is incorrectly sized.

Memory Based Example

```
# sndradm -P
/dev/vx/rdsk/data_a5k_dg/vol0 -> priv-230:/dev/vx/rdsk/data_a5k_dg/vol0
autosync: off, max q writes: 4096, max q fbas: 16384, async threads: 2, mode:
async, state: replicating

# dsstat -m sndr -d sq
name              q role    qi     qk  qhwi   qhwk    kps   tps  svt
data_a5k_dg/vol0  M  net  3609   8060  3613   8184     87    34   57
k/bitmap_dg/vol0     bmp     -      -     -      -      0     0    0
```

This example shows the default queue settings but the application is writing more data than the queue can handle. The `qhwk` value of 8184 Kbytes compared to `max q fbas` of 16384 blocks (8192 Kbytes) indicates that the application is approaching the maximum allowed limit of 512-byte blocks. It is possible that the next few I/O operations are not going to be placed into the queue.

Increasing the queue size would be a solution in this case. However, consider improving the network link (such as using larger bandwidth interfaces) to achieve long-term benefits. Alternatively, consider taking point-in-time volume copies and replicating the shadow volumes. See the *Sun StorEdge Availability Suite 3.2 Point-in-time Copy Software Administration and Operations Guide*.

Summary

- If the fill rate is less than or equal to the drain rate, the default queue size is sufficient.
- If drain rate is less than the fill rate, increasing the queue size might provide a temporary solution. However, if the write operations continue for a prolonged period, the queue eventually fills.

Setting the Correct Size for Disk-based Asynchronous Queue

Consider the following example. In this example, `iostat` was run at an hourly interval to profile the I/O load that will be replicated. In this example, we assume a DS3 (45Mb/S) link. Also assume that this application uses a single consistency group, consequently, a single queue is involved.

After collecting stats over a 24 hour period, and assuming that this is a typical day for the application in question, one may determine the average write rate, the proper sizing for the async queues, how far out of date the remote site may become over the course of the day, and whether the network bandwidth chosen is adequate for this application.

time	kwr/s	wr/s	network throughput	queue growth	queue size
	A	B	C	A/1000 - C)*3600	
6am	0	0	4MB/S		
7am	1000	400	4MB/S		
8am	2000	1000	4MB/S		
9am	2000	1000	4MB/S		
10am	4000	1800	4MB/S		
11am	5000	2400	4MB/S	3.6GB	3.6GB
12pm	1000	400	4MB/S	-10GB	
1pm	1200	600	4MB/S		
2pm	1000	500	4MB/S		
3pm	1200	400	4MB/S		
4pm	2000	600	4MB/S		
5pm	1000		4MB/S		
6pm	800		4MB/S		
7pm	800		4MB/S		
8pm	3200	1000	4MB/S		
9pm	8000	2500	4MB/S	14GB	14GB
10pm	8000	2500	4MB/S	14GB	28GB
11pm	1000	400	4MB/S	-10	18
12pm	0		4MB/S	-14	4

time	kwr/s	wr/s	network throughput	queue growth	queue size
1am	0		4MB/S	-14	
2am	0		4MB/S		
3am	0		4MB/S		
4am	0		4MB/S		
5am	0		4MB/S		
Average bandwidth	1.8MB/S				

After filling in the table and calculating queue growth and size, it is evident that a 30gb queue is sufficient. Although the queue grows large and, consequently, the secondary grows out of sync, a batch job run in the evenings ensures that the queue is empty by normal business hours and the two sites are in sync.

This exercise also validates that the network bandwidth is adequate for the write load the application produces.

Configuring Asynchronous Queue Flusher Threads

The Sun StorEdge Availability Suite 3.2 software provides the ability to set the number of threads flushing the asynchronous queue. Changing this number allows for multiple I/Os per volume or consistency group on the network at one time. The Remote Mirror software on the secondary node handles write ordering the I/Os using sequence numbers.

Many variables must be considered when determining the number of queue flusher threads that is most efficient for your replication configuration. These variables include the number of sets or consistency groups, available system resources, network characteristics, and whether or not there is a file system. If you have a small number of sets or consistency groups, a larger number of flusher threads might be more efficient. It is recommended that you do some basic testing or prototyping with this variable at slightly different values to determine the most efficient setting for your configuration.

