Analysis of HPSS Performance Based on Per-file Transfer Logs

Wayne Schroeder, Richard Marciano, Joseph Lopez, Michael K. Gleicher,
George Kremenek, Chaitan Baru, Reagan Moore
San Diego Supercomputer Center, San Diego, CA
{schroeder, marciano, lopez, gleicher, kremenek, baru, moore}@sdsc.edu

Abstract

This paper analyses HPSS performance and, to a lesser extent, characterizes the SDSC HPSS workload, utilizing per-file transfer logs. The performance examined includes disk cache hit rates, disk file open times, library tape mount times, manual tape mount times, transfer speeds and latencies. For workload characterization, we examine daily activity in terms of file counts for get and put, and total bytes transferred, as well as activity loads in 10-minute intervals.

Our results largely confirm our expectations but provide more accurate and complete descriptions, with unexpected subtleties. The visual representations provide additional insights into the functioning of the system.

1. Introduction

The High Performance Storage System (HPSS) [1, 2] is software that provides hierarchical storage management and services for very large storage environments. The HPSS architecture is based on the IEEE Mass Storage Reference Model version 5 [3, 4, 5] and is network-centered, including a high-speed network for data transfer and a separate network for control. HPSS supports both parallel and sequential input/output (I/O) and standard interfaces for communication between processors (parallel or otherwise) and storage devices. In typical use, clients direct a request for data to an HPSS server. The HPSS server directs the network-attached storage devices to transfer data directly, sequentially, or in parallel to the client node(s) through the high-speed data transfer network. The primary HPSS development team consists of IBM Global Government Industry and five United States DOE laboratories: Los Alamos, Lawrence Livermore, Lawrence Berkeley's National Energy Research Scientific Computing Center, Oak Ridge, and Sandia.

The load on the HPSS system at the San Diego Supercomputer Center (SDSC) [6] is primarily in support of the high-performance computing environment of SDSC and the National Partnership for Advanced Computational Infrastructure (NPACI). In addition to other roles, SDSC provides the national research community with access to high-performance computers (currently including a CRAY T90, CRAY T3E, and an IBM RS/6000 SP) and conducts applications and technology research and development projects. In 1997, under the new National Science Foundation Partnerships for Advanced Computational Infrastructure (PACI) program, SDSC became the leading-edge site for NPACI [7].

SDSC is running one of the largest production HPSS system in the world, storing, as of August 1998, 65 terabytes in 5.2 million files, and growing rapidly. SDSC also has the most extensive HPSS user activity logs of any HPSS site, with SDSC-added logging for both the PFTP and HSI [8] user interface systems, and for the SDSC Storage Resource Broker (SRB) [9, 10]. HSI is an advanced HPSS Interface utility developed at SDSC that provides basic FTP-style and Unix file-oriented commands, recursive operations, and other capabilities. The SDSC SRB is client-server-based middleware that provides a uniform access interface (API) to heterogeneous, distributed storage resources and devices.

Information recorded in the logs includes date/time, time-to-complete, client host name, HPSS server name, file size, file name, user name, transfer type, transfer speed, Class Of Service (HPSS service partition), and open time. The open time, for tape-resident files, normally includes the time to mount and position the tape.

These activity logs have been ingested into an Oracle database for analysis. SQL queries and plots have been generated to investigate various aspects of HPSS operation. This approach is useful for workload characterization, service interrupt analysis, performance analysis, and tuning.

2. Workload characterization goals

At SDSC, workload characterizations are done on the HPSS to develop an understanding of how well user requirements can be met. The goal is to make archive data retrieval as efficient as possible. Workload characterizations are complicated by rapid variations in user request rates, by the desire to access very old data, and by the need to handle very large data sets. These requirements tend to be mutually exclusive. It is not possible to optimize across all data set sizes when the largest data sets can exceed the size of the disk class. Thus there is no simple set of metrics which can be used to adequately identify the diversity of user requirements, but a number of optimizations and performance measurements can be performed.

A standard approach is to support user requests by providing multiple service classes, each targeted towards a particular service requirement. Examples are service classes based on file size with very large files written directly to tape and small files cached on disk. This allows the system to support files that are larger than the disk cache size, while allowing small files to be retrieved from the lowest latency storage peripheral. Service classes can also be defined based on expected retrieval time. Data sets that will be accessed fairly quickly would be kept on disk, while data sets that are being archived to serve as a backup copy should go directly to tape.

The HPSS is a Hierarchical Storage Management system (HSM), controlling multiple cache levels. The cache levels include:

- Supercomputer file system.
- HSM disk cache (MAXSTRAT and SSA attached RAID disk).
- HSM near-line tape (IBM 3494 and StorageTek silos).
- HSM shelf tape.

Data sets are retrieved from the HSM onto the supercomputer file system through several APIs (parallel FTP; HSI, which supports recursive operations on HPSS directories; and the SDSC SRB, which supports record level access to files on HPSS). Resources are allocated to service classes by explicitly associated disk cache partitions, or tape drives. At SDSC, the service classes are based primarily on file size and number of tape copies. Data sets are automatically assigned to a service class that is best equipped to handle the corresponding data set size. Typical challenges encountered in matching the user workload to the available resources are: Is the HSM able to handle the total workload without choking and building up queue lengths? Can the performance be optimized for the most heavily used service class (minimize access latency and maximize access bandwidth)? Can cache use be optimized such that data is used multiple times from the cache before it is purged?

3. Background

To provide the context for an analysis of archival storage performance and workload study, some background information is needed. The following sections describe the SDSC HPSS hardware and software environments, the SDSC HPSS interfaces (PFTP, HSI, SRB, and the integrated DB2/HPSS system), and logging procedures and issues.

3.1 HPSS hardware (SP) configuration

SDSC is running HPSS on a 14-node IBM RS/6000 SP system running the AIX 4.2 operating system. Each node of the SP is essentially an independent RS/6000 computer but is interconnected with the other nodes of the SP system via a high-performance switch (up to 40 MB per second in each direction). Most of the nodes in SDSC's HPSS SP are multiple-CPU SMPs.

Table 1. SDSC HPSS node configuration.

Node Type	**Number of Nodes**	**Memory per Node**
Silver	8 (32 CPUs)	2-3 GB
High	1 (4 CPUs)	512 MB
Wide	1 (1 CPU)	512 MB
Thin	4 (4 CPUs)	256 MB

This is the current configuration as of late 1998. In September, we upgraded from HPSS 3.1 to 3.2 and moved HPSS production from a set of older nodes to the Silver nodes.

HPSS disk cache space is provided by a combination of locally attached and HIPPI attached RAID, as shown in Table 2.

Table 2. SDSC HPSS disk cache.

Type	**Size**
SSA RAID (7133)	750 GB
MAXSTRAT RAID	1 TB

Tapes are mounted and read/written via tape libraries at SDSC, as described in Table 3. A tape library consists of multiple tape drives and a tape robot to mount/dismount tapes.

Table 3. SDSC HPSS tape libraries.

