Domino clusters (Part 1)

by
Harry Murray
and Gary Sullivan

Level: Advanced
Works with: Domino 5.0
Updated: 08/02/99

Inside this article:

Related links:

Workload balancing with Domino clusters
Fine points of configuring a cluster
NotesBench Web site
Domino Performance Zone

Get the PDF:

[Editor's Note: This is the first article in a two-part series on the performance analysis of Domino clusters. This article introduces you to clusters, and then looks at our performance tests of R4.6 clusters. It also includes recommendations from our performance testing on R4.6. The second article focuses on performance tests of R5 clusters, including data on the Internet Cluster Manager and cluster replication.]

Introduction

As a Domino administrator, your top concern is ensuring 24x7 server availability to your user community. At the same time, you need Domino to scale well, and to continue offering fast response times as the needs and numbers of your users grows. You can address both of these concerns by creating a Domino cluster.

This is the first article in a two-part series that examines the performance benefits of using Domino clusters. This article first introduces you to Domino clusters, focusing on those aspects that relate to performance. Then, it looks at our performance testing of Domino R4.6 clusters. The next article will look at our performance testing of Domino R5 clusters, including data on the Internet Cluster Manager and cluster replication.

As you'll see in this article, our performance tests of Domino R4.6 clusters involved various system configurations. In these tests, we examined different evaluation scenarios and saw how each configuration measured up under a particular workload. These evaluation scenarios helped us draw conclusions about the following aspects of performance:

- How clustering affects the load on a server
- How adding multiple Cluster Replicators affects performance
- How to improve performance on an existing cluster

The test data comes from a variety of groups, including IBM's Distributed Systems Performance Analysis department in Austin, Texas; the Iris Performance group; and the IBM Lotus Integration Center (ILIC), which supports all Lotus products in-the-field, including Domino clustering. We hope that you'll be able to use some of our conclusions to put clustering to work for you.

What is Domino clustering?

A Domino cluster links multiple Domino servers together so that they appear as one resource from the client perspective. The cluster functions as a “single” provider of resources, enabling client requests to be processed in a timely manner. If any given server is unavailable or too busy at the time the request arrives, the cluster transparently passes the request to a server capable of handling the work. The cluster members can be on a mixture of the supported Domino platforms, including Windows NT and various UNIX systems, IBM AS/400, MVS, or OS/2. The clusters support Notes clients as well as Web browser clients.

Domino clustering is accomplished entirely at the application level. No special hardware is needed. With clustering, multiple copies of databases on multiple servers provide high availability. In addition, Domino distributes the workload between the cluster members (called *workload balancing*), allowing for lower

overall response times and more consistency in response times during peak intervals.

In a Domino cluster, if one member of the cluster fails, another member of the cluster transparently assumes the failing member's workload. This action is called *failover.* Domino servers provide failover to clients by redirecting requests to another server in the cluster that has a replica of the database needed to service the request. (For information on when cluster failover occurs, see the **Domino Administration Help**.) Redirection is a function of the Cluster Manager. The Cluster Manager tracks cluster membership and the status of all clustered servers. Individual cluster members may be located in the same room, or in locations around the world.

Domino clusters replicate database changes as the changes occur to all replica copies of the database. This synchronization of cluster components is key to Domino's high availability. This style of replication is referred to as *event-driven* (immediate) replication, in contrast to standard replication that occurs on a schedule. Event-driven replication is a function of the Cluster Replicator.

Other clustering solutions, such as Microsoft Clusters (Wolfpack), provide failover of databases to other cluster members using only a single instance of the database. The two cluster members share the same RAID set. If the database is inaccessible because the disk drive or RAID set is down, failover cannot occur. In Microsoft Clusters, the database fails over only at the hardware level. Plus, because Microsoft Clusters only have a single copy of the database, you cannot distribute databases geographically for "hot site" failover.

About workload balancing
Domino clusters provide workload balancing by redistributing user requests to an overloaded server to other servers in the cluster that have available capacity. To optimize the workload balancing, you can use the following three techniques:

1. Make sure that you distribute databases evenly in the cluster.
When a server in the cluster fails or becomes overloaded, user requests automatically redirect to other servers in the cluster. Ideally, this load should be spread equally across all other servers in the cluster. However, this can only happen when replicas of the databases on the failed server are spread roughly equally across the other servers in the cluster. For an example of how to distribute databases in a cluster, see "**Workload balancing with Domino clusters**."

Note that if you distribute the databases evenly across the servers, you're assuming that the databases have about the same activity. If you have some power users or particularly active databases, you may need to fine tune the distribution of those databases to make the activity on each server approximately equal.

As the test data shows later in this article, distributing databases evenly is a key aspect of effective workload balancing. For more information on balancing databases among cluster members, see the second article in this series (coming soon).

2. Set the threshold for when the server is considered Busy.
Each server in a cluster periodically determines its own workload, based on the average response time of requests recently processed by the server. The *server availability index* indicates how busy the server is. The index is a value between 0 and 100, where 100 indicates a lightly loaded server (fast response times), and 0 is a heavily loaded server (slow response times). With the NOTES.INI setting SERVER_AVAILABILITY_THRESHOLD, you can specify a threshold that determines the lowest value of the server's availability index

for which the server is not considered "Busy." When the server's availability index goes below the threshold value, the server is in the Busy state. A server in the Busy state redirects users to another server in the cluster.

The server's availability index is derived from the ratio between the current response time and the response time in optimum conditions (with no Domino transactions). Note that the response times that are taken into account are server-based and do not include any consideration for network time. The Cluster Manager process on each server monitors the average response time of a set of server operations over roughly the last 75 seconds.

Domino uses the NOTES.INI setting SERVER_TRANSINFO_NORMALIZE when calculating the server availability index to "normalize" the response times observed at the server (that is, it divides the observed response times by this normalize value). Until now, this setting was undocumented, but it is available in both R4.6 and R5.

For the availability index calculation to work properly, the normalize value should be roughly equal to the average Domino transaction time (for the server in question) in milliseconds*100. The default value is 3000ms, corresponding to an average response time of 30ms per transaction. This default setting was appropriate for "the average server" when clustering was first shipped several years ago, but it is too large for the current generation of servers. You should use a lower normalize value with today's faster servers, so loads failover correctly.

