subject: **SPEC SFS Release 1.1 User Manual**

date: **November 28, 1994**

from: **SPEC OSSC**

ABSTRACT

In the continuing process of establishing and developing suites of benchmarks, SPEC has released an update to the SPEC System File Server Benchmark, Release 1.1 (referred to as SPEC SFS 1.1). Along with the history, goals, and description of SPEC SFS, this manual provides information about running the benchmark, including the Run and Reporting Rules. This manual replaces files released previously as part of SPEC SFS 1.0: README, INTRO, PREREQS, TUNE, and DBUG. This manual includes the files RUN_RULES and RPT_RULES.

Memorandum to
Performance Analysts

subject: **SPEC SFS Release 1.1 User Manual** date: **November 28, 1994**

from: **SPEC OSSC**

1. LICENSE STATEMENT

The SPEC license contains the terms and conditions of your agreement with SPEC and must be carefully read before opening the benchmark program and accompanying documentation (collectively identified as the "materials"). The license represents the entire agreement between you and SPEC concerning the materials. This supersedes any prior proposal, representations or understanding between the parties. By opening the package containing the materials, you have accepted the terms of the license, including that of properly following the SPEC Run Rules and the SPEC Reporting Rules.

2. BACKGROUND

Computer environments are changing rapidly and recent advances in hardware architecture and network software layers have provided significant improvement in computer performance. In view of this, benchmark suites must be updated to provide accurate and adequate performance measurements of today’s systems. The Nhfsstone benchmark originally created by Legato Systems to test the various implementations of NFS[1] (Network File System) Architecture has been taken as the base for developing the 097.LADDIS benchmark by the 097.LADDIS council (L for Legato, A for Auspex, D for Digital, D for Data General, I for Interphase and S for Sun). Later in 1991, 097.LADDIS benchmark was submitted to SPEC for further development, porting and testing as per SPEC’s strictures. After 2 years of extensive testing, the benchmark was released as SPEC SFS Release 1.0 on March 18, 1993, at the UNIFORUM conference in San Francisco, CA. A maintenance release, called SPEC SFS Release 1.1, was issued on November 28, 1994.

The main differences between nhfsstone and 097.LADDIS are:

— the NFS protocol stack is built into 097.LADDIS.

— it compiles on both AT&T and BSD based systems.

— 097.LADDIS presents a load that more accurately represents the desired mix (up to the load generating capacity of the client workstation).

— 097.LADDIS incorporates small random variations in the rate of issuing calls, issues random operations weighted by the desired mix of operations, and chooses files at random (from those available) for each operation.

— 097.LADDIS supports multi-client operation allowing driving a server from multiple networks to higher loads, and getting a single composite report back.

1. NFS: Network File System is the registered trademark of Sun Microsystems, Inc.

An excellent source of additional background information is the USENIX Association white paper entitled "LADDIS: The Next Generation In NFS File Server Benchmarking" included on the release tape.

The SPEC SFS Release 1.1 Benchmark may be similar, but NOT identical to benchmarks or programs with the same name which are available from sources other than SPEC; therefore, it is not valid to compare SPEC benchmark results with anything other than other SPEC benchmark results.

3. ABOUT SPEC SFS RELEASE 1.1

The 097.LADDIS benchmark is a system level file server benchmark for NFS file servers. The benchmark generates a synthetic NFS workload based on a workload abstraction of an NFS operation mix and an NFS operation request rate. SPEC SFS is a suite of benchmarks that currently consists of the 097.LADDIS benchmark. SPEC SFS Release 1.1 is a maintenance update for the 097.LADDIS benchmark. Following the release of SPEC SFS Release 1.1, all future SFS testing will use SPEC SFS Release 1.1.

The SPEC SFS Release 1.1 benchmark includes functionality to manage and synchronize 097.LADDIS execution on multiple clients. In a multi-client 097.LADDIS test, multiple client systems execute the 097.LADDIS benchmark at the same time to generate a load on the server. You run the laddis_mgr program on one system to control the 097.LADDIS tests on two or more clients and to produce aggregate results of the multi-client run. The system that controls the test is called the Prime-Client. The Prime-Client, if it is not the server, can also execute the 097.LADDIS benchmark load generating code at the same time it is performing multi-client management. The laddis_mgr program executes 097.LADDIS on the various systems using the same 097.LADDIS parameters. Because of this restriction, it is best to use similar, if not identical, load generators. Any difference in load generator capacity will create an imbalanced load. The laddis_mgr program can automatically execute multiple benchmark runs with an incrementing load.

The workload generated by the 097.LADDIS benchmark is based on several independent studies of NFS file servers in software development environments conducted by both academia and industry vendors. The workload is also based on heuristics in the absence of hard workload data.

It is not intended that the SPEC benchmarks be used as a replacement for the benchmarking of actual customer applications to determine vendor or product selection.

Running of the benchmark is described in Section 9, "SPEC SFS Release 1.1 Run Rules". The reporting of results must follow the guidelines established in Section 10, "SPEC SFS Release 1.1 Reporting Rules".

