Best Practices Guide

Sendmail Servers: Scaling Performance through Solid State File Caching

March 2000

Solid Data Systems, Inc. ◆ 2945 Oakmead Village Court ◆ Santa Clara, CA 95051 ◆ Tel +1.408.845.5700 ◆ Fax +1.408.727.5496 ◆ www.soliddata.com

Sendmail Servers:
Scaling Performance through Solid State File Caching

Summary

As e-mail systems scale to support large numbers of users and corresponding message loads, I/O bottlenecks typically occur. The traditional approach to solving I/O problems has been to add servers to an installation, which entails significant increases in capital costs and ongoing overhead costs. Recently Internet Service Providers (ISPs) and other large organizations have been taking advantage of the speed of solid state file caching to efficiently scale their platforms. File caching systems are easily added to e-mail servers, and multiply the performance of each server.

Applications: E-mail servers utilizing Sendmail as the message transfer agent (MTA). Also applies to Q-mail, SIMS, and many other e-mail applications.

Target Environments: I/O bottlenecks often arise when usage patterns reach the following levels:

Total mailboxes	50,000
Number of mail servers	5
Number of users per server	10,000
Number of messages/day	300,000 per day
Messages/second (at peak load)	10

Hot Files: Move Sendmail message queues to solid state storage.
Typical size of a message queue is 500 Mbytes to 1 Gbyte.
(When using IMAP for message access also consider moving the *mailboxes* file; see Solid Data customer success story on University of North Carolina)

Performance Gains: Performance increase of 2X - 4X
(Actual gains depend on site specifics and usage patterns)

Business Results: Improve client performance by vertically scaling e-mail servers. Multiply existing server capacity, reduce the total number of servers required, improve mean time between failure (MTBF), and lower equipment acquisition and management costs. The file caching storage required typically has an acquisition price of $15,000-$25,000. The table below summarizes the overall incremental savings of adding a file cache versus adding more mail servers (on a per-server basis).

Key Attribute	**Adding solid state file cache**	Adding one server to increment performance	Adding servers to match file cache performance
Performance gain	**300% increase**	100% increase	300% increase
Number of servers needed	**One, no addition**	Two, one additional	Four, 3 additional
Ease of installation	**Best**	Lower	Lowest
Floor space required	**None**	Server footprint	3x server footprint
Reliability (MTBF)	**Highest**	Medium	Lowest
Scalability	**Best**	High cost	High cost
Initial hardware costs	**$20,000**	$40,000	$120,000
Annual maint & sys mgmt costs	**$5,000**	$10,000	$30,000
Total annual cost, 3 yr depreciation	**$12,000**	**$23,000**	**$70,000**

Overview

E-mail is the most pervasive application in this "e"-everything age, and the growth in the daily number of e-mail messages is mind-boggling. Estimates are that message count in the United States has increased from 300 million per day in 1995 to over 4 billion per day in 1999. By the year 2003 e-mail traffic may be approaching 20 billion messages per day worldwide (Reference: "E-mail Forecasts and Trends", *International Data Corporation*, July 1999). This growth is fueling demand for an ever-larger base of message servers with dramatically increased performance characteristics.

Sendmail is a software application that provides the foundation for a large percentage of UNIX and NT-based e-mail servers for universities, corporations and ISPs. The software functions as a message transfer agent, managing the queuing, routing and transferring of mail through the Internet. Sendmail has proven to be a robust, functional and effective package – hence its broad adoption. In addition to the original open-source Sendmail application, a number of other e-mail packages are based on Sendmail code and share many of its architectural features. Examples include Qmail and Sendmail Pro.

Problem & Current Practice

Like many applications today, Sendmail was not designed anticipating the tremendous usage levels that are being driven by the broad adoption of e-mail and the Internet. It is now common – in medium-sized ISPs and large user organizations – for peak loads to exceed levels that result in unacceptable response times. This typically occurs when the e-mail traffic per server exceeds 300,000 messages per day and/or 10 messages per second during peak load periods. At such high activity levels, the application experiences difficulty in handling and delivering messages already in the queue, while concurrently accepting new messages.

A common method used to address this situation is to deploy additional e-mail servers. This horizontal scaling approach results in significant cost impacts – including hardware acquisition, software licensing, system administration, maintenance, and floor space/facilities.

Recommended Best Practice

A simpler and more cost-effective approach is to use solid state storage systems to cache the message queues. This will multiply the performance by a factor of 2, 3 or even 4 times, providing a compelling value proposition compared to the proliferation of servers.

Application Discussion

The Sendmail architecture incorporates a Message Relay Server (also called a Message Transfer Agent) to capture incoming messages and hold them temporarily in a buffer file called the Message Queue, as illustrated in Figure 1. The message queues must be placed on an external, non-volatile storage device (such as disk storage or persistent solid state storage) to ensure that no e-mails are lost as a result of system crashes or malfunctions.