Our testing on Windows NT shows that you can coordinate the threshold and normalize settings to achieve even load balancing among cluster members. That is, you can cause failover to another server when a server is too "busy" or is unavailable. For more information on specifying these NOTES.INI settings, see the second article in this series (coming soon).

3. Set the maximum number of users for a server.
With the NOTES.INI setting SERVER_MAXUSERS, you can specify the maximum number of users allowed to access the server concurrently. When the server reaches this limit, it rejects requests for additional sessions. So, users failover to another server in the cluster. This setting is not specific to clustering, but it is useful for redirecting users when a cluster member is in trouble.

Setting up your cluster topology
When setting up your cluster, you should consider the benefits of using a private LAN for intra-cluster communication. This way, you can offload the cluster's probe and replication network traffic from the LAN, leaving more bandwidth for client communication with the cluster servers. You can also eliminate the network as a single point of failure in your cluster. For more information on using a private LAN for intra-cluster communication, see "**Fine points of configuring a cluster**."

Setting up multiple Cluster Replicators
In addition, you should consider setting up multiple Cluster Replicators. When a server is added to a cluster, Domino loads the Cluster Replicator (CLREPL) and adds it to the ServerTasks= line in the NOTES.INI file. This way, the Cluster Replicator automatically loads whenever you restart the server. This should be sufficient for most cluster configurations. However, in some cases, a single Cluster Replicator may not be able to keep up with the replication workload. (When a database is replicated, all transactions that update the original cause an update in each replica.)

If you run multiple Cluster Replicators, you can split up the replication workload and process it in parallel. This capability is similar to the support in the (standard) Domino replicator for running multiple replicator tasks. You can also specify multiple instances of CLREPL on the ServerTasks= line, which causes the specified number of Cluster Replicators to load at server

startup.

To determine if you might benefit from an additional Cluster Replicator task, you should monitor the Replica.Cluster.WorkQueueDepth statistic (in the Replica Statistics report). This statistic shows the current number of modified databases awaiting cluster replication. If this value is consistently greater than zero, you may need to enable more Cluster Replicators. For more information on cluster statistics, see "**Fine points of configuring a cluster**."

For more information on deciding when to add additional Cluster Replicator tasks, see the test data later in this article. Also, check out the R5 test data in the second article in this series (coming soon).

Test methodology for Domino R4.6 clusters

This section outlines the overall test methodology that we used for our Domino R4.6 cluster test scenarios. It includes information about the system configurations, the workloads, and our evaluation scenarios.

Please note that the configurations we used in our performance tests are not necessarily meant to be recommended configurations. The primary reason that we chose these particular configurations was that they were the easiest way to measure the resource utilization of the various cluster components. In addition, note that these are very low-end "servers" -- we only used them to show the relative changes of various resources. The limiting resource in the configuration was the disk resources, which you will see in the data described later in this article.

Also, remember that Domino clusters are extremely flexible. For example, some sites can create a cluster of Domino servers that span across multiple operating system partitions on the same hardware server. Our test example is just one permutation of Domino clusters. In fact, the second part of this article will show R5 data on mid-range clustered servers.

System configurations

To run the scenarios, we used the following configurations:

Servers

- CPUs: Dell PowerEdge 2200 with one Pentium II/333MHz processor
- Memory: 512MB RAM
- 523MB page file
- Two SCSI controllers and two disks
- Network: 100Mbit Ethernet (private)
- OS: Windows NT Server 4.0
- Domino: Release 4.6.2

Client

- CPUs: Dell Dimension with one Pentium II/400MHz processor
- Memory: 256MB RAM
- 267MB page file
- One disk
- OS: Windows NT Workstation 4.0
- Notes: Release 4.6a
- Mail workload from the IBM Center of Competency

About the workloads

In our first three test scenarios, we set up the Notes client running an R4 mail workload from the IBM Center of Competency called the “IBM Geoplex site” workload. This workload uses standard Notes mail -- that is, mail transferred using the Notes Remote Procedure Call (RPC) protocol, *not* the Internet protocols. The workload simulates three types of mail users: light, medium, and heavy. In each iteration of the 15-minute script, users:

- Send mail
- Navigate through their mail databases

- Refile, delete, and update mail documents

The workload is modeled after a very active IBM site and is “heavily update bound" (that is, it causes lots of disk writes). The updates are significant because they cause cluster replications. The IBM Geoplex site workload is approximately 3.5 times as heavy as the NotesBench R4 mail workload.

The following table shows more details for the IBM Geoplex site workload for the light, medium, and heavy user types:

	Unit	Light	Medium	Heavy
Client throughput (per 100 users)	# APIs/min	245	337	850
Server throughput (w/o cluster) (per 100 users)	# trans/min	488	500	901
Mail Throughput (per 100 users)	# messages/min	11	46	240
Average mail size	bytes	1,000	1,000	6,888
# of Geoplex users simulated	per user thread	2	1	1

We used the **NotesBench** R4 mail workload in Scenario 4. The workload uses a *nthiteration* setting to determine how often the simulated users send a 1K message. When nthiteration=1, the maximum number of mail messages are sent during the time period -- about 32 times to three recipients. In each iteration of the script, users:

- Read about 160 mail documents per day
- Update about 64 mail documents per day
- Delete about 64 mail documents per day

For more information on the R4 mail workload, see the **NotesBench Web site**.

About the evaluation scenarios
As mentioned earlier, the evaluation scenarios helped us draw conclusions about the following aspects of performance:

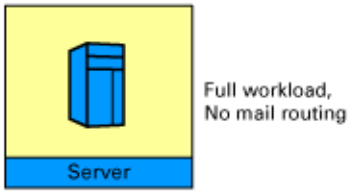
- How clustering affects the load on a server
- How adding multiple Cluster Replicators affects performance
- How to improve performance on an existing cluster

Each scenario focused on cluster replication, because virtually all the overhead of clustering is due to cluster replication. Our performance tests have shown that other cluster processes (the Cluster Database Directory Manager task, cluster probing, and so on) add very little overhead.