4. GETTING STARTED

This section describes how to load the SPEC SFS Release 1.1 benchmark from tape and lists the steps necessary to run the benchmark. The guidelines on how to run the benchmark for reporting results are found in Section 9, "SPEC SFS Release 1.1 Run Rules".

4.1 To read the tape

The tape has been written in QIC-24 format on a 600 ft. cartridge tape. The tape was written with the **tar** command. The default blocking factor for the **tar** command is 20, for a block size of 10240 bytes per block. You will need approximately 4 MB of disk space to hold the benchmark sources. You will need an additional 8 Mbytes to hold all of the benchmark executables and result files for a total of approximately 12 Mbytes of disk space.

A sample command to read the tape is:

```
mkdir spechome
cd spechome
tar xv
```

This will read the tape from the default tape drive into the directory spechome.

If you need to specify an alternate tape device:

```
mkdir spechome
cd spechome
tar xvf /dev/mt/ctape0
```

This will read from the magnetic tape device (mt) which is cartridge tape drive zero (ctape0).

4.2 Problems With Reading

If you have problems with reading the tape, first make sure that you are using the correct blocking factor. The default blocking for your system may be different.

If **tar** is not able to recognize your tape, you may need to use the **dd** command to swap the bytes as they are read from the drive and before they are processed by **tar**. A sample command to read the tape this way might be:

```
dd if=/dev/mt/ctape0 ibs=obs conv=swab | tar xvf -
```

This will use the **dd** command to read the tape, swap the bytes as the tape is read and pass the swapped bytes to the standard input of the **tar** command. /dev/mt/ctape0 should be replaced by the name of the system's tape drive. The **f** flag tells tar to read from a file and the **'-'** is the standard notation for a standard input file to the **tar** command.

4.3 To run the benchmark

1. Read this entire manual, paying particular attention to Sections 9 and 10 giving the Run and Reporting Rules for SPEC SFS 1.1. The manual document file is located in the documents directory.
2. Make a special note of Sections 14 and 15, "Reading Documents" and "Important Note". Section 14 points to the SPEC specific online manual page and describes how to format the manual document file.
3. Build the tools, set the environment, and configure the system under test as specified in Section 5, "Setting Up the Benchmark", and Section 6, "Prerequisites to Running the Benchmark", of this manual.
4. Run the benchmark in one of the ways described in Section 7, "Running The Benchmark", of this document. Section 8, "SFS Tools Interface", gives a detailed description of the run tools available with the benchmark.

5. To generate results according to SPEC guidelines, the benchmark must be run according to the guidelines specified in Section 9 entitled "SPEC SFS 1.1 Run Rules". This is required by license agreement for publicly distributed results.
6. Section 10 entitled "SPEC SFS 1.1 Reporting Rules" describes how results should be formatted. This format must be followed for publicly distributed results.
7. Additional information that may be helpful to you is presented in Sections 11 and 12, "General Debug Information" and "Tuning for Better Performance".

5. SETTING UP THE BENCHMARK

5.1 Environment Initialization

After loading the SPEC suite from the cartridge tape, switch into the uppermost SPEC directory (e.g. cd spechome). This directory is known as the SPEC home directory and is often referred to as $SPEC.

Initialize the user's home environment by doing one of the following:

For C-Shell users :

```
source cshrc
```

For Bourne or Korn shell users:

```
. ./shrc
```

NOTE: Beware of possible incompatibilities between Berkeley Bourne shell and AT&T Bourne shell in the first line of shrc (AT&T users should be O.K.).

5.2 M.<vendor> files

Within each benchmark directory and in some of the binsrc tool directories, there is a set of files of the form M.<vendor>, where vendor is a file extension specifying what vendor/machine for which the file is to be used. These files are provided by computer manufacturers that are members of the SPEC organization. These files provide the necessary overrides for the Makefile to provide vendor-specific compiler options, and the like, for creating the tools and running the benchmark.

If this benchmark is being run on a machine for which no vendor-specific M.vendor files are provided, the user must examine the applicable Makefile in the binsrc and benchspec directories. Careful reading of the Makefiles, along with examination of typical M.<vendor> files, will give valuable clues to the process of preparing an M.<vendor> file for a machine which is not supported by a SPEC member.

5.3 C.<vendor> files

Within the $SPEC/cmdwrappers directory there is a set of files of the form C.<vendor>, where vendor is a file extension specifying which vendor/machine the file is to be used. Like the M.<vendor> files, these files are provided by computer manufacturers that are members of the SPEC organization. These files provide the necessary vendor-specific command paths and commands for the 097.LADDIS remote tools.

If this benchmark is being run on a machine for which no vendor-specific C.vendor files are provided, the user must examine the default C.vendor file to insure that the proper remote command paths and commands are being accessed.

5.4 Build SPEC commands

```
make IDENT=xxx bindir
```

This will compile all of the necessary SPEC-specific commands, create a bin subdirectory and install man pages to describe the SPEC provided tools.