Knowledge of the configuration, network characteristics, and operation of the Remote Mirror software can provide guidelines to proper selection of the number of network threads. The Remote Mirror software utilizes Solaris RPCs as a transport mechanism: These RPCs are synchronous. For each network thread, the maximum throughput the individual thread can achieve is I/O size * Round trip time. Consider a workload that is predominately 2k I/Os, and a round trip time of 60 msecs. Each network thread would be capable of:

2k/0.060s = 33k/s

In the case that there is a single volume, or many volumes in a single consistency group, one can see that the default of 2 network threads would limit network replication to 66k/S. Tuning this number up would be advisable. If the replication network were provisioned for 4MB/S, then theoretically, the optimal number of network threads for a 2k workload would be:

(4096K/s) / (2K/0.060 IO/s) = 123 threads

This assumes linear scalability. In practice it has been observed that adding more than 64 network threads yields no benefit. Consider the case where there is no consistency group, 30 volumes being replicated over a 4MB/S link, and 8k I/Os. The default of 2 network threads per volume would yield 60 network threads, and if the workload were spread evenly across these volumes, the theoretical bandwidth would be:

60 * (8K / 0.060 IO/s) = 8MB/s

This is more than the network bandwidth. No tuning is required.

The default setting for the number of asynchronous queue flusher threads is 2. To change this setting, you would use the `sndradm` CLI with the `-A` option. The description for the `-A` option is: `sndradm -A` specifies the maximum number of threads that can be created to process the asynchronous queue when a set is replicating in asynchronous mode (default 2).

To determine the number of flusher threads that are currently configured to serve an asynchronous queue, you can use the `sndradm -P` command. For example, you can see that the set below has 2 asynchronous flusher threads configured.

```
# sndradm -P
/dev/md/rdsk/d52 -> lh1:/dev/md/sdsdg/rdsk/d102
autosync: off, max q writes: 4096, max q fbas: 16384, async threads: 2, mode:
async, group: butch, blocking diskqueue: /dev/md/rdsk/d100, state: replicating
```

An example of how to use the `sndradm -A` option to change the number of asynchronous queue flusher threads to 3 is:

```
# sndradm -A 3 lh1:/dev/md/sdsdg/rdsk/d102
```

Network Tuning

The Remote Mirror software injects itself directly into the system's I/O path, monitoring all traffic to determine if it is targeted to remote mirror volumes. The I/O commands that are targeted for remote mirror volumes are tracked and replication of these write operations is managed. Due to the fact that the Remote Mirror software is directly in the system's I/O path, some performance impact to the system is expected. Additional TCP/IP processing that is required for network replication also consumes host CPU resources.

Perform the procedures in this section on the primary and secondary remote mirror hosts.

TCP Buffer Size

The *TCP buffer* size is the number of bytes that the transfer control protocol allows to be transferred before it waits for an acknowledgment. To get maximum throughput, it is critical to use optimal TCP send and receive socket buffer sizes for the link you are using. If the buffers are too small, the TCP congestion window will never fully open. If the receiver buffers are too large, TCP flow control breaks and the sender can overrun the receiver, causing the TCP window to shut down. This event is likely to happen if the sending host is faster than the receiving host. Overly large windows on the sending side are not a problem as long as you have excess memory.

Note – Increasing the buffer size to a much higher value over a shared network might impact the network performance. See the Solaris System Administrator Collection for information about tuning the size.

TABLE 1 shows the maximum possible throughput for a 100BASE-T network..

TABLE 1 Network Throughput and Buffer Size

Latency	Buffer Size = 24 Kbytes	Buffer Size = 256 Kbytes
10 milliseconds	18.75 Mb per second	100 Mb per second
20 milliseconds	9.38 Mb per second	100 Mb per second
50 milliseconds	3.75 Mb per second	40 Mb per second
100 milliseconds	1.88 Mb per second	20 Mb per second
200 milliseconds	0.94 Mb per second	10 Mb per second

Viewing and Tuning TCP Buffer Sizes

You can view and tune your TCP buffer size by using `/usr/bin/netstat`(1M) and `/usr/sbin/ndd`(1M) commands. TCP parameters to consider tuning include:

- `tcp_max_buf`
- `tcp_cwnd_max`
- `tcp_xmit_hiwat`
- `tcp_recv_hiwat`

When you change one of these parameters, restart the Remote Mirror software with the `shutdown` command, allowing the software to use the new buffer size. However, after you shut down and restart your server, the TCP buffers return to a default size. To keep your change, set the values in a startup script as described later in this section.