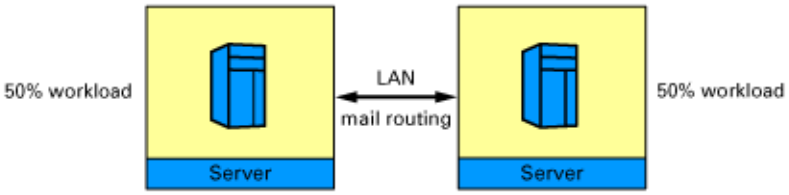
Scenario 1: Cluster replication with mail workloads

This scenario tests the performance impact of cluster replication during light, medium, and heavy mail usage (using the IBM Geoplex site workload). We first measured the CPU utilization for 100 users. We then ran the workload with varying numbers of users and measured the client response time, probe response time, CPU utilization, disk write times, and disk utilization.

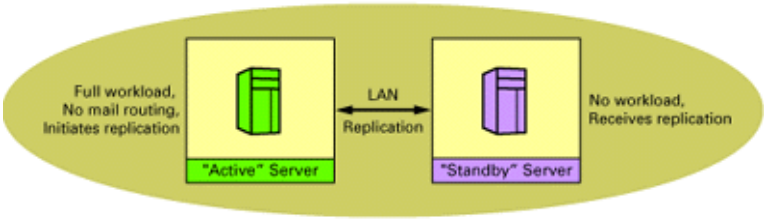
Domino configurations
We set up the Domino servers in four clustered/non-clustered configurations. In the first configuration (**Config 1**), we used a single server with no cluster:



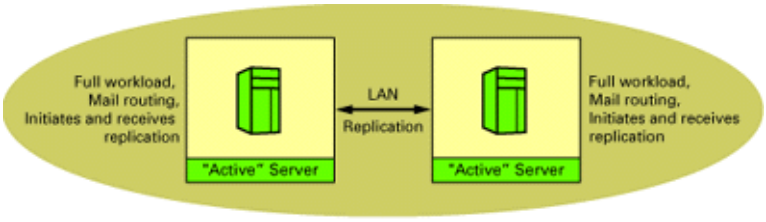
In the next configuration (**Config 2**), we equally divided users between two servers, again with no cluster. Each server transfers all mail messages to the other server.



The third configuration (**Config 3**) used two servers in a cluster. All users were on one server, which we call the *active server*. The other server is a *standby* member because its only load is cluster replication. The active server does *not* transfer mail messages to the standby server. Rather, all mail is addressed to users on the active server, so all messages are delivered locally. The active server replicates all databases to the standby server. (Note: Although you do not usually use this configuration in production systems, this is the preferred testing configuration because you can measure the replication loads separately on both the system pushing the cluster replications and the receiving system. Once you know what each load is separately, it's easier to predict what will happen if you add a load to the second server.)



The final configuration (**Config 4**) again used two servers in a cluster, but this time, they were both *active* servers. Users were on both servers. Users on the first server sent mail to users on the second server, and vice versa, so both servers transferred messages to the other. In addition, both servers used cluster replication for the databases.



About the tests

Our first tests used the workload with 100 users running on the first three Domino clustered/non-clustered configurations. Then, we ran the workload with varying numbers of users on the final Domino cluster configuration. The 100-user tests helped us establish a baseline for predicting resource usage in the varying-number-of-users test. To see the results of these tests, see the sidebar "**Cluster replication test results**."

Note: We used 100 users as a baseline because our systems were limited by disk I/O. At higher numbers, the disk I/O restrictions caused our results to go nonlinear. We used more appropriately-sized systems in our R5 clustering tests, which are covered in the second article in this series.

In general, our test results show that for a workload with heavy updates, the increase in CPU and disk usage is significant. In addition, the response time increases as the number of users increases. (The response time is probably the most important performance metric because it measures how responsive the server appears to users. If users often experience response times of more than one second, the server is generally considered to be too busy.) For a complete list of conclusions, see the "**Recommendations from our R4.6 clustering tests**" section later in this article.

Scenario 2: Multiple Cluster Replicators

In this scenario, we measured the CPU and disk utilization when varying the number of Cluster Replicator tasks. This way, we can determine how adding multiple replicators affects performance. The **Domino Administration Help** recommends that you use the same number of Cluster Replicators as the number of cluster members that you replicate to. In Scenario 3 (and in the second part of this article), you will learn more about the actual benefits of using multiple Cluster Replicators.

Domino configurations

To run this scenario, we used the same clustered configuration as in Config 4 above, except that we used multiple standby servers as follows:

- Three servers in a cluster -- one active and two standby
- Five servers in a cluster -- one active and four standby
- Six servers in a cluster -- one active and five standby

Notice that in each configuration, there is only one active server with a workload on it. The active server then replicates to the rest of the standby servers. (For each database on the active server, there is a replica of that particular database on each standby server.)

About the tests

Our tests used the "medium-type" mail user workload with 100 users running on the three clustered configurations. We varied the number of Cluster Replicator tasks from 1 to *N*, where *N* is the number of standby servers. To see the results of these tests, see the sidebar "**Multiple Cluster Replicator test results**."

In general, our test results show that the CPU and disk utilization increase with multiple replicators. This means that if you are already fully utilizing the CPU and disk resources, you should not use multiple Cluster Replicators. For a complete list of conclusions, see the "**Recommendations from our R4.6 clustering tests**" section later in this article.

Scenario 3: Concurrence in updating replicas

As a follow-up to Scenario 2, we again measured the CPU utilization when varying the number of Cluster Replicator tasks. We used the same type of clustered configuration with five servers -- one active and four standby. Again, only the active server had the workload on it. The active server then replicated to the rest of the standby servers.

Our test again used the "medium-type" mail user workload, except that this time, we sent a single message to 50 users on the active server. This modification caused a heavy "pulse" of load on the server, which made it easier to measure the time it took for resultant replications. We varied the number of Cluster Replicator tasks from one to four, where four is the number of standby servers. To see the results of this test, see the sidebar "

Concurrence in updating replicas results."

In general, our test results show that increasing the number of Cluster Replicator tasks may increase the concurrency in propagating updates to replicas. Multiple tasks running on a server can shorten the lag time between when updates are made to the databases on that server, and when changes are propagated to the replicas on the multiple standby servers. However, as stated before, since additional Cluster Replicator tasks may increase the CPU and disk load, do not increase them if the server is already overloaded. For a complete list of conclusions, see the "**Recommendations from our R4.6 clustering tests**" section later in this article.