Type	**Tape Drive Type**	**Number of Tape Drives**	**Approx. Number of Tapes**	**Approx. Capacity (TB)**
IBM 3494 MagStar	3590	8	2,490	50
IBM 3494 MagStar	3590	6	2,490	50
StorageTek 9400 Powderhorn	3590	4	500	10
	3490	4	5,400	4.3

The compression ratio for the 3590 tapes on SDSC data is about 2:1, providing a user data capacity of about 20 GB per tape.

Additional tapes are stored in the machine room and are operator mounted for reading as needed.

In addition to the SP nodes, there is one RS/6000 that runs mover clients for controlling the 3490 tape drives in the StorageTek library.

Network connections include Ethernet, FDDI and HIPPI.

3.2 HPSS software configuration

The core servers run on one SP node (hpss01). These includes the Bitfile server, the Nameserver, the Storage System manager, the Log daemon, Physical Volume Library, Tape and Disk Storage servers, Metadata monitor, Migration, and Purge servers.

All hosts run a Log Client and Startup Daemon.

The SDSC HPSS software components are distributed as shown in Table 4. The PVR is a Physical Volume Repository (tape library controlling software).

Table 4. SDSC HPSS software configuration.

Node	Host	PVRs	Disk Movers	Tape Movers
1	hpss01			
2	hpss03	2	3	1
3	hpss05		2	1
4	hpss07		2	1
5	hpss09		3	1
6	hpss11		2	1
7	hpss13		1	1
8	hpss15		1	1
9	hpss29		1	
RS/6000	val	1		1

For further details see “Configuring and Tuning Archival Storage Systems” [11] presented at this conference.

3.3 HSI/PFTP automatic class of service selection

In an effort to efficiently manage the load between small, medium, and large files, separate HPSS Classes of Service were defined, each with storage characteristics tailored to the type of file destined for that class of service. Based upon analysis of existing UniTree archival storage data, files were initially categorized as follows:

- small files: 0 MB - 2 MB
- medium files: 2 MB - 200 MB
- large files : 200 MB - 1 GB
- huge files : > 1 GB

SDSC also supports automatic creation of a second copy of data, and therefore a second set of classes of service was defined for each of the above categories, with the only difference being that the migration policies for the disk storage class would result in 2 tape copies.

In addition, “direct to tape” classes of service were defined, to allow very large files to be archived without the overhead of first writing to the disk cache, and without needlessly using up valuable disk cache space.

In order to relieve the end user of the burden of making an appropriate Class of Service (COS) selection, SDSC developed code to automatically select a class of service based upon the tuple (filesize, user ID, group ID, account ID, number of copies).

The design of this feature, which is implemented outside of HPSS per se (i.e., did not require changes to HPSS code) allows restriction of Classes of Service by individuals, groups, or accounts, and makes it easy to add resources dedicated to particular projects.

This code, originally developed for HSI (see below), was also incorporated in the PFTP daemon and has been implemented at other HPSS sites. We plan to turn the COS selection code into a library which should eventually become a supported part of the HPSS release.

3.4 HPSS interfaces

SDSC is running four major interfaces into HPSS: Parallel FTP, HSI, the SRB, and an experimental DB2/HPSS system.

3.4.1 Parallel FTP (PFTP). PFTP is currently the primary means of accessing HPSS at SDSC from the Cray and workstation platforms. PFTP is a standard part of the HPSS product, and provides a client-server method for storing and retrieving files using any RFC-compliant FTP client. Normal FTP protocol is used when a standard FTP client is used, and none of the special HPSS features are visible to the end user. HPSS also provides a Parallel FTP Client, which includes full support for parallel transfers, using new HPSS-specific commands such as *pget*, *pput*, and *pappend*. The end user can select the stripe width (number of concurrent transfer operations) and buffer size, if so desired, in order to optimize transfers. Each separate piece of the stripe is handled by a subprocess which is launched when the file transfers begin, and terminated at the end of each transfer.

PFTP (both client and server) was heavily modified at SDSC prior to initial installation to effect a transparent conversion from the NSL UniTree system that was previously used in production. Several features of the UniTree *uftp* interface were deemed mandatory, including SDSC’s passwordless access capability, ability to stack multiple commands on the execute line, and the ability to optionally automatically verify puts to ensure that files had been successfully stored to the first level of the storage hierarchy without errors or without having been modified or corrupted, e.g., by a program which changed the file while it was being saved.

In addition, extensive logging capability was added to the PFTP daemon in order to capture information required for problem analysis/audit trails, and to obtain transfer logging information for analysis. The standard HPSS PFTP logging capabilities were augmented by adding information such as Class of Service selection for files written to HPSS, and “open” times for files read or written. At SDSC, most classes of service are configured such that the entire file is staged to disk cache at open time; the amount of time it takes to open a file is therefore useful in determining tape mount delays, and in bottleneck analyses.

3.4.2 HSI (HPSS Interface). HSI (HPSS Interface) is a new HPSS interface utility that provides a superset of FTP capabilities, as well as a Unix-like interface into the HPSS system. It includes features such as automatic Class of Service selection; recursion for most commands, including the ability to put and get entire file trees; standard Unix commands such as *ls*, *cp*, and *mv*; FTP-like commands such as *get* and *put*; and conditional commands such as *cget* and *cput*. These later commands also provide for the ability to do project management by only putting or getting files which are newer than the existing ones.

3.4.3 SDSC Storage Resource Broker (SRB). The SDSC Storage Resource Broker (SRB) is client-server middleware that provides a uniform interface for connecting to heterogeneous data resources over a network and accessing replicated data sets. SRB, in conjunction with the Metadata Catalog (MCAT), provides a way to access data sets and resources based on their attributes rather than their names or physical locations.

Storage systems handled by the current release of the SRB include the Unix file system, archival storage systems such as UniTree and HPSS, and database Large Objects managed by various DBMSs including DB2, Oracle and Illustra.

3.4.4 Integrated DB2/HPSS system. For storing data collections containing a large number of (small) objects, we are experimenting with a prototype, integrated DB2/HPSS system developed by the IBM T.J. Watson Research Center. In this prototype, the DB2 database system has been extended to allow data to be stored in HPSS files. Database tables are created in logical DB2 storage objects called “tablespaces.” A tablespace consists of one or more “tablespace containers.” Typically, a container is either a UNIX file or a “raw” device. The DB2/HPSS system has extended DB2 such that an HPSS file can be specified as a tablespace container. Thus, when a row is inserted into a table, its data is automatically stored in the corresponding HPSS file. Using this feature, if one defines a BLOB column in a table and stores small data sets in this column, DB2 will automatically store that data in HPSS. Rather than managing each small data set, HPSS only needs to manage the DB2 tablespace containers, each of which can be up to 2 GB in size. Thus, if the average data set size is, say, 1KB then a single HPSS tablespace container file would be able to accommodate about 2 million data sets.