For example:

```
make IDENT=mips bindir
```

IDENT is the generic identifier for your CPU type. It is required as some of the tools need to be built in a machine dependent fashion. This is used to select vendor specific Makefiles of the form M.<vendor> and C.<vendor>.

If no vendor specific options are needed, the IDENT=xxx may be omitted from this command.

Note that if you run runsfs before building the tools, runsfs will build the tools for you. Problems could be encountered with some architectures using the default vendor files.

5.5 Rehash for C-Shell

```
rehash
```

This command is required on some systems so that the C-Shell will know about the commands that have just been added to the SPEC bin directory.

6. PREREQUISITES TO RUNNING THE SPEC SFS RELEASE 1.1 BENCHMARK

The following prerequisite list should be checked before starting a benchmark run.

6.1 Server

1. If the remote tools will be used to mount the target filesystems on the client, the server must allow the "spec" user to NFS mount the target directories.
2. Write access to the exported filesystems for the client system user ID that will run the benchmark.
3. Enough space on the exported filesystems to hold the 097.LADDIS test fileset. The size of the test fileset depends on the load 097.LADDIS offers to the server. For each unit of load (one NFS operation per second), 097.LADDIS requires 5 megabytes (5 * 1024 * 1024 = 5,242,880 bytes) of file space initially, plus 20% more for expansion for a total of 6 megabytes (1.2 * 5 * 1024 * 1024 = 6,291,456 bytes).

For example, to measure a server at a load of 3,500 Ops/sec, 097.LADDIS will require 22,020,096,000 bytes (approximately 20 gigabytes) of file space.

Allow additional space for directories and symbolic links. Every load generator process will create 21 directories, 20 symbolic links, and 100 regular files, plus 40 more regular files per unit of load.

4. RPC services relevant to NFS: portmapper, mountd, and nfs.

5. To eliminate interference between the benchmark and other work, there should be no active users or processes on the server other than 097.LADDIS.

6.2 Load Generators

1. The exported server filesystems must be mounted using identical mount point pathnames on all load generators.

2. The SFS file tree, especially including the benchspec/097.laddis subdirectories, must be in the same location on all load generators. This tree need not be local to the load generators and may be NFS mounted from another machine; or it may be duplicated. If it is not convenient to keep the location constant, symbolic links may accomplish the same effect: for example, /tmp/laddis can be a symbolic link to the actual location if it varies among load generators.

3. The user running the benchmark must be able to run a remote shell from the Prime-Client to each of the load generators. This is accomplished with /etc/hosts.equiv or $HOME/.rhosts.

4. The RPC portmapper must be running on the load generators because laddis_syncd uses it.

5. If the server insists that mount requests come from a ‘‘privileged’’ port (a port with a number less than 1024), then the "laddis" program must be SUID root.

6. To eliminate interference between the benchmark and other work, there should be no active users or processes on the load generators other than 097.LADDIS.

6.3 Prime-Client

1. The Prime-Client, the machine which controls the load generators, need not be a unique machine: it may be one of the load generators or even the server under test.

2. The SFS file tree, especially including the benchspec/097.laddis subdirectories, must be present on the Prime-Client, and it is recommended that they reside in the same location as on the load generators.

3. There must be enough free space in the benchspec/097.laddis/result directory for the files containing results and logs from all of the clients. Each run generates a log file of 10 to 100 kilobytes, plus a result file of 10 to 100 kilobytes, plus one client log from each load generator of one to ten kilobytes each.

6.4 Networks

1. There must be networks to connect the Prime-Client to each load generator, and to connect each load generator to the server under test.

---

1. A disk drive vendor generally reports sizes in powers of 10, so 1 megabyte is 1,000,000 and 1 gigabyte is 1,000,000,000. A memory vendor generally uses powers of 2, so considers 1 megabyte to be 1,048,576 bytes and 1 gigabyte to be 1,073,741,824 bytes. SPEC is using the powers of 10 notation.

2. There must be enough networks to carry the SFS load between the server and the load generators. One Ethernet can carry about 300 Ops/sec, and one FDDI can carry about 3000 Ops/sec.

   For example, to measure a server at 3,500 Ops/sec will require at least twelve Ethernets or two FDDIs.

3. To eliminate interference between the benchmark and other network traffic, there should be no active machines on the test networks other than the server and the load generators.

4. There must be at least two load generators on each network.

7. RUNNING AND TROUBLESHOOTING THE BENCHMARK

The benchmark can be run using one of the methods indicated in this section.

7.1 Running with 'runsfs'

The command, "runsfs", gives a guided step-by-step operation of the benchmark procedures. This interface consists of a main menu providing entry into the various operations the user can perform. Each operation leads the user into submenus that allow the user to modify various files. These menus also keep the user following preset procedures before, during and after running the benchmark suite, and will indicate which other external activities may need to be completed. The menu also provides a convenient escape to shell prompt from which regular UNIX[2] commands can be executed.

The command, "runladdis" also runs the main menu, skipping the setting of the default vendor type and the first two information pages. "runladdis" is in the "$SPEC/benchspec/097.laddis" (or $BENCH directory).