Note: You can use the Replica.Cluster.SecondsOnQueue.Avg statistic to get a good indication of how quickly the Cluster Replicator propagates changes to other servers. You can use this statistic to determine if an additional Cluster Replicator task reduces the time to update replicas on the other cluster members. For more information on cluster statistics, see "**Fine points of configuring a cluster**."

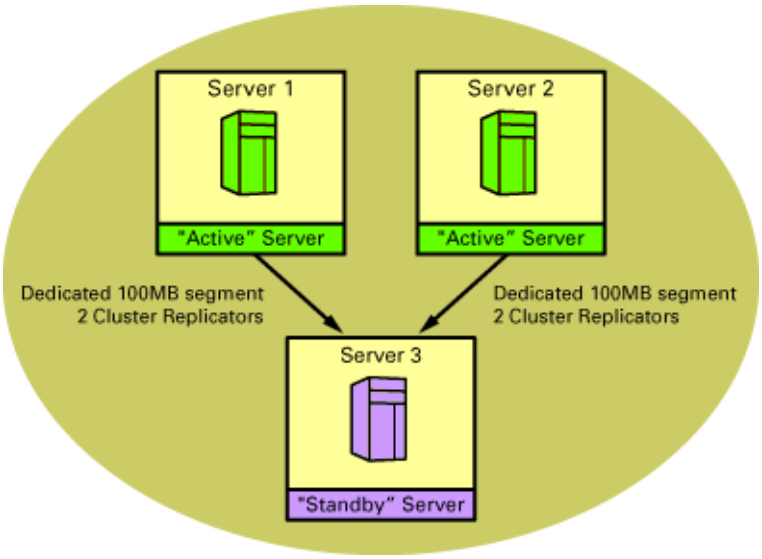
Scenario 4: Solving a cluster performance problem, from-the-field

This final scenario comes from the IBM Lotus Integration Center's work with a client from-the-field. The client was experiencing performance problems with a three-member cluster. So, in this scenario, we tested their configuration and figured out the solution for their problems.

To run this scenario, we set up a Domino cluster with three servers with the following configuration:

- CPUs:
  -- Two IBM Netfinity 7000 with four 200MHz 1MB and 1GB RAM (for Servers 1 and 2)
  -- One IBM PC Server 704 with four 200MHz 1MB and 768MB RAM (for Server 3)
- RAIDs:
  -- Two IBM ServRAIDII configured for RAID5, BIOS 2.40 (one each for Servers 1 and 2)
  -- One IBM ServRAIDI configured for RAID5, BIOS 2.27 (for Server 3)
- Disks:
  -- Six 9GB drives, RAID5
  -- One 44GB array
  -- 8K stripe size RA-WT (write ahead/write-through) cache (C: for OS; D: for Utils; E: for Scanmail quarantine; F: for Domino files and data)
- Two IBM 10/100 EtherJet
- Network: Dedicated 100Mbit segment
- OS: Windows NT Server 4.0 SP3
- Domino: Release 4.6.2a
- Scanmail virus applications
- ARCServe backup

The cluster consisted of two active Domino servers replicating their mail files to one standby server for failover purposes only (no load balancing). Each active server (Servers 1 and 2) had roughly 1450 registered users, and the standby server (Server 3) had about 2650 clustered replica databases. The following diagram shows more details about the cluster configuration:



About our analysis
To determine the best way to improve performance in this scenario, we analyzed both the hardware settings and the Domino cluster configuration.

First of all, we noticed that the servers use default settings on the Windows NT and RAID levels. You can get better performance if you modify these settings. For more information, see the next section "Recommended settings to optimize this configuration."

We then monitored the hardware using the Windows NT Performance Monitor, which revealed a *saturated disk I/O subsystem*. The average disk queue length was *greater than the number of physical drives in the array (>6)*, indicating that writes and reads were waiting on disk to finish. Also, the average disk transfer time was 108ms. This number should be in the 10 to 30ms range. The %Disk Time revealed 100% utilization (75% read and 25% write). The average disk bytes per transfer averaged about 12k. Plus, network utilization was less than 10% on average for both network cards.

Next, we analyzed the Domino configuration by gathering statistics from the servers and analyzing the NOTES.INI settings. Our results showed that most of the performance problems existed on Server 1, which then affected the performance of the cluster. To see the results of the statistics, see the sidebar "**Cluster performance results, from-the-field**."

Recommended settings to optimize this configuration
On Servers 1 and 3, we made the following changes and saw a dramatic improvement in performance:

- Upgraded the BIOS and device drivers for the IBM ServRAID adapter to 2.82 or higher
- Added the following lines to the NOTES.INI of all the clustered servers:
  SERVER_MAXSESSIONS=550
  SERVER_SESSION_TIMEOUT=45 (minutes)
  NSF_BUFFER_POOL_SIZE = 360000000

To further optimize the performance on this configuration, we came up with the following additional recommendations (Note: These recommendations apply to this specific configuration with cost limitations, so they may not apply to all cases):

- Replace Server 3 with a more powerful CPU (such as, Netfinity 7000 M10). In general, you should build clusters with the same class of machine (that is, All Netfinity 7000s or 7000 M10s).

- Use Enhanced RAID1 (RAID 0+1) instead of RAID5. (RAID1 offers higher performance in this case.)
- On Servers 1 and 3, divide the disk I/O into separate arrays. You should use multiple arrays for the operating system and Domino data directory.
- Upgrade the BIOS levels of all components to the latest versions
- Keep the RAID stripe size to 16k with a formatted block size to match
- Move LOG.NSF to another array to free up disk I/O, or turn off replication logging
- Modify the cluster configuration to an active-active-active cluster role, versus active-active-failover

The bottom line is that the disk subsystem is a bottleneck in the Domino system from a hardware perspective. The processors can handle the current load with no problem. Adding drives and rebuilding the array with the proper specifications will improve performance dramatically on the hardware level.

Recommendations from our R4.6 clustering tests

We've drawn the following recommendations and observations from our cluster performance testing on R4.6. We hope that you can use these recommendations to improve the performance of your own clusters. (The second part of this article will include additional recommendations and observations.)