The prototype DB2/HPSS system provides an optional disk buffer between DB2 and HPSS, so that frequently accessed data sets will reside in the DB2 disk buffer thereby avoiding an access to HPSS. Thus, the DB2 disk buffer can relieve a significant amount of load on HPSS for accesses related to HPSS tablespace container files. In order to quantify this benefit, it is necessary for DB2 to monitor the appropriate information related to HPSS access. To facilitate performance studies, it is also necessary to log some of this information. If a disk buffer is utilized between DB2 and HPSS, then several of the parameters that are monitored by DB2 for in-memory buffers will be applicable in this case as well. These include, logical and physical reads, buffer writes, asynchronous reads and writes, victim page cleaners and threshold cleaners triggered, and time spent waiting for a prefetch to complete. Where applicable, these parameters need to include the name of the corresponding file in HPSS. If the optional disk buffer is not utilized then parameters that are monitored in DB2 for non-buffered I/O activity will apply. These include, the number of “direct” reads and writes, and the time taken for direct reads and writes. Finally, the time taken to open and seek HPSS containers needs to be recorded.

Currently, SDSC is evaluating and experimenting with the DB2/HPSS system and logging of this activity has not yet begun.

3.5 Logging information

The primary HPSS external data transactions are recorded in HPSS log files. These include all file get, put and open operations issued by FTP, PFTP and HSI protocols of the HPSS data interface. This information is collected every three hours from every HPSS node and staged on disk in temporary text files. Once a day at 1 a.m. this file is imported into the SDSC Oracle database using the *sqlload* command.

The Oracle HPSS table contains HPSS performance data with the following structure for each transaction: date/time stamp when the HPSS operation took place; how long the operation lasted; name of the client computer issuing the HPSS data request; number of transferred bytes; file name; user name; data transfer direction (read, write); data transfer type (ftp get, ftp put, ftp open, hsi get, hsi put, hsi open); class of service used; transfer speed and name of HPSS data server node. All rows in the HPSS Oracle database are unique, which guarantees that we do not have multiple data for the same transfer.

We also provide a Web-based interface to the HPSS performance database that enables summary queries to be carried out from any Web browser [12]. Using this interface it is possible to display the top 10 users (in categories of either megabytes or files transferred) for the previous day, week, month, and year. It is also possible to narrow the search by type of operations (read/write), by HPSS client name, or by HPSS Class of Service. Using this, we have determined that we have transferred 54.74 TB of data over the 365 days preceding October 22, 1998, the highest transfer speed achieved was 32.9 MB/sec, and the largest file transferred had 35.82 GB. (This is somewhat under-reported, as some transactions were not being logged early in the year.)

As of October 22, 1998, the HPSS transaction database contained 4,931,084 entries. (There is not a one-to-one correspondence between file transfers and rows in this database; an FTP get transfer results in two records: an ftpopen and an ftpget.)

3.6 Unusual activity

It is difficult to characterize the typical SDSC HPSS workload environment. The primary load is from the major applications running on the central high-performance computer systems and from various system backups. These loads vary considerably in sub-hour time-frames but provide fairly constant "background" load in weekly and monthly periods (described below).

In addition to normal loads, however, a number of unusual major events occurred in the time-frame analyzed (CY98). It is actually fairly common for some type of atypical load to occur (populating a digital library, for example), but the following bear special mention.

As a result of the NSF PACI [13] recompetition, data stored in the Cornell Theory Center's (CTC) HPSS system, and data stored in the Pittsburgh Supercomputing Center's (PSC) Data Migration Facility was transferred to SDSC and NCSA.

These moves were accomplished in two different ways:

- The Cornell Theory Center data was physically moved to SDSC, where a second read-only HPSS was set up to allow CTC users to copy their data to wherever they intended to compute.
- The PSC data was transferred across the vBNS network and stored directly into the production HPSS system.

This second transfer began in April 1998. The mode of operation was for PSC personnel to create 2 GB-sized CPIO-format "data packs" of data on a per-user basis, which were then transferred to SDSC into HPSS.

Once they had arrived, SDSC personnel read the 2 GB-sized packs onto Cray local disk, unpacked the CPIO files to recreate the original user's file tree on the local file system, and then used HSI to recursively store the file tree and change the owner and group to the correct settings for all the files.

The transfers from PSC were performed in parallel with the retrieval, unpack, and restore steps, to move the 6.4 TB of data as quickly as possible and minimize the downtime for the users who were being moved. This 6.4 TB of data resulted in 19.2 TB of data movement as the files were moved from PSC to HPSS, from HPSS to the Cray, and back to HPSS.

This influx of PSC data affected the study period in two ways. First, although it is common to have considerable inflow of data related to various projects, the PSC collection was unusually large and extended in time, adding significantly to the normal workload described in the statistics. Second, as described below, the influx of data required HPSS configuration changes that affected the nature of the statistics collected.

The PSC transfers were essentially completed by August 1998, with occasional short bursts of new or retransferred data as users continue to transition from PSC to other NPACI sites.

3.7 HPSS configuration vs. statistics

It is important to note that the HPSS configuration options can significantly affect the meaning of logged values. When configured normally, files read from tape are staged to HPSS disk. In this case, the time value in the ftpopen records have the tape-to-disk time (mount, seek, and transfer), and the ftpget records have the disk-to-disk transfer time. As configured at SDSC, the ftpopen tape-to-disk transfer is for the entire file.

When HPSS is configured to bypass disk cache (reading directly from tape to conserve disk), the ftpopen records only the time it takes to communicate with the tape server. The actual tape mount/seek delays do not appear until the first read is attempted and so is included in the ftpget records. This is due to details in the HPSS I/O design.

HPSS was configured to bypass cache April 23 through April 25. This was needed since the cache was frequently filling due to very heavy use related to the PSC influx.

This difference also appears in specific Class Of Service (COS) configurations. For a tape-only COS, the open time does not include the time mount and seek time. As when disk cache is bypassed, the ftpget time includes these delays.

Unfortunately, HPSS configuration changes are not logged by the system. Although these configuration changes are infrequent, they do occur and a logging of such key parameters would be valuable. Not only do they directly affect the statistics gathered as noted above, but also indirectly affect performance through the normal functioning of the parameters. HPSS, as configured at SDSC, maintains multiple detailed logs on transactions and interactions but not these key high-level factors. Fortunately, such configuration changes are infrequent.

3.7.1 Handling the PSC data. With the tremendous influx of data, metadata transaction processing proved to be a limiting factor, particularly in the area of space deallocation. At times, the system had to be suspended in order for the deallocation mechanisms to catch up and free up space for new data. The extra transaction load resulted in overloading the mirrored file systems used to contain the Encina transaction media archive files, and required constant attention to avoid having the file systems fill up, which would eventually cause the Encina server to die, and crash HPSS. New mechanisms were introduced in the current release (R3.2) of HPSS, which have reduced the transaction load by approximately an order of magnitude, and HPSS version 4.1 reduces the load even further. Metadata transaction processing should no longer be a limiting factor.

The disk caches for HPSS were of insufficient size to contain both the normal non-PSC production load (new data being written as well as staged data being read back from HPSS) and the new PSC load (two reads and two writes for each file), with the result that the disk caches had to be monitored around the clock to keep them from filling up. As a stopgap measure, file staging was disabled for several of the Classes of Service, which caused files to be read directly from tape when retrieved by users, and resulted in long queue delays waiting for tape drives and tape cartridges.