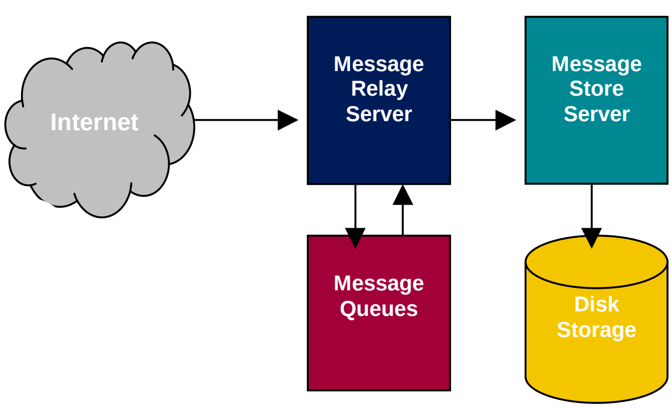

Figure 1
Block Diagram of Sendmail Architecture

A significant structural aspect of the Sendmail program is that it makes use of three temporary files for each delivered message (the file header, message body, and control file for recovery). Three I/O operations are required to write each temporary file to storage, for a total of nine I/O operations per message. As the volume of messages increases, the system reaches a point where the I/O requirements of the message queue exceed the capabilities of rotating disk storage. The result is that the e-mail system response time degrades, as the CPU wastes a large percentage of its time waiting for I/O requests to complete.

The profile of a typical corporate e-mail user might be as follows:

- Each user sends 5 messages per day and receives 25 per day, for a total of 30 messages/day
- Average message is 5-10K bytes in size
- Users log in twice per day
- Busiest hours each account for 10-15% of daily workload

Using this profile, the average user will send or receive about four messages during the peak hours, creating thirty to forty I/O requests. A Sendmail server, using a single rotating disk to store the message queue, can support on the order of 100 I/Os per second. This means 11 messages per second can be processed (at nine I/Os per message), or 40,000 messages total during the peak hours. This translates to a capability of supporting 10,000 users (40,000 messages, divided by 4 messages per user). System response times will start to suffer if more users are added without addressing the I/O problem.

Adding higher-performance processors will not improve throughput, because the processors must wait for the storage device to service requested I/O operations. Adding more disk drives can increase the total available I/Os, but the actual time to complete any one I/O is still about 10 milliseconds. The process requesting the I/O remains open on the CPU waiting for I/O to complete. In the 10 milliseconds that each I/O request waits, the CPU opens thousands of new processes. “I/O Wait” time inefficiently burns up large chunks of CPU cycles.

One way to handle increased users and mail volume is to scale horizontally, by spreading the load across more servers and associated storage. The most efficient way to improve performance is to have each I/O request serviced immediately, freeing up the CPU to do more work. This can be accomplished through vertical scaling by simply moving the message queues to solid state disk. (Reference: *Configuration and Capacity Planning for Solaris Servers* by Brian L. Wong, 1997, pages 169-171)

Solid State File Caching

Solid-state file caching systems have no moving parts, so they experience no mechanical delays when accessing data. They can support random I/O rates measured in thousands per second. This compares to rotating disk products that can support I/O rates on the order of one hundred per second. Figure 2 contrasts the I/O performance of a Sun E450 server using a Solid Data file caching system, versus the same Sun server with mechanical disk drives. File caching delivers an order of magnitude advantage at small block sizes. For more information on solid state storage and file caching see the Solid Data white paper titled “Solid State File-Caching for Performance and Scalability”, http://www.soliddata.com/whitepapers/file_caching.html

I/Os per Second – Solid State vs. Rotating Disk

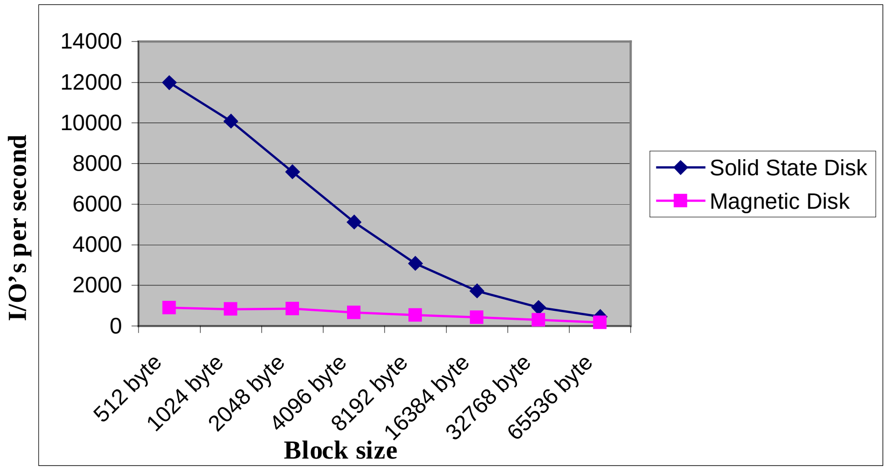

Figure 2

Sendmail is an application that benefits tremendously by placing the message queues onto a solid state file caching system. The individual files are typically quite small, and are ideal for file caching. This allows the overall system to perform at a significantly enhanced level – typically 40-60 messages per second. The total message queue in most systems is about 500MB in size, so the required investment in solid state storage is very cost-effective when compared to other alternatives.