1. The load that clustering adds to a server is proportional to disk write rates due to the workload on the server. The reason is that *cluster replication* causes most of the additional load on a server and on the network. Cluster replication occurs when there are any database changes written to disk, such as the creation of new documents, delivery of new mail messages, and so on. Thus, you can measure the disk writes to estimate the additional load. (For more information on this, see the second article in this series.)

2. You should not rely exclusively on the NOTES.INI settings SERVER_AVAILABILITY_THRESHOLD (SAT) and SERVER_TRANSINFO_NORMALIZE (STN) to properly load balance among cluster members. Domino only redistributes the workload in a cluster when a failover-causing transaction occurs, such as double-clicking on a database icon. You should manually adjust the distribution of users and databases on a regular basis to ensure an even distribution. (For more information on this, see the second article in this series.)

3. If possible, you should restart failed servers during off-peak hours. The reason is that the failed server needs to replicate with the running servers to bring the databases up to date. This replication can cause a heavy load on the cluster. When you restart the server, make sure that the NOTES.INI setting SERVER_MAXUSERS is 0 -- at least until replication gets the databases in synch. In Domino R5, the Cluster Replicator detects servers rejoining the cluster much more rapidly, usually within a minute or two. If you had noticed delays in updating failed servers in your R4.x environment, this should no longer be an issue.

4. You should monitor server statistics, particularly the disk queue, to keep up with your clustering performance. Client response times are not always an indicator that the server is near saturation, because work is buffered in the server queues. The disk queue is typically the most impacted resource. On Windows NT, the disk queue length is the best indicator of I/O saturation, and it should be less than the number of disks in the data RAID set minus one. On UNIX, the %iowait (the percentage of time that

the CPU waits for disk I/O) is a good indicator of I/O saturation, and it should average less than 25 percent. (Note: When you have more than one processor on AIX, getting a valid indication of I/O wait may be a problem.) You should, for example, use the Windows NT Performance Monitor (PerfMon) to monitor your data disk (RAID set). In PerfMon, add the object "Logical disk," counter "disk queue," and instance your data disk. (For more information on I/O and RAID levels, see "**Optimizing server performance: I/O subsystems**.")

5. Consider the load on your server before adding more Cluster Replicator tasks. You can improve the concurrence, and thus, the speed of replication by setting the number of Cluster Replicator tasks equal to the number of servers that you replicate to in the cluster. However, multiple replicator tasks may add load to the server's CPU and disk. This may not be wise if the server is already overloaded.

6. For each cluster database replica, network traffic associated with write disk operations is approximately doubled. If there are two replicas (replicating to two other cluster members), the network traffic associated with writes is approximately tripled. If network LAN traffic becomes a performance bottleneck, you can add a separate intra-cluster communication network over TCP/IP just for cluster communication. Clustering across a WAN is not a problem, as long as you have sufficient WAN network bandwidth to handle the replication traffic.

   Latency exists based on the specific network topology, and increases when overhead is added into the LAN. Overhead is caused by multiple hops. Reducing network path (hops) through gateways and routers helps to improve performance. Upgrading to fast hubs also helps performance. This is particularly true of Ethernet-switched hubs, where the performance gain can be up to three times more data handled on the same network.

   Finally, for the best performance for end-users, locate the heavily accessed servers in close proximity (network wise) to the users.

7. Keep your clustering topology simple by using the least number of servers in a cluster to meet your application availability needs. In most cases, a two or three-member cluster is usually adequate. They work efficiently, and are the easiest to configure and manage. The advantage of a three-member cluster is that in the case of a server failure, there are two remaining members to handle the load. If you cluster more than two servers, you should maintain only one replica copy of a database (as long as this can meet your availability needs). Multiple replica copies can significantly increase the server resource needs. Combining high-level hardware fault tolerance and a Domino cluster with single database replicas should meet most needs. However, if the cluster load is primarily reads -- as is the case with Web sites like Notes.net -- multiple cluster members are not a problem. The reason is that little replication traffic is generated.

8. Make sure to size your servers properly. When update activity is excessive, the ability of cluster replication to maintain database synchronization becomes limited if the server is not sized properly. In rare cases, this may also cause a network traffic problem.

9. Build clusters with nearly identical configurations, using the same class of machines. This way, when failover occurs, the other members can easily assume the additional load (minus the replication traffic from the failed system).

10. You should not use clustering to extend the life of outdated or undersized

equipment that is experiencing performance problems. The overhead in clustering requires that you use appropriately sized equipment.

ABOUT HARRY

Harry Murray joined the Iris Performance Group in 1998. He is currently involved in the testing of Domino R5 on IBM AIX UNIX and NT systems. Prior to joining Iris, he worked for Digital Equipment Corp. in their performance group doing NotesBench testing of Domino on Digital servers. Before that, Harry was involved in the system management of many Digital production systems and was manager of System Technical Support in a number of Digital facilities.

ABOUT GARY

Gary Sullivan joined IBM in 1987 and is currently a marketing support specialist in the IBM Lotus Integration Center. Prior to joining IBM, Gary was the Capacity Planning manager for FMC Corporation. Before that, he worked as a research consultant for Atlantic Richfield.

[back to "Optimizing server performance: Domino clusters (Part 1)"]

Cluster replication test results (sidebar)

The following sections first show our test results for the CPU utilization for 100 users. Then, you can see the results for our light, medium, and heavy user workload tests with varying numbers of users.

CPU utilization for 100 users

The following table shows the percentage of CPU utilization for 100 users in the different clustered/non-clustered configurations. In general, the results show that cluster replication causes the CPU utilization to increase significantly on the active server.

Notice that for Config 4, the results are approximately equal to the sum of the results for Config 2 and 3 -- that is, it's the result of the base workload in Config 2, plus the initiating and receiving replication results in Config 3. The heavy workload is an exception. The disk I/O became a bottleneck for that workload, so the CPU usage was restricted. (The columns don't add exactly since they are derived from separate tests.)