4. Workload/performance analysis utilizing log information

The HPSS interface via the SRB is somewhat different than FTP or HSI. The SRB API provides a file I/O interface and applications may intermix reads, writes, seeks, and non-I/O processing, whereas FTP and HSI will open files either for reading or writing and will quickly process all I/O. So for SRB HPSS activity we log some of the same and some different information. Total transfer time, for example, would be meaningless. The logged information is: current time, time it took to open the file, time the file was open, storage type (HPSS), number of read calls performed, number of bytes read, number of writes performed, number of bytes written, number of seeks performed, open type (open or create), file name, user name, and host name.

The level of SRB HPSS activity is currently very low in comparison to the other HPSS interfaces. Some SRB-related storage activity is via FTP and/or HSI, which are used when large collections are ingested. Much of the SRB activity is to non-HPSS storage systems: Oracle and DB2 databases, and Unix file systems.

Since the PFTP and HSI interfaces handle the vast majority of the SDSC HPSS load, analysis based on those logs will provide an accurate picture of the overall HPSS workload and performance.

The SQL commands used in generating the following graphs are available online [14]. The values produced via the SQL queries were imported into Excel spreadsheets to produce each graph.

4.1 Activity level

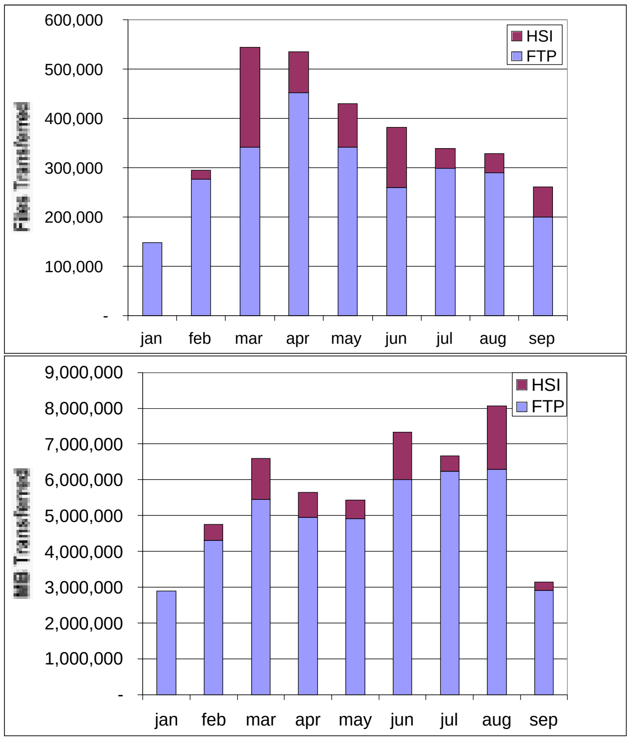
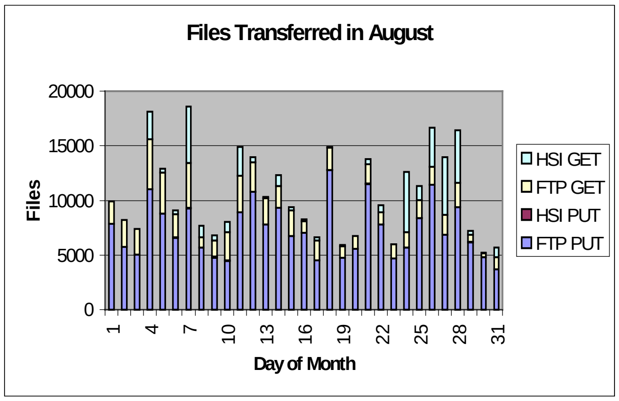


Figure 1. Monthly FTP and HSI activity.

Figure 1 plots the activity for both FTP and HSI in terms of files transferred (into or out of HPSS) and megabytes transferred. In a typical month, more than 5 TB are transferred in 300,000 files. FTP is the predominate interface, but HSI activity is significant.

Much of the activity in 1H98, for both FTP and HSI was the insertion of PSC data into the SDSC archive. Activity for September was somewhat lower than usual due to HPSS software and hardware upgrades performed over the Sept 19-20 weekend. In addition to some downtime, HPSS was run with reduced disk caching for many days.



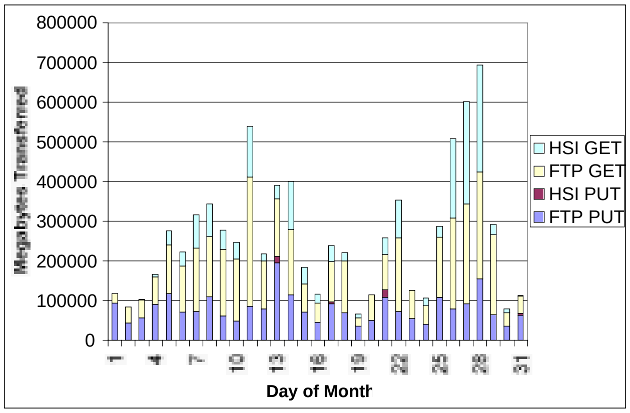


Figure 2. FTP/HSI activity in August.

Figure 2 plots the activity via FTP and HSI for both gets and puts in terms of files and megabytes transferred, for each day in August 1998. Daily megabytes transferred varied from a low of about 80 GB to a high of about 700 GB. Daily count of files transferred varied from a low of about 6,000 to a high of about 18,000. Note that maximum data transferred and number of files transferred occurred on different days.

Figure 3 is file transfer activity (FTP/HSI gets/puts) in 10-minute windows for August 21 (a typical day). There is high variability in activity throughout the day. The first bin starts at midnight (i.e. 12:00 to 12:10 a.m.). The period of no or very low activity in the early part of the day may be due to HPSS configuration changes (reducing disk cache) made in preparation for the upgrade from HPSS 3.1 to 3.2.