Refer to Section 8, "SFS Tools Interface", for more details about "runsfs and "runladdis".

7.2 Using Individual Makefiles

This is the way to run the benchmark when you are looking at possible portability bugs in the provided benchmark, or using vendor supplied targets not supported at the top level (e.g., pixie). This is done within the $SPEC/benchspec/097.LADDIS directory.

IMPORTANT: If you are going to work at this level, you must first read the Makefile in the $SPEC/benchspec/097.LADDIS directory to understand the process by which the benchmark is run.

For example:

```
cd $BENCH
make -f M.<vendor> target
```

where vendor specifies which vendor specific Makefile to use and target specifies the action to take (e.g. - compare, compile, etc.)

---

2. UNIX is a registered trademark licensed exclusively by X/Open Company Ltd.

For example:

make -f M.mips

7.3 Cleaning up

At times it may be necessary to "clean up" the benchmark directories. There are two makefile targets for this, clean and clobber.

Specifying "clean" removes compiler intermediate files, core files and results. "clobber" performs the same action as "clear" but also removes the benchmark executables.

For example,

make -f M.<vendor> clean

will clean within an individual benchmark directory.

7.4 Failed Makefiles

If you notice that the benchmark is failing in a particular fashion, change to the directory where the failure occurred. In many cases, there are README files that can offer valuable clues as to additional work that must be done to make the program or benchmark run on a machine that it has not previously encountered. If that fails, careful examination of Makefile and the M.* files may be instructive. Also, refer to Section 11, "General Debug Information".

7.4.1 Simple Troubleshooting If you have gotten this far, congratulations! Not that it's hard, but the opportunities for something to go wrong are tremendous. Some simple tips in case you end up having problems:

i. Check out the ends of the laddislog.<suffix> and the laddisc0<client number>.<suffix files>. They will often contain valuable clues as to what went wrong.

ii. Check and make sure that all of the requested file systems are mounted appropriately on all the load generators.

iii. Make sure the benchmark is installed and compiled on all the load generators. Note: The paths have to be exactly the same on all the load generators. The Prime-Client is also typically a load generating client and would also have to be the same.

iv. Make sure that all the systems (load generators, server and Prime-Client) are up and all the appropriate network connections are working.

v. Make sure that all the requested information is correct. Especially check the CLIENTS and MNT_POINTS values in the laddis_rc file.

vi. Make sure the permissions on the underlying subdirectory of the LADDIS mount point allows the "spec" user search permission to allow the LADDIS load generating process to determine the current working directory.

vii. Make sure there is space available in the /tmp directory for synchronization files. The benchmark can fail silently if there is no space available.

viii. If you cannot get the benchmark to work in the multiclient framework, try running the LADDIS benchmark directly and interactively on each client to ensure the basic benchmark is running. Then attempt debugging the multiclient setup.

ix. Pay attention to netmasks and IP address setup. Try pinging the Prime-Client from the failing load generating clients to see if basic communication is possible. Situations can arise where the

Prime-Client can access the load generator, but the load generator does not have a valid route to the Prime-Client.

7.4.2 Other sources of information

See Sections 11 and 12, "General Debug Information" and "Tuning for Better Performance" for further hints on troubleshooting.

8. SFS TOOLS INTERFACE

This section briefly describes the usage of the run tools provided with the SPEC System File Server (SFS) Release 1.1 suite. These tools give helpful menus and submenus to set up the environment, set various LADDIS benchmark parameters, compile the benchmark programs, conduct benchmark validation, execute the benchmark, view results after a successful run and archive the results if required. The results obtained from multiple runs are also collected in a form amenable for ease of use with other result formatting tools. These tools are used on the primary load generator (Prime-Client) for benchmark setup and control as well as on the rest of the NFS load generators (clients) to assist in compiling the programs.

While not required to run the benchmark, these tools can facilitate the "quick" running of the benchmark for tuning various components of the system and results reporting.

This section does not cover the complete Client-Server environment setup in detail. It touches only the portions currently handled by the tools. For information on how to set up and run the SFS suite the reader is advised to refer to Sections 9 and 10 covering the Run Rules and Reporting Rules.

8.1 SFS structure

The SFS Benchmark uses the UNIX "Makefile" structure (similar to other SPEC Suites) to build tools, compile the benchmark source into executables, and to clean directories of all executables. If you are familiar with other SPEC suites, navigating around SFS should be very similar. However, unlike other SPEC suites (for example, "SPEC cfp 92"), the 097.LADDIS "Makefile" does not have run and validate targets. These validation and execution functions are built into the "laddis_mgr" script supplied with the benchmark. This script is used by the menu tools when validate or run targets are chosen.

The benchmark tools are kept (per SPEC convention) in the "binsrc" directory. These tools must be built as described in the next section before they can be used. During the tools build, the executables are transferred to the "bin" directory.