	Config 1	Config 2	Config 3		Config 4
Mail User Type	**Single server, no mail transfers (Total CPU %)**	**Two servers, mail transfers to "standby" server (Total CPU % on active server)**	**Clustered "active" server (CPU % due to initiating replication only)**	**Clustered "standby" server (CPU % due to receiving replication only)**	**Clustered "active" server, initiates & receives replication (Total CPU %)**
Light	5.7	6.5	5.3	4.4	16.6
Medium	7.0	9.3	7.7	6.1	23.2
Heavy	16.0	20.0	18.0	17.0	35.2

Light user workload for varying numbers of users (using Config 4)

Next, we ran the light user workload for 100, 200, 300, and 400 users. This time, we used the Config 4 clustered configuration -- two clustered, "active" servers with mail transfers and replication.

In our configuration, each server has mail transfers (like in Config 2), and both servers initiate and receive replication (like the clustered servers in Config 3). As shown in the table below, the disk write rate for 100 light users is 201Kbytes/s. This value is a result of receiving mail from the clients and from replication of mail from the other cluster member.

The following table shows the complete results for the light user workload:

# of users	CPU utilization (%)	Disk writes (Kbytes/s)	Disk utilization (%)	Client response time (ms)	Probe response time (ms)
100	16.6	201	82.0	49.4	65.3
200	34.6	295	98.5	69.9	106.0
300	34.9	342	99.9	148.5	152.0
400	34.2	346	100.0	246.2	227.0

Note: The client response time is from a NotesBench measurement, and the probe response time is from a Server.Planner measurement. For more information on NotesBench and Server.Planner, see the **NotesBench Web site**.

Medium user workload for varying numbers of users (using Config 4)

Next, we ran the medium user workload for 100 and 200 users. Again, we used the Config 4 clustered

configuration -- two clustered, "active" servers with mail transfers and replication.

The following table shows the complete results for the medium user workload:

# of users	CPU utilization (%)	Disk writes (Kbytes/s)	Disk utilization (%)	Client response time (ms)	Probe response time (ms)
100	23.2	245	85.6	82.0	85.0
200	40.5	384	99.0	161.0	147.0

Heavy user workload for varying numbers of users (using Config 4)
Next, we ran the heavy user workload for 20, 40, 60, 80, and 100 users. Again, we used the Config 4 clustered configuration -- two clustered, "active" servers with mail transfers and replication.

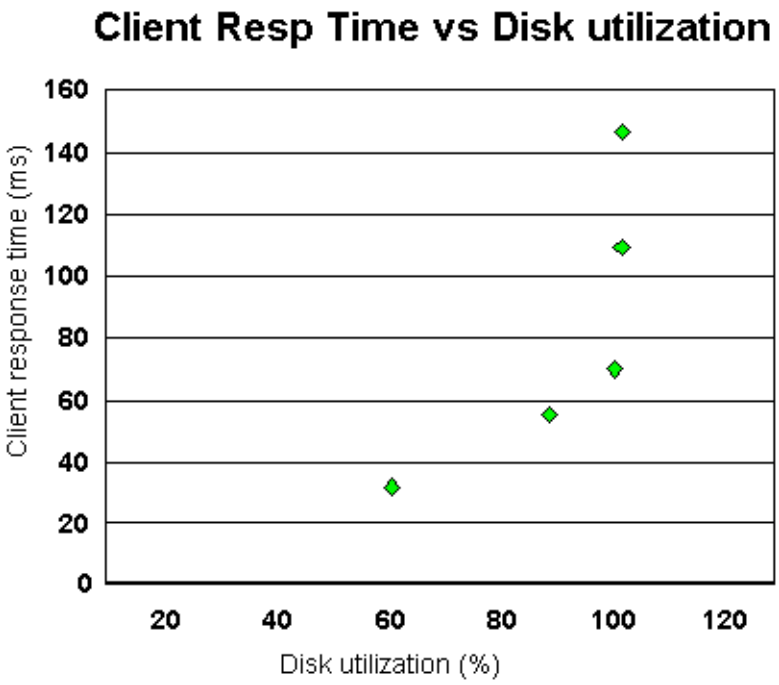
By this time, you can notice that the client and probe response times increase rather steadily with additional users. The response time is probably the most important performance metric because it measures how responsive the server appears to users. If users often experience response times of more than one second, the server is generally considered to be too busy.

The following table shows the complete results for the heavy user workload:

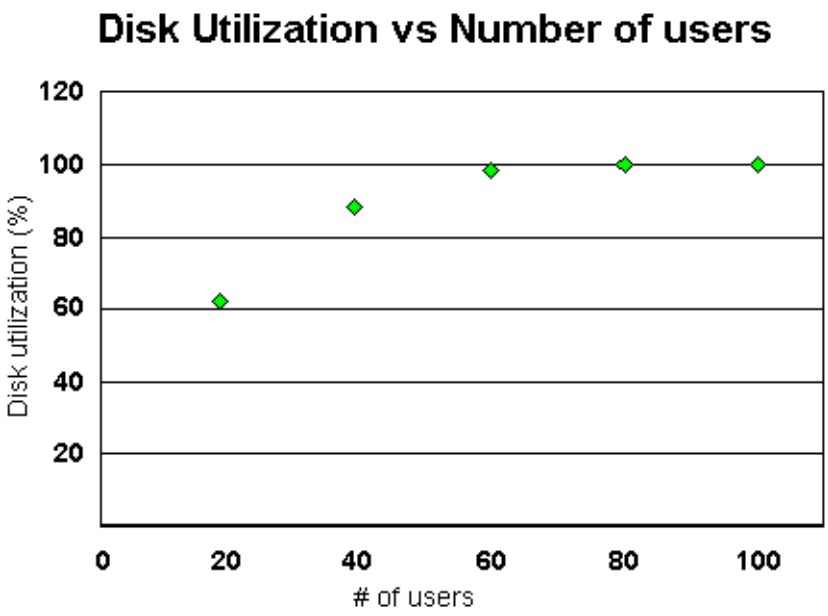
# of users	CPU utilization (%)	Disk writes (Kbytes/s)	Disk utilization (%)	Client response time (ms)	Probe response time (ms)
20	14.6	164	61.2	30.3	50.3
40	25.4	312	89.2	55.4	65.4
60	33.5	417	99.3	70.0	82.5
80	34.0	503	99.9	109.6	85.8
100	35.2	529	100.0	146.6	109.2

Note that at 60 users, the disk subsystem was saturated and was a bottleneck. This is the reason that the CPU usage did not increase for the 80-user and 100-user tests.

The following graph shows further how the disk saturation affected the client response time.