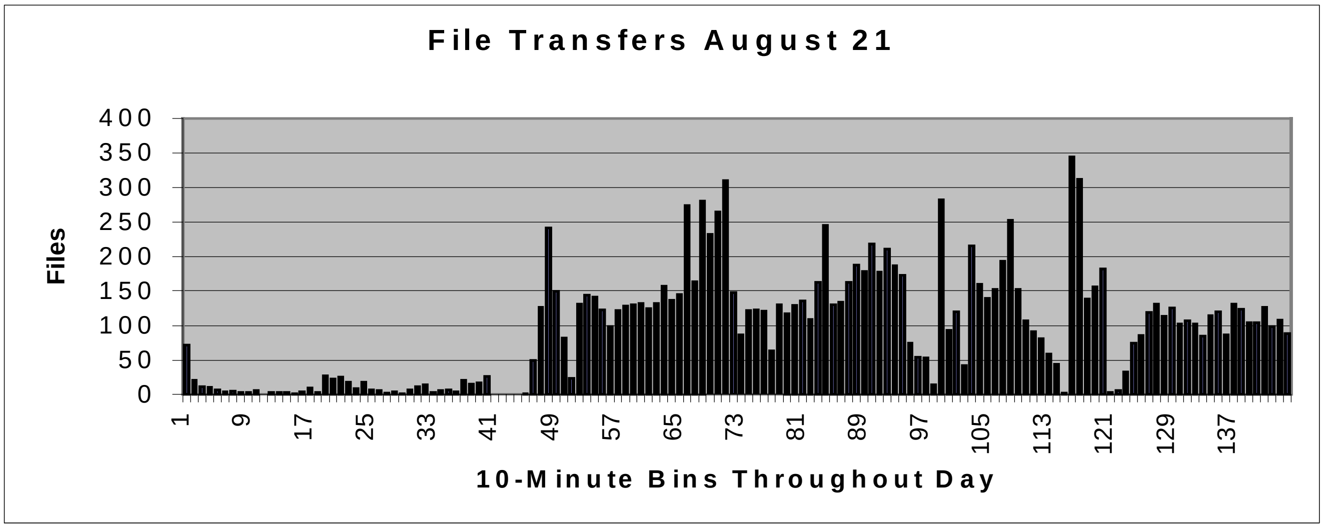


Figure 3. File transfers in 10-minute intervals on August 21.

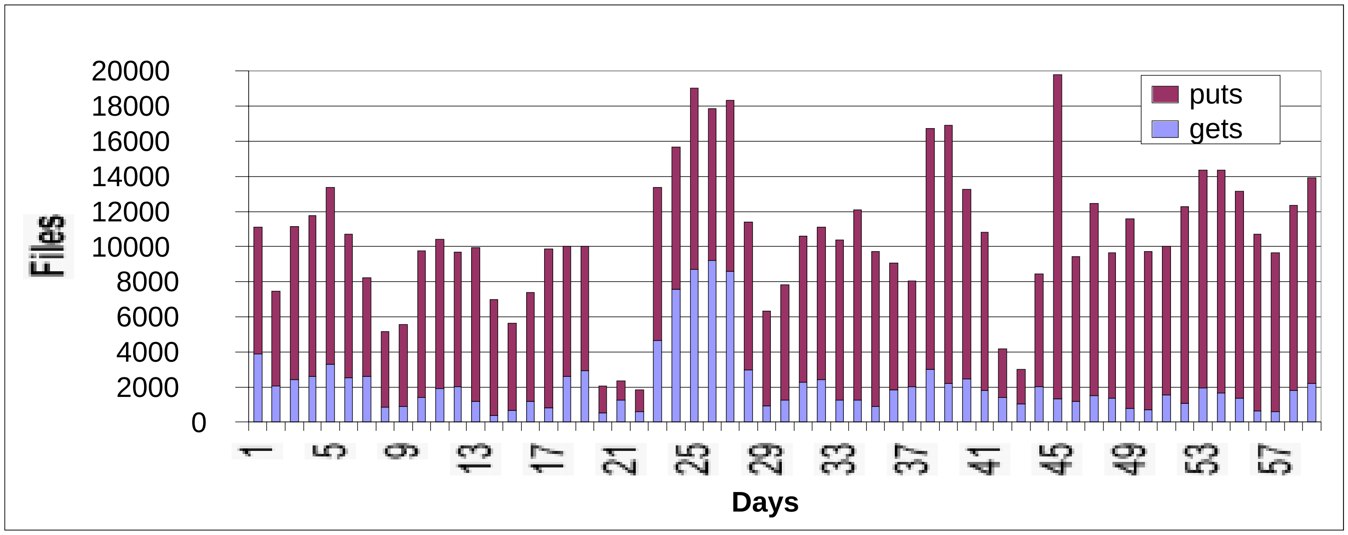


Figure 4a. FTP activity February-March 1998.

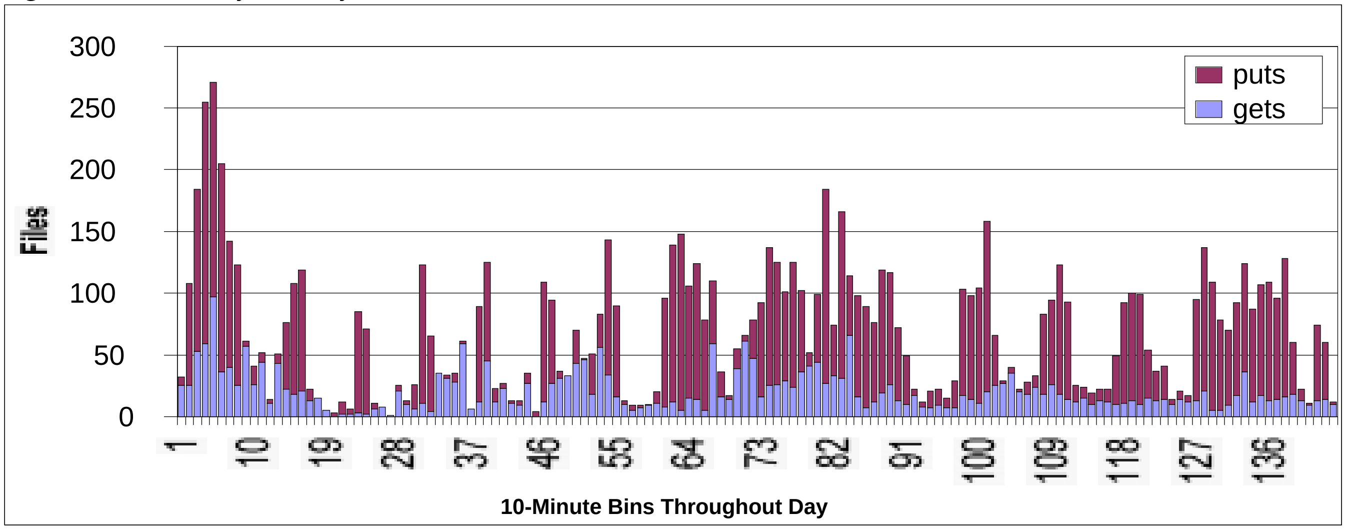


Figure 4b. FTP activity February 19 in 10-minute intervals.

Since the August 21 plot revealed a high degree of variability, graphs of previous data were examined. Figure 4a plots FTP get/put activity in February-March 1998. Figure 4b displays 10-minute intervals for a typical day, February 19. What is notable again, is the high variability.

In developing the Figure 3 plot, we first worked with days preceding August 21 and noticed many 10-minute bins in which no activity took place. This led us to investigate periods of inactivity further.

Figure 5 is the number of 10-minute periods in each day in August in which no file transfers occurred. There were two days in which all 144 10-minute intervals included one or more file transfer (the 13th and 28th, a Thursday and Friday). Of the seven days with high no-activity counts, five were Mondays when HPSS maintenance is performed (August 3, 10, 17, 24, and 31). Wednesday August 19 a log file system filled causing some unscheduled downtime. The other one, August 1, was probably scheduled (Saturday) down-time for system backup.