Network Tuning To View TCP Buffers and Values

▼ To View all TCP buffers

● **Type the following:**

```
# /usr/sbin/ndd /dev/tcp ? | more
```

▼ To View settings by buffer name

● **This command shows a value of 1073741824.**

```
# /usr/sbin/ndd /dev/tcp tcp_max_buf
1073741824
```

▼ To View Buffer Sizes for a Socket

- **Use the** /usr/bin/netstat**(1M) command to view the buffer size for a particular network socket.**

  For example, view the size for port 121, the default remote mirror port:

```
# netstat -na |grep “121 “
*.121 *.* 0 0 262144 0 LISTEN
192.168.112.2.1009 192.168.111.2.121 263536 0 263536 0 ESTABLISHED
192.168.112.2.121 192.168.111.2.1008 263536 0 263536 0 ESTABLISHED

# netstat -na |grep rdc
*.rdc *.* 0 0 262144 0 LISTEN
ip229.1009 ip230.rdc 263536 0 263536 0 ESTABLISHED
ip229.rdc ip230.ufsd 263536 0 263536 0 ESTABLISHED
```

The value 263536 shown in this example is the 256 Kbyte buffer size. It must be set identically in the primary and secondary hosts.

▼ To Set and Verify the Buffer Size in a Startup Script

Note – Create this script on the primary and secondary hosts.

---

1. **Create the script file in a text editor using the following values:**

```
#!/bin/sh
ndd -set /dev/tcp tcp_max_buf 16777216
ndd -set /dev/tcp tcp_cwnd_max 16777216

# increase DEFAULT tcp window size
ndd -set /dev/tcp tcp_xmit_hiwat 262144
ndd -set /dev/tcp tcp_recv_hiwat 262144
```

2. **Save the file as** /etc/rc2.d/S68ndd **and exit the file.**

3. **Set the permissions and ownership to the** /etc/rc2.d/S68ndd **file.**

```
# /usr/bin/chmod 744 /etc/rc2.d/S68ndd
# /usr/bin/chown root /etc/rc2.d/S68ndd
```

4. **Shut down and restart your server.**

```
# /usr/sbin/shutdown -y g0 -i6
```

5. **Verify the size as shown previously.**

Remote Mirror's Use of TCP/IP ports

The Remote Mirror software on both the primary and secondary nodes listens on a well-known port specified in `/etc/services`, port 121. Remote mirror write traffic flows from primary to secondary site over a socket with an arbitrarily assigned address on the primary site and the well known address on the secondary site. The health monitoring heartbeat travels over a different connection, with an arbitrarily assigned address on the secondary and the well know address on the primary. The remote mirror protocol utilizes SUN RPCs over these connections.

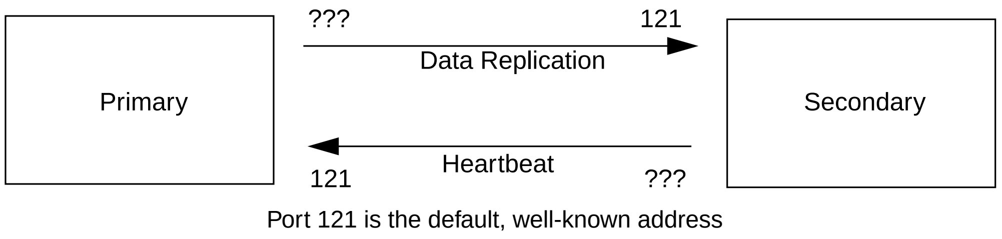

FIGURE 1 Remote Mirror's Use of TCP Port Addresses

Default TCP Listening Port

Port 121 is the default TCP port for use by the remote mirror `sndrd` daemon. To change the port number, edit the `/etc/services` file using a text editor. See the *Sun StorEdge Availability Suite 3.2 Software Installation Guide* for more information.