This next graph shows the disk utilization as related to the number of users.



[back to "Optimizing server performance: Domino clusters (Part 1)"]

Multiple Cluster Replicator test results (sidebar)

The following sections show our test results for the CPU and disk utilization for 100 "medium-type" mail users.

CPU utilization

The following table shows the percentage of CPU utilization for 100 "medium-type" mail users on the *active* server in the clustered configurations. The table also shows the percentage difference in CPU utilization when increasing the number of Cluster Replicator tasks.

Cluster configuration	One replicator task (N=1)	*N* replicator tasks (N=# of standby servers)	Difference in utilization (%)
3-way replication (1 active and 2 standby)	23.43	19.98	14.72 (-)%
5-way replication (1 active and 4 standby)	37.08	41.02	10.62 (+)%
6-way replication (1 active and 5 standby)	42.69	43.13	1.03 (+)%

The following table shows the percentage of CPU utilization for 100 "medium-type" mail users on the *standby* server in the clustered configurations. The table again shows the percentage difference in CPU utilization when increasing the number of Cluster Replicator tasks.

Cluster configuration	One replicator task (N=1)	*N* replicator tasks (N=# of standby servers)	Difference in utilization (%)
3-way replication (1 active and 2 standby)	13.30	11.90	10.52 (-)%
5-way replication (1 active and 4 standby)	29.50	32.60	10.50 (+)%
6-way replication (1 active and 5 standby)	27.00	27.30	1.11 (+)%

Disk utilization

The following table shows the percentage of disk utilization for 100 "medium-type" mail users on the *active* server in the clustered configurations. The table also shows the percentage difference in disk utilization when increasing the number of Cluster Replicator tasks.

Cluster configuration	One replicator task (N=1)	*N* replicator tasks (N=# of standby servers)	Difference in utilization (%)
3-way replication (1 active and 2 standby)	55.68	59.60	7.04 (+)%
5-way replication (1 active and 4 standby)	63.86	65.73	2.93 (+)%
6-way replication (1 active and 5 standby)	53.31	58.23	9.23 (+)%

The following table shows the percentage of disk utilization for 100 "medium-type" mail users on the *standby* server in the clustered configurations. The table again shows the percentage difference in disk utilization when increasing the number of Cluster Replicator tasks.

Cluster configuration	**One replicator task (N=1)**	***N* replicator tasks (N=# of standby servers)**	**Difference in utilization (%)**
3-way replication (1 active and 2 standby)	53.00	50.20	5.28 (-)%
5-way replication (1 active and 4 standby)	52.00	54.30	4.42 (+)%
6-way replication (1 active and 5 standby)	49.70	49.30	0.80 (-)%

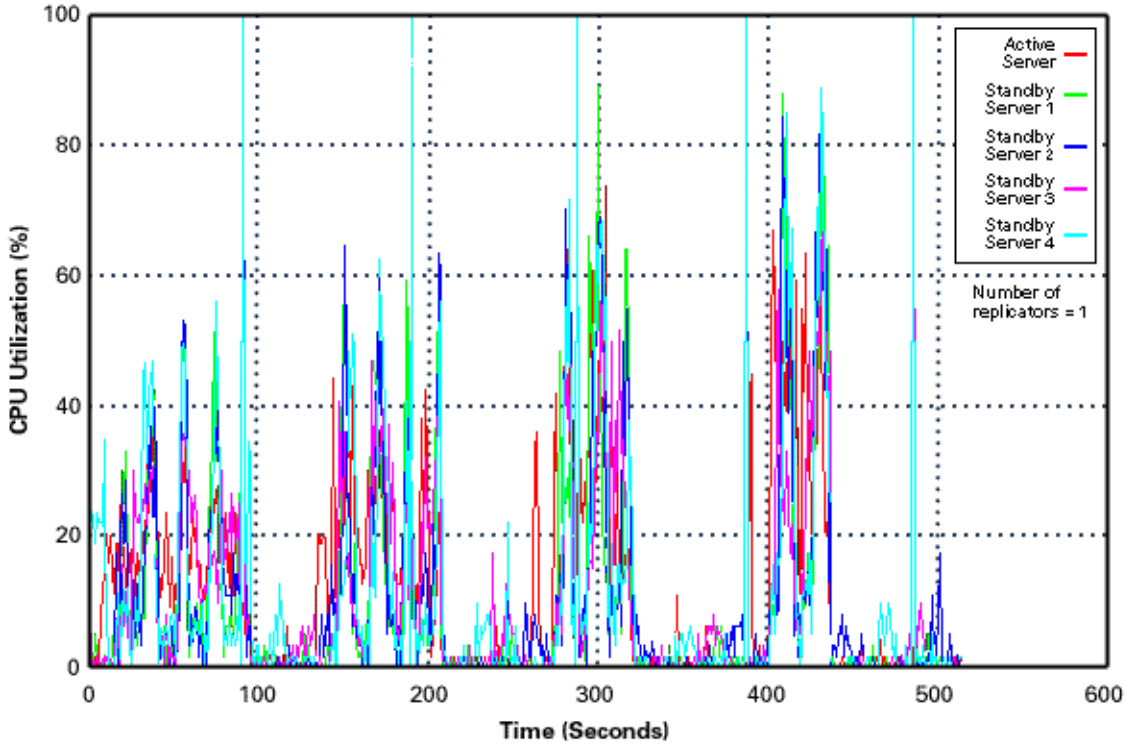
Notes.net Iris Today
Home Download Iris Today Iris Cafe All About Domino Iris Sandbox Doc Library

[back to "Optimizing server performance: Domino clusters (Part 1)"]