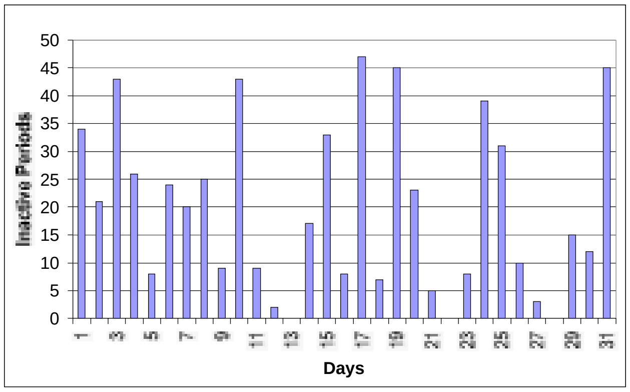


Figure 5. 10-minute periods of inactivity in August (maximum of 144).

Analysis like this could be used to track HPSS service interrupts if supplemented with information from operator logs and HPSS administration records. In SDSC's environment, periods of inactivity exceeding 10 to 20 minutes at any time in any day are very likely to be due to HPSS, client host, or network problems. Some of these could, however, occasionally be due to a lack of demand.

4.2 Time to Open Statistics

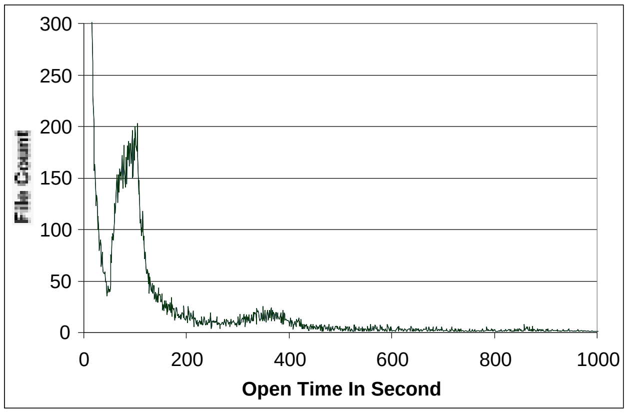


Figure 6. Time to open files.

Figure 6 plots the time it took to open files for reading in February and March 1998, focusing on key areas of the data. This is the time it takes for the HPSS system, in response to a request from the HPSS FTP, HSI, or other daemon, to prepare to read data. If the file is on tape, this normally includes the time to mount and seek to the correct position on the tape. For each FTP get, there is an ftpopen and ftpget record, for FTP puts, there is an ftpstor record.

Not visible in this view is a peak of 74,000 at 0 representing the very large proportion of files opened in less than one second. There is also an extensive tail to the right for one or more files that take well over 1,000 seconds to open.

The following chart contains some this query information. Column A is the seconds taken to open each file; this is truncated to whole seconds (e.g. '0' means between 0 and 0.999999 seconds). Column B is the number of opens that took 'A' time. Column C is a running total of column B. Column D is the running percent of the total. 57% of the file opens took less than 1 second, 73.9% took less than 2, 77.39% took less than 3, and 85% took less than 19. About 80% of the gets were from HPSS disk cache.

Table 5. Selected open time statistics.

A	B	C	D
0	74071	74071	0.570852761
1	21822	95893	0.739031251
2	4536	100429	0.773989442
3	1518	101947	0.785688413
4	903	102850	0.792647682
5	741	103591	0.798358445
6	611	104202	0.803067319
7	534	104736	0.807182768
8	778	105514	0.813178683
9	988	106502	0.820793033
10	842	107344	0.827282186
11	608	107952	0.83196794
12	469	108421	0.835582444
[...]			
11898	1	129746	0.999930639
11991	1	129747	0.999938345
12381	1	129748	0.999946052
12864	1	129749	0.999953759
13347	1	129750	0.999961466
13830	1	129751	0.999969173
14313	1	129752	0.99997688
14624	1	129753	0.999984586
16992	1	129754	0.999992293
28798	1	129755	1

This plot focuses on key areas of the data. A second peak appears at about 90 to 100 seconds corresponding to the typical time for a tape library mount. Another small peak that occurs at about 350 seconds (5 to 6 minutes) is the typical operator time for manual (shelf) tape mounts.

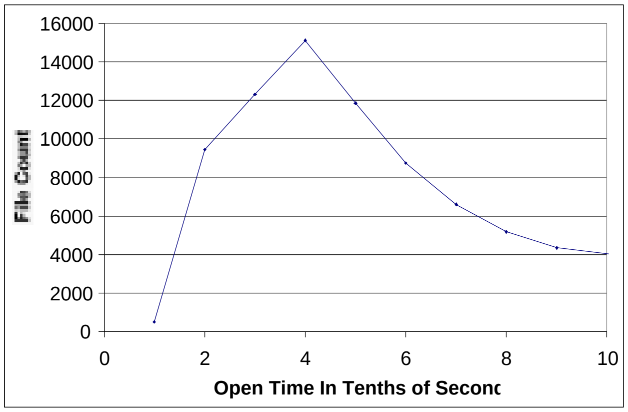


Figure 7. Open time for disk files.

In Figure 7, we can see that disk-cache file opens (opens that take less than 7 seconds) typically take about 400 to 500 milliseconds. In this plot the X-axis units are hundreds of milliseconds. The highest number of open times occurs in the 400 to 499 millisecond bin.

4.2.1 Open times for 100 MB files. Figure 8 plots to open times for files of sizes ranging from 90 MB to 110 MB opened in March 1998. A few data points are not visible: an anomalous open that took over 3,000 seconds, and four points between 500 and 800.

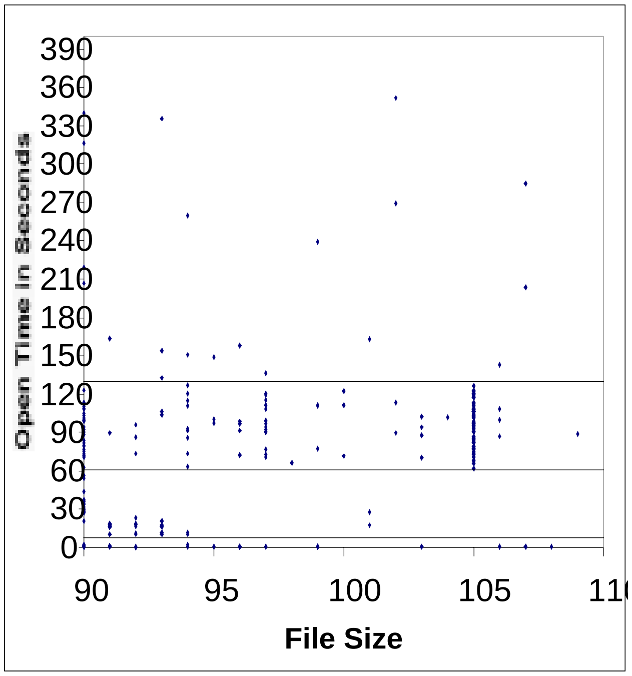


Figure 8. Open times for ~100 MB files.

The natural clustering is visible into four groups, centered at about 1, 20, 100, and above 150. The first corresponds to disk cache, the second probably for files that only require a seek (i.e., a series of files on a tape), the third for the tape library, and the fourth for manually mounted tapes or retrieves that were delayed due to resource contention or tape or system problems. Resource contention delays will occur, for example, when all the tape drives are in use and, briefly, when multiple requests require a particular robot arm.