If you change the port number, you must change it on all remote mirror hosts within this configuration set (that is, primary and secondary hosts, all hosts in one-to-many, many-to-one, and multihop configurations). In addition, you must shutdown and restart all hosts affected, so that the port number change can take effect.

Using Remote Mirror With a Firewall

Because RPCs require an acknowledgment, the *firewall* must be opened to allow the well-known port address to be in either the source or destination fields of the packet. If the option is available, be sure to configure the firewall to allow RPC traffic as well.

In the case of write replication traffic, packets destined for the secondary will have the well-known port number in the destination field, acknowledgments of these RPCs will contain the well-known address in source field.

For health monitoring, the heartbeat will originate from the secondary with the well-known address in the destination field, and the acknowledgment will contain this address in the source field.

Remote Mirror Software with Point-in-Time Copy Software

To help ensure the highest level of data integrity and system performance on both sites during normal operations, the Sun StorEdge Availability Suite Point-in-Time Copy software is recommended for use in conjunction with Remote Mirror software.

A point-in-time copy can be replicated to a physically remote location, providing a consistent copy of the volume as part of an overall disaster recovery plan. This is commonly referred to as batch replication, and the process and advantages of this practice are described in the best practice guide: *Sun StorEdge Availability Suite Software - Improving Data Replication over a Highly Latent Link.*

The point-in-time copy of a remote mirror secondary volume can be established prior to starting synchronization of a secondary volume from the primary site (the site the primary volume is hosted from). Protection against double failure is provided by enabling the Point-in-Time Copy software to create a point-in-time copy of the replicated data at the secondary site before beginning resynchronization. If a subsequent failure occurs during resynchronization, the point-in-time copy can be used as a fallback position, and resynchronization can be resumed when the subsequent failure issues have been resolved. Once the secondary site is fully synchronized with the primary site, the Point-in-Time Copy software volume set can be disabled, or put to other uses, such as remote backup, remote data analysis, or other functions required at the secondary site.

The Point-in-Time Copy software I/O performed internally during an enable, copy, or update operation can alter the contents of the shadow volume without any new I/O coming down the I/O stack. When this happens, the I/O is not intercepted in the SV layer. If the shadow volume is also a remote mirror volume, the Remote Mirror software will not see these I/O operations either. In this situation, the data modified by the I/O will not be replicated to the target remote mirror volume.

To allow this replication to occur, the Point-in-Time Copy software can be configured to offer the Remote Mirror software the changed bitmap. If the Remote Mirror software is in logging mode, it accepts the bitmap and performs an OR comparison of the Point-in-Time Copy software bitmap with its own bitmap for that volume, adding the Point-in-Time Copy software changes to its own list of changes to be replicated to the remote node. If the Remote Mirror software is in replication mode for the volume, it rejects the bitmap from the Point-in-Time Copy software. This, in turn, will fail the enable, copy, or update operation. Once remote mirror logging has been re-enabled, the Point-in-Time Copy software operation can be reissued.

Note – A remote mirror volume set must be in logging mode for the Point-in-Time Copy software to successfully perform an enable, copy, update, or reset operation on a remote mirror volume. If not, the point-in-time copy operation fails and the Remote Mirror software reports that the operation is denied.

Remote Replication Configurations

The Remote Mirror software enables you to create one-to-many, many-to-one, and multihop volume sets.

- One-to-many replication enables you to replicate data from one primary volume to many secondary volumes residing on one or more hosts. One primary and each secondary site volume is a single volume set. For example, with one primary and three secondary host volumes, you need to configure three volume sets: primary A and secondary B1, primary A and secondary B2, and primary A and secondary B3.
- Many-to-one replication enables you to replicate volumes across more than two hosts through more than one network connection. The software supports the replication of volumes located on many different hosts to volumes on a single host. The terminology differs from the one-to-many configuration terminology, where the one and the many referred to are volumes.
- Multihop replication indicates that the secondary host volume of one volume set acts as the primary host volume of another volume set. In the case of one primary host volume A and one secondary host volume B, the secondary host volume B appears as primary host volume A1 to the secondary host volume B1.