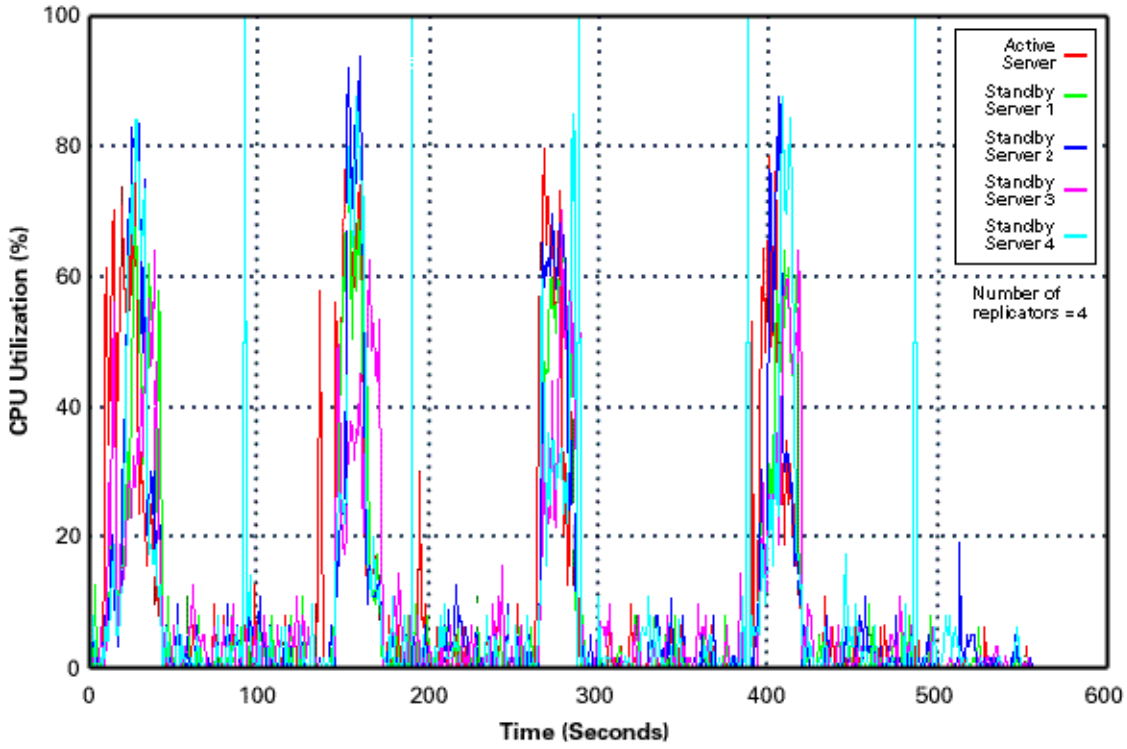
Concurrence in updating replicas results (sidebar)

The following two charts show the CPU utilization when varying the Cluster Replicator tasks from one to four. The CPU utilization indicates the level of activity in a server. Therefore, it shows the timings of when the updates to the mail databases in the active server are propagated to the replicas on the four standby servers. (Note: These results show the timings, while Scenario 2 only showed overall CPU utilization.)

This chart shows the results with one Cluster Replicator on the active server.



This next chart shows the results with four Cluster Replicators on the active server. Notice that in this chart, the bands are narrower than in the previous one, indicating that the replication occurred faster.



[back to "Optimizing server performance: Domino clusters (Part 1)"]

Cluster performance results, from-the-field (sidebar)

The following sections show the Domino statistics that we gathered on the clustered servers. The statistics show a performance problem on Server 1. Server 1 was the first server installed, which resulted in larger mail files. In addition, Server 1 has the most sophisticated users (who send larger mail messages, create more calendar entries, and so on). In fact, Server 1 has twice the activity of Server 2. This then affects the performance on Server 3, which is the failover server for both Servers 1 and 2. The final outcome is that Server 1 experiences serious bottlenecks with cluster replication.

Server 1: Load statistics

The mail, database cache, and database buffer pool statistics all indicate that this server has the maximum number of users that it can support (1450 users). The following statistics show the high mail activity on Server 1.

- Mail.TotalKBTransferred = 409338
- Mail.TotalRouted = 46031
- Mail.Transferred = 12173

The following statistics show the database cache level on Server 1. The DbCache is a table of databases in cache memory. Notice that the database cache peaked out at well over 723 maximum entries, and reached the "high water mark" of 1084 -- the hard limit of Domino R4.x. (R5 allows a much larger DbCache.)

- Database.DbCache.CurrentEntries = 754
- Database.DbCache.HighWaterMark = 1084
- Database.DbCache.MaxEntries = 723
- Database.DbCache.OvercrowdingRejections = 3764

The following statistics show the database buffer pool level on Server 1. The database buffer pool is a cache of 300k per database in the DbCache for view pointers, and so on. You can expect the database buffer pool to reach limits in environments with RAM levels of 1GB or less.

- Database.BufferPool.Maximum = 247692522
- Database.BufferPool.Peak = 244166400

Finally, we looked at the following statistics that show the transaction and user peak levels on Server 1.

- Server.Trans.PerMinute = 1231
- Server.Trans.PerMinute.Peak = 14308
- Server.Users.Peak = 617

Next, let's look at how the heavy load on Server 1 affects the performance on Server 3, the failover server.

Server 3: Load statistics

Remember that in the clustered configuration, both Servers 1 and 2 failover to Server 3. So, Server 3 has more than 2600 mail files. Meanwhile, Server 1 has the heaviest load and its users' mail files often need to failover to Server 3. The following statistics show the database levels on Server 3, which indicate serious performance degradation.

- Database.BufferPool.Maximum = 184183296
- Database.BufferPool.Peak = 185068800
- Database.DbCache.CurrentEntries = 1072
- Database.DbCache.HighWaterMark = 1084
- Database.DbCache.MaxEntries = 723

- Database.DbCache.OvercrowdingRejections = 21321

Server 1: Clustering statistics
With cluster replication and view indexing, the load on Server 3 backs up the cluster replication on Server 1. Server 2 does not send as much data as Server 1, so the cluster replication can continue with no noticeable performance hit. However, Server 1 experiences serious bottlenecks with cluster replication. The following statistics reveal the clustering bottleneck on Server 1:

- Replica.Cluster.SecondsOnQueue.Avg = 1565 (that is, it takes almost 30 minutes for requests to be fulfilled)
- Replica.Cluster.SecondsOnQueue.Max = 5505
- Replica.Cluster.WorkQueueDepth.Avg = 280 (that is, 280 sends are pending)
- Replica.Cluster.WorkQueueDepth.Max = 704
- Replica.Docs.Added = 25594
- Replica.Docs.Deleted = 27038
- Replica.Docs.Updated = 1437
- Replica.Failed = 4826 (not good)
- Replica.Successful = 19682