Thus an accurate grouping of the statistics seems to be 0-8, 9-60, 61-130, and 131 and above. Although this is a bit inaccurate at the boundaries, these groupings seem to fit the visual pattern well. Notice that the maximum for opens less than 8 is under 2, illustrating this natural boundary. Table 6 summarizes the data.

Table 6. Open time summary for ~100 MB files.

seconds	min	max	average	count	percent
< 8	.2464	1.9072	.781	91	27
8-60	10.106	55.881	21.84	60	18
60-130	61.562	126.617	94.9	159	47.5
>130	132.85	3145.67	405.5	25	7.5

It is significant that quite a large number of files (18%) are in the "just seek" group. For our access pattern, placing of files can reduce access time significantly.

4.3 Tape access vs. last reference time

This section looks at the age of some of the files that were accessed from tape in March. As noted above the vast majority of files are in the disk cache and open in within a few seconds. This section looks at the 90 MB to 110 MB files that were retrieved from tape in March.

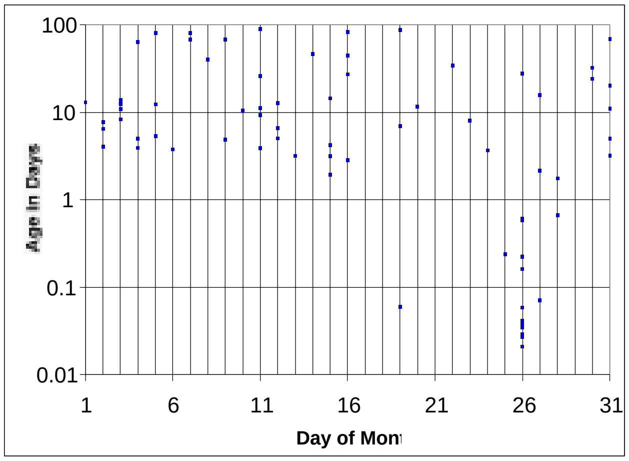


Figure 9. Age of files read from tape.

Each plot point in Figure 9 represents a tape file read. The day of month (March 1-31) on the X axis and the time, in days, of the most recent previous access (get or put) of the file on the Y. The logarithmic Y scale is used to visualize the data points for both the large and small range of the data.

What is plotted is 137 file gets from tape, which is a subset of the total. There were a total of 184 gets from tape in March for 90-110 MB files. The extra 47 gets did not have corresponding previous accesses in the logs and database, which implies that these files were originally stored many months ago. Thus the plot should have an additional 47 points in a range greater than 100 days.

Toward the end of the month, disk caching for these files was not working well. Many files were retrieved from tape after being accessed fairly recently (many within a couple of hours). Presumably, this was due to the disk cache filling, most likely due to the start of the major data import from PSC. This occurred briefly on the 19th, and more commonly on the 25th through the 28th, especially on the 26th.

Information like this can be used to predict the impact of installing additional disk cache space. Most files retrieved from tape are many days old and even with additional disk cache would still have been purged from disk. However, additional cache would avoid a few tape accesses and, of course, a smaller cache would necessitate additional tape accesses.

This pattern is typical of archives in general supercomputing centers such as SDSC. Most stored files are never retrieved. Those that are, are usually accessed within a few hours or few days of being stored. The older files are occasionally accessed.

The SQL queries used to acquire the above data points were relatively compute intensive since it was necessary to correlate multiple records via file name to determine age. To keep this work manageable, we focused on moderate-size files.

4.4 Transfer speeds

Another key aspect of archive performance is the rate at which data are transferred once client and server are prepared. The following chart and plot display the transfer rates seen in July, August, September, and October (through October 27) for files to and from the SDSC CRAY T90 that were larger than 199 MB in size. Units are in megabytes per second (MB/sec).

Table 7. Monthly transfer rates.

		Minimum	**Maximum**	**Average**
July	From HPSS	. 066	6.995	2.674
	To HPSS	.206	5.733	3.394
Aug	From HPSS	.046	5.599	2.046
	To HPSS	.021	6.159	2.626
Sept	From HPSS	.052	6.287	2.831
	To HPSS	1.007	5.762	3.304
Oct	From HPSS	.054	15.156	6.301
	To HPSS	.396	6.375	5.034

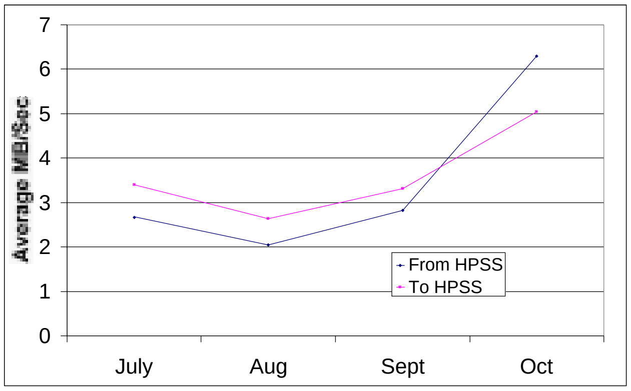


Figure 10. Average transfer speeds (MB/sec).

Typical speeds of 2.5 MB/sec to somewhat over 3 MB/sec were seen with our previous version of HPSS. With the new Silver nodes and HPSS 3.2 installed in September, the average transfer rates have jumped to 5 MB/sec and 6.3 MB/sec, a clear and substantial improvement.

4.5 Workload characterization

As it has often been pointed out in the literature, there are many forms of workload characterization. The approach followed may depend on the purpose of the characterization. Our main objective is to derive a workload that can be used for capacity planning purposes. The tools we will use are simulation-based, rather than analytic.

The studies described in this paper will be the basis for deriving a series of parameters which characterize the workload. Workloads we need to derive include steady-state activity as well as peak period activities. The typical parameters used to describe workloads include: request arrival time, file transfer completion time, HPSS class of service, file size request, type of request, among others.

As described in Figure 8, natural visual clustering of data points can be described, however, a more general approach for workload characterization can be applied, whereby automatic clustering algorithms can be run on data analyzed and compiled from the Oracle database. This is the approach taken in Pythia [15], where a k-means clustering algorithm triggered from a graphical user interface, provides a convenient tool to generalize and automate the process of workload characterization. In this framework, one can visually determine the cluster centroids, and cluster sizes, for a given number of clusters.

5. Related work

The following subsection describes several studies on the analysis of access patterns in mass storage systems, representative of the typical work found in the literature.

The “Analysis of the access patterns at GSFC DAAC” [16], presents an analysis of the GSFC DAAC Oracle database that stores information on client product orders and file requests. File access patterns, caching, clustering, migration, and system loading are discussed.

A more recent study, “The Mass Storage Performance Information System (MSPIS)” [17], discusses a system that allows efficient querying and analysis of UniTree mass storage system logs. The system is to be used for future hardware and software acquisition planning. MSPIS integrates a variety of logs on FTP sessions, tape mounts / dismounts into a Sybase database. Plans are to load the data into a data warehouse and make use of data mining tools.