The benchmark source programs are available in the "$SPEC/benchspec/097.laddis/src" directory (where $SPEC is the path in the file system at which the benchmark is loaded (see below)). The "Makefile Wrappers" which contain specific vendor compiler options and flags, are kept in the directory "$SPEC/benchspec/097.laddis". The "Command Wrappers" which contain the vendor specific command paths and commands for the remote utilities are kept in the $SPEC/cmdwrappers directory. After the LADDIS source files are compiled, the resultant executables, along with copies of the necessary scripts, are moved to the "$SPEC/benchspec/097.laddis/result" directory.

The LADDIS parameter file "laddis_rc" is copied into "$SPEC/benchspec/097.laddis/result" directory after the LADDIS programs are compiled. For a single run or a set of runs of the benchmark, some or all parameters of this file, need to be changed. This file is used by the LADDIS manager as well as the tools driving the menus and submenus.

8.2 Setting up the SFS Environment and building tools

After loading the SPEC SFS 1.1 suite from the cartridge tape, change directory into the uppermost SPEC directory (SPEC home directory). The user's home environment can be initialized by executing:

For C-shell users: "source cshrc"

For Bourne or Korn shell users: ". ./shrc"

By executing this command, the SPEC environment variables SPEC, BENCH, RESULTDIR, TESTSRESULTS, etc. are all defined. The SPEC home directory can now be referenced as $SPEC.

After setting up the SPEC home environment, the tools used by all the menus can be created by: "make bindir"

Once the make command completes, the "runsfs" script can be used to complete the installation process, to run the benchmark and to view or archive the results.

8.3 Using the Menus

The command "runsfs" consists of a main menu providing entry into the various operations the user can perform. Each operation leads the user into submenus that allow the user to modify various fields.

The main menu which is listed below, can be reached by executing the "runsfs" command (from $SPEC) or by invoking the "runladdis" command (from $BENCH) and skipping the first two information pages. If "runsfs" is used, the user is initially prompted for a preferred vendor type. If there is a corresponding M.<vendor> and/or C.<vendor> file, they are set as the default wrapper files. If they are not available, the generic laddis wrapper files (M.vendor, C.vendor) are set as the defaults.

The main menu is listed here:

```
MENU (start) >>>>>>>

     Main Menu : 097.LADDIS Benchmark

      1) View/Change/Create M.vendor file
      2) View/Change/Create C.vendor file
      3) View/Change/Create RC file
      4) Remote Client Setup Utilities
      5) Clean LADDIS Source files
      6) Start Compilation
      7) Start Run
      8) View Results
      9) Archive Results
     10) Shell Escape
     11) Exit 097.LADDIS Benchmark


                    Choice : 1 (Hit return after typing)

MENU (stop) >>>>>>>
```

8.4 Wrapper files & Compiling the Benchmark Programs

Wrapper file Modification and compiling of the benchmark programs need to be done on all clients including the Prime-Client. The "Choice" of "1" in the above menu gives a listing of all the vendor makefile-wrappers currently available on the tape. The user can look into any vendor wrapper file and modify it suitably and store the same file on the system under the same or a different name and use it to compile the benchmark programs. These wrappers are all named with a "M." prefix. For example, the AT&T vendor wrapper is named "M.att".

```
MENU (start) >>>>>>>

       List of Available M.vendor wrapper Files
       ----------------------------------------

       att      compaq   dec_osf  dec_ult
       dgc      hp       ibm      ingr
       intel    mips     moto     sgi
       sni      solaris2 sunos4   unicos
       unisys   unixware          vendor


       Current M.vendor file is: M.att
       Enter only vendor part of M.vendor File name.
       Hit Return if using M.att :
```

```
Checking Wrapper file .......

            To Continue Please Press The <RETURN> key:
            Thank You!


     Current Settings

 1)  MACHID          ->  att
 2)  C COMPILER      ->  cc
 3)  C OPTIONS       ->  -O
 4)  C FLAGS         ->
 5)  LOAD FLAGS      ->
 6)  EXTR CFLAGS     ->
 7)  EXTR LFLAGS     ->
 8)  LIBS            -> -lm -lnsl -lnet -lsocket
 9)  EXTRA LIBS      ->
10)  OSTYPE          -> -DSVR4
11)  SETPGRP CALL    ->
12)  RESVPORT_MOUNT  ->

13)  Shell Escape
14)  Save Wrapper File
15)  Return to Main Menu


    Select Setting : 15

MENU (stop) >>>>>>>
```

Each item in the above menu is user definable and it is good practice to "save" the wrapper file under a different name if any parameter is modified.

At this time, option 6, Start Compilation, of the main menu can be used to compile the benchmark programs. Compilation must be done on each client, or on each location that is NFS mounted by a client, before the run is started.

At the end of compilation, the tool sets "root" ownership on the "laddis" executable so that it can perform port binding to a privileged port as shown below, which may necessitate the typing of root password. If requested, please enter the password required by the su(1) command on your system. If you do not have the root password, hit RETURN and laddis will be installed without SUID root; you will need to chown it to root and chmod it to SUID by other means, e.g. asking your system administrator.

MENU (start) >>>>>>>

Setting root ownership on "laddis" setuid executable in order to
perform binding to privileged port. You may be asked to enter
the root password.