Any combination of the above configurations is also supported by the Remote Mirror software.

Glossary

asynchronous queue	A local area of disk or memory used to store writes that are to be replicated to a remote site. After the writes have been put into the queue, the write is acknowledged to the application, and the writes are forwarded to the remote site at a later time, as the network capabilities permit.
asynchronous replication	A synchronous replication confirms to the originating host that the primary I/O transaction is complete before updating the remote image. That is, completion of the I/O transaction is acknowledged to the host when the local write operation is finished and the remote write operation has been queued. Deferring the secondary copy removes the long distance propagation delays from the I/O response time.
auto synchronization	With the auto synchronization option enabled on the primary host, the synchronization daemon (autosyncd) attempts to resynchronize volume sets if the system reboots or a link failure occurs.
blocking	(asynchronous queue) In blocking mode, if the asynchronous queue fills, all future writes are delayed until the queue drains enough to allow for a write to occur. Blocking mode, which is the default Asynchronous running option, ensures write ordering of the packets to the secondary site. If the asynchronous queue fills with the blocking option set, response time to the application may be impacted.
configuration location	Location where the Sun StorEdge Availability Suite software stores configuration information about all enabled volumes used by the software.
consistency group	A consistency group is a group of remote volumes that share a single asynchronous queue to maintain w rite ordering.
dsstat	A tool from the Sun StorEdge Availability Suite tool that can be used to display kernel statistics from the remote mirror and point-in-time snapshot products.

firewall	A computer that acts as an interface between two networks and regulates traffic between those networks for the purpose of protecting the internal network from electronic attacks originating from the external network.
forward resynchronization	See Update synchronization.
full synchronization	Full synchronization performs a complete volume-to-volume copy, which is the most time-consuming of the synchronization operations. In most cases, a secondary volume is synchronized from its source primary volume. However, restoration of a failed primary disk might require reverse synchronization, using the surviving remote mirror as the source.
hwm	See High water mark
high water mark	The high water mark is the largest amount of the asynchronous queue that has been used.
lazy clear	
logging	Mode where a bitmap tracks writes to a disk, rather than a running log of each I/O event. This method tracks disk updates that have not been remotely copied while the remote service is interrupted or impaired. The blocks that no longer match their remote sets are identified for each source volume. The software uses this log to re-establish a remote mirror through an optimized update synchronization rather than a complete volume-to-volume copy.
non-blocking	(asynchronous queue) In non-blocking mode, if the asynchronous queue fills, the Remote Mirror software goes into scoreboarding mode and the contents of the queue are discarded. Non-blocking mode, does not ensure write ordering of the packets to the secondary site, but it assures that response time to the application will not be impacted if the asynchronous queue fills.
primary or local: host or volume	The system or volume on which the host application is principally dependent. For example, this is where the production database is being accessed. This data is to be replicated to the secondary by the software.
replication	Once a volume set has been initially synchronized, the software ensures that the primary and secondary volumes contain the same data on an ongoing basis. Replication is driven by user-layer application write operations; replication is an ongoing process.
reverse synchronization	An operation used during recovery rehearsals. Logging keeps track of test updates applied to the secondary system during the rehearsal. When the primary is restored, the test updates are overwritten with the blocks from the primary image, restoring matching remote sets.

secondary or remote: host or volume	The remote counterpart of the primary, where data copies are written to and read from. Remote copies are transmitted without host intervention between peer servers. A server might act as primary storage for some volumes and secondary (remote) storage for others.
synchronization	The process of establishing an identical copy of a source disk onto a target disk as a precondition to the software mirroring.
synchronous replication	Synchronous replication is limited to short distances (tens of kilometers) because of the detrimental effect of propagation delay on I/O response times.
TCP buffer	The TCP buffer size is the number of bytes that the transfer control protocol allows to be transferred before it waits for an acknowledgment.
update synchronization	Update synchronization copies only those disk blocks identified by logging, reducing the time to restore remotely mirrored sets.
volume set file	A text file containing information about specific volume sets. This text file is not the same as the configuration location, which contains information about all configured volume sets used by the remote mirror and Point-in-Time Copy software.