“Automated clustering-based workload characterization” [18] discusses tools to automate the workload characterization process imposed on mass storage systems to drive performance prediction tools. FTP gets and FTP puts are aggregated into a small number of groups of similar requests. This technique is useful in driving capacity planning studies where system saturation points are predicted.

The term *workload* designates all the processing requests submitted to a system by the user community during any given period of time [19]. Workload characterization is the process of partitioning the global workload of a computer system into smaller sets of workload components—which are in turn composed of transactions or jobs that have similar characteristics—and assigning numbers that represent their typical resource demand and intensity [20].

6. Future work

The analysis of HPSS performance presented in our paper is the first step in a larger methodology for analyzing mass storage systems. Following the general approached prescribed in “Capacity Planning and Performance Modeling: From Mainframes to Client-Server Systems” [20], the analysis we present here will help generate a workload model of the system. This workload model will be used to drive a mass storage performance model, which we are currently building using SES/workbench [21], a discrete event simulation system. Once validated and calibrated, performance prediction studies can be undertaken.

An assessment of workload over the coming months is also of interest. The workload plots from January through September will indicate more about prevailing trends when extended to the end of the year and beyond.

An analysis of the new HPSS system (installed in September) in comparison to the previous is also of interest. Is the disk-cache hit-rate about the same? Is the open time for disk-resident files about the same? And if so, an in-depth study of why HPSS disk opens require 400-500 milliseconds would be in order.

7. Conclusions

A wealth of archive performance and workload information can be obtained by utilizing per-file transfer logs and database technology. This approach is useful in HPSS performance analysis, workload characterization, service interrupt analysis, and modeling. This paper presents some of the initial findings. A number of promising avenues of additional research and analysis are apparent.

The following can be concluded concerning the performance of the SDSC instance of HPSS:

- About 80% of the file gets are from HPSS disk cache. This is, as expected, a fairly high percentage and provides excellent access performance for a large majority of file gets. This is possible due to the adequately large HPSS disk cache and our well-designed Class Of Service structure that meshes well with our workload.
- Tape mounts via the tape library typically require about 100 seconds. This again, matches expected values but adds confidence that our methodology is sound.
- Operator-handled tape mounts (from the shelf) typically take about 350 seconds, again, matching expected values.
- A large proportion of tape mounts are via the tape library indicating that our management of tape location is working quite well.
- Disk-resident files typically require 400-500 milliseconds to be opened. This is somewhat longer than expected. A repetition of this analysis for the current version of HPSS is of interest and possibly further investigation.
- A small, but noticeable, number of files fall into the “just seek” group (i.e., additional files on an already mounted tape).
- An analysis of age of files retrieved from tape could be used to estimate the impact of additional disk cache.

The following can be concluded about the characteristics of our workload:

- The workload is highly variable in the 24-hour, hourly, and sub-hour timeframes.
- The ingest of PSC data, a major unusual event, can be seen in the HPSS activity data as the load increased steadily in February and March, remained high in April and beyond.
- The decline in activity in September is likely due, to a large extent, to the HPSS upgrade which required HPSS to be down for a few days and to operate in degraded mode (utilizing less disk cache) for many days.

It is difficult to separate demand-based workload changes and service-based workload changes. When a particular time period has a relatively low level of activity, it could be either due to changes in demand from the client applications or due to HPSS server or network problems.

As has been the case for several decades, an archive’s workload and performance are fundamentally interdependent. An archive with infinitely high performance would have a much more substantial workload. An archive with poor performance would be used a great deal less. Thus the question of what characterizes an archive’s workload can only be answered in relation to its current performance and the value of the data stored—either in terms of intrinsic value or, if reproducible, in terms of the cost to recreate it.

Acknowledgements

In addition to some of the authors, Tom Sherwin, Antoinette Long-Chinnis, and Charles Lofton have participated in the management of the HPSS system at SDSC.

References

[1] R.A. Coyne, H. Hulen, R.W. Watson, "The High Performance Storage System," *Proc. Supercomputing 93,* Portland, OR, IEEE Computer Society Press, Nov. 1993.

[2] High Performance Storage System (HPSS) home page: http://www.sdsc.edu/hpss.

[3] S. Coleman, "Storage System and Design Issues as Reflected in the IEEE Storage Standards Project—A Tutorial on the IEEE Mass Storage System Reference Model," 11th IEEE Symp. Mass Storage Systems, 1991.

[4] R.A. Coyne and H. Hulen, "An Introduction to the Mass Storage System Reference Model, Version 5," *Proc. 12th IEEE Symp. Mass Storage,* Monterey CA, IEEE Computer Society Press, April 1993.

[5] IEEE Storage System Standards Working Group public documents, http://ssswg.org/public_documents/MSSRM/V5toc.html.

[6] San Diego Supercomputer Center home page: http://www.sdsc.edu.

[7] National Partnership for Advanced Computational Infrastructure home page: http://www.npaci.edu.

[8] M. Gleicher, HPSS Interface (HSI) Hypertext Manual, http://www.npaci.edu/HPSS/HSI.

[9] C.K. Baru, R. Marciano, R.W. Moore, A. Rajasekar, M. Wan, "Metadata to Support Information-Based Computing Environments," http://www.sdsc.edu/~baru/MD97_Paper.html.

[10] SRB Technical Information Page: http://www.npaci.edu/Research/DI/srb/.

[11] R.W. Moore, J. Lopez, C. Lofton, , W. Schroeder, G. Kremenek, M.K. Gleicher, "Configuring and Tuning Archival Storage Systems," Sixteenth IEEE Mass Storage Systems Symposium held jointly with the Seventh NASA Goddard Conference on Mass Storage Systems & Technologies, March 9-11, 1999.

[12] SDSC HPSS Performance Database: http://www.sdsc.edu/hpss_sdsc/cgi-bin/hpss_database.

[13] NPACI award press release: http://www.npaci.edu/News/97/NPACI.html.

[14] SQL Commands used in the production of graphs for this paper: http://www.sdsc.edu/~schroede/sql_hpss.html.

[15] Pythia and Pythia/WK: O. Pentakalos, D. Menasce, Y. Yesha, "Tools for the Performance Analysis of Mass Storage Systems," *Software-Practice and Experience,* Vol. 27(9), 1035-1054 (September 1997).

[16] T. Johnson, J-J. Bedet, "Analysis of the access patterns at GSFC DAAC," NG CMSST 96, p. 153-178.

[17] M. Fahnestock, "The Mass Storage Performance Information System (MSPIS)," talk on April 21, 1998.

[18] O. Pentakalos, D. Menasce, Y. Yesha, "Automated clustering-based workload characterization," 5th GSFC.

[19] D. Ferrari, G. Serazzi, A. Zeigner, "Measurement and Tuning of Computer Systems," Prentice Hall, 1983.

[20] D. Menasce, V. Almeida, L. Dowdy, "Capacity Planning and Performance Modeling: From Mainframes to Client-Server Systems," Prentice Hall, 1994.

[21] Scientific and Engineering Software, Inc. home page: http://www.ses.com.