To Continue Please Press The <RETURN> key:

MENU (stop) >>>>>>>

8.5 Setting up the LADDIS Parameters

The LADDIS benchmark runtime parameters can be specified by selecting option 3 of the main menu and following the sub-menu as shown here.

Note: The CLIENTS, LOAD and MNT_POINTS parameters MUST be supplied in order to run the benchmark.

Warning: The _rc files may be hand edited, however, any error introduced into the file may cause the tool to abort.

MENU (start) >>>>>>>

Main Menu : 097.LADDIS Benchmark

1) View/Change/Create M.vendor file
2) View/Change/Create C.vendor file
3) View/Change/Create RC file
4) Remote Client Setup Utilities
5) Clean LADDIS Source files
6) Start Compilation
7) Start Run
8) View Results
9) Archive Results
10) Shell Escape
11) Exit 097.LADDIS Benchmark

Choice : 3

List of Available RC Files That End With _rc Latest First
----

laddis1_rc laddis2_rc laddis_rc

Enter your RC File name

Hit Return if using original laddis_rc templates

with default values

Else pick up an " _rc" file from above list: laddis1_rc

Checking RC file ....

Current RC Parameter Settings (Page 1)

Default values are in parentheses.

```
 1)  LOAD                  -> 10
 2)  BIOD_MAX_WRITES       -> 2
 3)  BIOD_MAX_READS        -> 2
 4)  NUM_RUNS              -> 1
 5)  INCR_LOAD             -> 0
 6)  CLIENTS               ->
 7)  MNT_POINTS            ->
 8)  SUT_FS_NAMES          ->
 9)  SUTNAME               ->
10)  PRIME_SLEEP           -> 0
11)  PRIME_MON_SCRIPT      ->
12)  PROCS                 -> 4
13)  LADDIS_DIR            -> /usr/spec-sfs1.1/benchspec/097.laddis/result
14)  WORK_DIR              -> /usr/spec-sfs1.1/benchspec/097.laddis/result
```

NOTE: The rest of the parameters CAN NOT BE CHANGED AS PER SPEC RUN RULES

```
15)  Shell Escape
16)  Save RC File
17)  Continue to Next Page to view the fixed parameters
18)  Return to Main Menu
```

Select Setting : 6

Clients:

To retain this value type <RETURN>

For null value type <space> & <RETURN>

Enter new Clients value : mach1 mach2 mach3

Current RC Parameter Settings (Page 1)

All parameters are "per client"

NOTE: The following 1 - 11 parameters can be user specified.

```
 1)  LOAD                -> 10
 2)  BIOD_MAX_WRITES     -> 2
 3)  BIOD_MAX_READS      -> 2
 4)  NUM_RUNS            -> 1
 5)  INCR_LOAD           -> 0
 6)  CLIENTS             -> mach1 mach2 mach3
 7)  MNT_POINTS          ->
 8)  SUT_FS_NAMES        ->
 9)  SUTNAME             ->
10)  PRIME_SLEEP         -> 0
11)  PRIME_MON_SCRIPT    ->
12)  PROCS               -> 4
13)  LADDIS_DIR          -> /usr/spec-sfs1.1/benchspec/097.laddis/result
14)  WORK_DIR            -> /usr/spec-sfs1.1/benchspec/097.laddis/result
```

NOTE: The rest of the parameters CAN NOT BE CHANGED AS PER SPEC RUN RULES

15) Shell Escape
16) Save RC File
17) Continue to Next Page to view the fixed parameters
18) Return to Main Menu

Select Setting :

MENU (stop) >>>>>>>

8.6 LADDIS Remote Client Setup Utilities

If you want "runsfs" to establish your clients as LADDIS load generators, choose option 4, "Remote Client Setup Utilities", from the main menu.

MENU (start) >>>>>>>

>>>> You have entered the remote_setup_menu

Sub Menu : Remote Client Setup Utilities

1) Copy LADDIS source to Remote Client(s)

2) Default List of remote Filesystem(s)

3) Mount Server File Partitions On Remote Client(s)

4) Unmount Server File Partitions On Remote Client(s)

5) Re-mount Server File Partitions On Remote Client(s)

6) Query Server File Partitions On Remote Client(s)

7) Shell Escape

8) Exit to Main Menu

Choice :

MENU (stop) >>>>>>>

You can select tasks 1, 2 and 3 in order to perform the three steps necessary to setup the Remote Client(s). The "runsfs" command will prompt for the vendor and the _rc file. The tool will offer to set up a "spec" user and prompt for "Y" or "N".

8.7 LADDIS run_prerequisites, validation & Execution

The prerequisites for running the LADDIS benchmark are prompted when "runsfs" command is used. They are listed here.

MENU (start) >>>>>>>

PREREQUISITES TO RUNNING THE 097.LADDIS BENCHMARK

The following prerequisite list should be checked before starting
a benchmark run.