Benchmark Test Results

The benchmark results were recorded by a leading ISP that was evaluating alternatives to expand the capability of its Sendmail server. They were projecting load increases of 300% over twelve months. The ISP was running Solaris on Sun E450 servers. To establish the baseline performance, System Activity Report (SAR) scripts were run during normal operating conditions to collect data that would characterize the application. Solid Data then processed the SAR results using its I/O Dynamics™ software tools to provide a clear representation of system activity. As shown in Figure 3, the system exhibited I/O Wait state conditions that consumed 20 - 30% of system resources during normal operations, and 40% at peak loads. In fact, the time consumed by I/O Wait significantly exceeds the time dedicated to the user application.

File Cache: Performance Impact

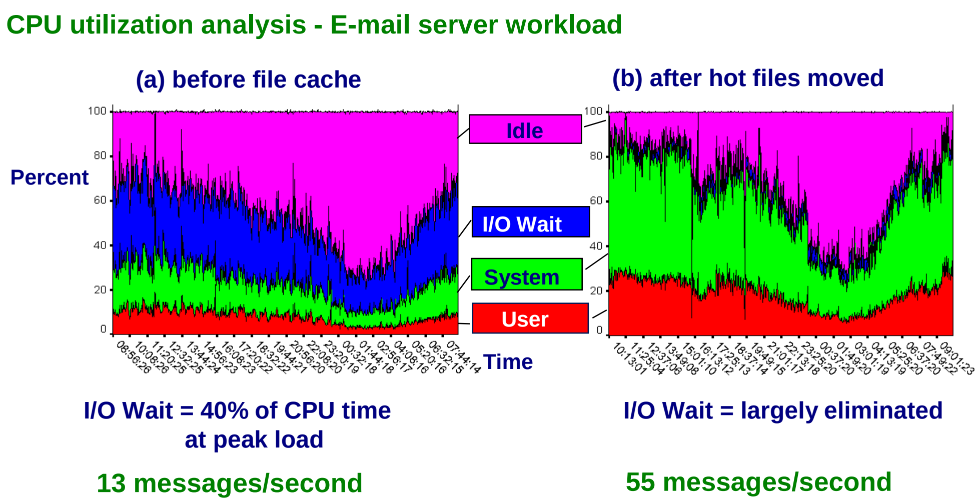

Figure 3
CPU Utilization Profile

Following this characterization of baseline performance, a 536MB solid state file cache was added to the system on a separate SCSI host adapter. The message queue was then assigned to the solid state storage and SAR scripts were run again during normal operations. As shown in the figure, the introduction of solid state file caching essentially eliminated the I/O Wait conditions in the system, allowing the existing user activity to more than double. At the application level, the user recorded an increase in overall system throughput from 13 messages per second to 55 messages per second – an improvement of greater than 300%.

Economics

Solid state file caching offers a better value proposition than the traditional alternative - adding more servers of the same configuration. Major savings are realized in three principal areas: equipment acquisition, system administration and management, and service. The cost of a solid state file caching system is substantially less than the cost of adding more servers to achieve the same throughput.

For the case of the ISP, an improvement of 300% was needed. Scaling with servers meant they would need to go from 4 to 16 servers and add some incremental disk storage. After characterizing the performance impact of solid state storage, they realized that adding 1 GB of file cache to each server would also provide a 300% boost in capacity. Using a 3-year depreciation schedule, the savings in total equipment and overhead are summarized on the following page.

Costs	4 Servers with File Cache	16 Servers with Disk
Server	$160,000	$640,000
Incremental Disk		$20,000
Solid State Storage	80,000	
Total Hardware Cost	*240,000*	*660,000*
Annual Depreciation	80,000	220,000
Annual Service @10%	24,000	66,000
Annual Admin @15%	36,000	99,000
Total Annual Cost	**$140,000**	$385,000
Annual Savings	$245,000	NA
3 Year Total Savings	**$735,000**	NA

Service and administrative costs of the two approaches vary significantly. The annual cost of managing and servicing an IT infrastructure can easily exceed the annual cost of the depreciated hardware. These costs include system administration and management, software licensing fees, service contracts, floor space, utilities and other facility overhead allocations. Using a more conservative estimate (annual service at 10% and system administration at 15% of hardware cost), the file caching approach yields impressive savings. For the ISP in this case study, the selection of solid state file-caching results in an annual savings of $245,000 while improving both reliability and future scalability.

The traditional solution – adding servers and associated disk storage – will have a negative impact on overall system reliability and increase the number of service events. This exposes the e-mail system to a higher risk that a malfunction will result in downtime, unacceptable response times and/or lost messages. In contrast, solid state storage systems, because they have no moving parts and exhibit very high availability, have a negligible impact on reliability and service costs. Solid Data's products have demonstrated a field reliability exceeding 2,000,000 operating hours.

Conclusions

This analysis reflects results achieved in a number of Sendmail applications where growing throughput requirements were exceeding the capabilities of existing platforms. Solid state file caching has been successfully implemented by ISPs that were seeking a scalable and manageable architecture to handle dramatic increases in demand. Large corporations and universities are also finding that the use of solid state file caching is the best approach for handling rapidly growing e-mail requirements.