1. The user must create a "spec" account on all LADDIS load generator machines with an identical home directory path, for example "/usr/spec/sfs".
2. Check that the ".rhosts" file on each LADDIS load generator contains the HOSTNAME of the prime LADDIS load generator.
3. Mount the NFS server's filesystems that will be targeted by the LADDIS benchmark on the LADDIS load generator's mount points. The names of the mount points on all LADDIS load generators must be identical.

To Continue Please Press The <RETURN> key:

MENU (stop) >>>>>>>

After the above prerequisites are satisfied, the LADDIS benchmark can be run by choosing option 7 from the main menu. "Run" option automatically forces validation of the benchmark for producing SPEC acceptable results. After validation, the menu reminds the user about "newfs"ing the shared server file partitions to assure that all data files are written afresh on the server disks.

Note: The "runsfs" script will not actually perform newfs's. You must escape the program and perform them manually at this time.

If a run fails for some reason, the tool will advise you of this and possibly direct you where you might be able to find more information. See Section 11, "General Debug Information" for more about tracking down problems with LADDIS.

Hint: The most common problem is usually that file server filesystems are not correctly mounted on the clients.

Reminder: The run step may take many hours to complete depending upon how many data points were requested. Also, some failures may take more than an hour to manifest themselves.

```
MENU (start) >>>>>>>


        Main Menu : 097.LADDIS Benchmark

         1) View/Change/Create M.vendor file
         2) View/Change/Create C.vendor file
         3) View/Change/Create RC file
         4) Remote Client Setup Utilities
         5) Clean LADDIS Source files
         6) Start Compilation
         7) Start Run
         8) View Results
         9) Archive Results
        10) Shell Escape
        11) Exit 097.LADDIS Benchmark


                        Choice : 7


  Using laddis1_rc as the RC file


           Enter suffix for log files, results summary etc
               (Do not exceed 3 chars if there is a 14 character limit): k7


           The Results from this run will be stored in
               /spec_sfs/benchspec/097.laddis/result/LADDISSUM.k7


>> Do you want to run the VALIDATION test ?
           Answer y or n (default is y): y


           >>>>> STARTED LADDIS VALIDATION ON 05/19/94  AT 15:03:57 <<<<<


        Laddis Validation completed


>> Prior to running LADDIS for valid publication data, all targeted
>> file systems on the server are required to be cleaned (newfs'ed).

   Have all targeted server file systems been NEWFS'ed ?
                              Answer y or n (default is y):
```

```
        >>>>> STARTED LADDIS RUNS ON 05/19/94  AT 15:06:37 <<<<<

    Thu May 19 15:06:40 PDT 1994
     Executing run 1 of 1 ...  done


           The results & log files are in /users/sfs/spec-sfs1.1.12/benchspec/097.l
addis/result


                      To Continue Please Press The <RETURN> key:

MENU (stop) >>>>>>>
```

8.8 Viewing the results and archiving

```
MENU (start) >>>>>>>

           Main Menu : 097.LADDIS Benchmark

         1) View/Change/Create M.vendor file
         2) View/Change/Create C.vendor file
         3) View/Change/Create RC file
         4) Remote Client Setup Utilities
         5) Clean LADDIS Source files
         6) Start Compilation
         7) Start Run
         8) View Results
         9) Archive Results
        10) Shell Escape
        11) Exit 097.LADDIS Benchmark


                                 Choice : 8

Current SUFFIX=k7

 List of Suffixes For Which Results Are Available:
 ------------------------------------------------
k1 k2 k3 k4 k5 k6 k7

Enter suffix string for the results you wish to view:
                                Press <RETURN> for k2:
```

```
Searching for Results file
        /spec_sfs/benchspec/097.laddis/result/LADDISSUM.k2 ...


The Results from this run are:

NFS Throughput              Avg. NFS Response Time
in Ops/second               in Milli-seconds

      15                        11.0
      30                        11.5
      60                        12.0
      90                        13.5
     120                        14.5
     150                        16.0
     180                        17.5
     210                        19.0
     240                        22.0
     270                        28.0
     278                        31.0
     300                        47.0
     307                        56.0
     309                        60.0


              To Continue Please Press The <RETURN> key:
              Thank You!


MENU (stop) >>>>>>>
```

8.9 Limitations of the Tools

The menus and sub-menus explained above may not be able to help the user much in case of problems, especially those related to the network layers. Many problems may be eliminated if the user follows the prerequisites mentioned in the "PREREQUISITIES" menu. (A list of key prerequisites is displayed when first running "runsfs". A more complete list of prerequisites is presented in Section 6.) Other problems related to "NFS" or "RPC" operations should be handled outside the tools.

More experienced users may find it more effective to interact more directly with the benchmark as described below.

8.10 Compiling and Running LADDIS without the menu-driven tools

The 097.LADDIS benchmark can be run without using the above-described menu tools. As with the tools, this necessitates setting up the SPEC environmental variables as indicated at the beginning of this section (cd $SPEC; source cshrc or . ./shrc).

cd $SPEC/benchspec/097.laddis. This is the parent directory for compiling the benchmark. The various vendor wrapper files may be found in this directory.

To compile, you need to identify the appropriate vendor wrapper for the load generator system(s) being used. An ls M.* will list the available vendor wrappers. The programs need to be compiled using one of the given wrapper files, (example: M.att) or one that was created using the given M.<vendor> wrapper files.

The command to compile the source programs is:

"make -f M.<vendor>"

The root password may be required in order to set the setuid bit. The executables and other necessary files are copied onto the $SPEC/benchspec/097.laddis/result directory by this command.

The benchmark can be run from this directory using the following command.

"laddis_mgr -r <laddis_rc file> -s <suffix>"

The "-s" option in the command line is required only when a tag is needed to identify the LADDIS run log files and the result files. Use of tags to differentiate results is highly recommended. Note that on 14 character name filesystems, the tag should not be more than 3 characters long. Long name filesystems are not so constrained. The laddis_rc file (which may have any name) supplies the parameters to the benchmark.

LADDIS Validation

For the validation of the SFS suite one of the following commands needs to be used, where the "-v" option indicates the level of validation to be done. In this example level 2 validation will be done.

"laddis_mgr -v 2 -r <laddis_rc file> -s <suffix>"

"laddis_mgr -v 2 -r <laddis_rc file>"

8.11 Troubleshooting Pointers

Refer to Section 7.4, "Failed Makefiles" under "Running the Benchmark", for some hints when troubleshooting the benchmark.

9. SPEC SFS RELEASE 1.1 RUN RULES

This section provides rules to follow for all submitted or reported runs of the SPEC System File Server (SFS) Benchmark suite according to the norms laid down by the SPEC SFS subcommittee and approved by the SPEC Open Systems Steering Committee (OSSC). The following section provides the rules for reporting results of official runs of the SPEC SFS Release 1.1 Benchmark suite.

The general philosophy behind this set of rules for running the SPEC SFS Release 1.1 suite is to ensure that a user can reproduce the results reported by a results submitter.

1. All data reported must be gathered from benchmark runs conducted according to the SFS Release 1.1 Run Rules that produce valid results.

2. The complete hardware, software, and network configuration must be documented within the confines of the reporting pages.
3. Use of software features (in pre-processors, compilers, network protocol suites, etc.) that invoke, generate or use software designed specifically for the SPEC SFS Release 1.1 Benchmark is not allowed.
4. The system, including all hardware, software, and network components must be available for general customer shipment within six months of the date of SPEC result publication.

9.1 Vendor Makefile Wrappers

Generic makefiles are provided in the benchmark directories. Typically a vendor makefile wrapper (M.vendor) is used in conjunction with the generic makefile to run the benchmark. The vendor makefile wrappers are used to compile and run the benchmark for a vendor's specific hardware and software platform.

The makefiles may be modified in a "performance-neutral" fashion to facilitate running benchmark on proprietary operating systems.

9.2 Software

In addition to the base operating system you will require the C compiler; the awk, id, make, nroff, ps, and sed system utilities; the System V version (sh5 on BSD-based systems) of the Bourne shell; and the ed line editor. You will also require the software necessary to run the NFS Version 2 protocol on the server being benchmarked and the NFS client systems used as LADDIS load generators.

All of the software components are required to be the standard software components distributed by the vendor. All the rules specified in the beginning of this Section, "SPEC SFS Release 1.1 Run Rules", apply to all software components. Use of benchmark-specific software components is not allowed.

9.3 Libraries

Special libraries may be used as long as:

1. They are products that must be available for general customer shipment within six months of the date of SPEC result publication.
2. They do not require changing source code,
3. They do not replace routines in the benchmark source code,
4. They are not "benchmark-specific".

Libraries can be specified in the makefiles.

9.4 Pre-Processors

The use of compiler pre-processors is governed by the following rules:

1. Use of a pre-processor on unmodified source code that is part of the identified released software is allowed and must be documented.
2. Use of benchmark-specific pre-processor features are not allowed.
3. Documented pre-processor options may be used, and must be specified on the results reporting page and in the vendor makefile (M.vendor) wrapper.

9.5 Source Code Changes

SPEC permits minimal performance-neutral portability changes. When source code changes are made a diff listing of the changes must be included in the testing report. All changes must be reviewed and deemed "performance-neutral" by the OSSC. These results must be marked with an asterisk ("*") to signify that a portability change was required. The asterisk must accompany such results the first time they are reported in a SPEC Newsletter. If the OSSC accepts these changes as "performance-neutral," the asterisk may be removed from subsequent results using these changes.

Source code changes required for standards compliance should be reported. Appropriate standards documents should be cited. All such changes should be reported to SPEC. SPEC may consider incorporating such changes in future releases. Whenever possible, SPEC will strive to develop and enhance the benchmark to be standards-compliant.

The portability change is allowed if, without the change, the:

1. Benchmark code will not compile,

2. Benchmark code does not execute, or,

3. Benchmark code produces invalid results.

For additional information concerning changes to source code, refer to Section 10, "SPEC SFS Release 1.1 Reporting Rules".

9.6 Server Features

9.6.1 General Features The following are the requirements for the use of general or special server hardware and software features:

1. If special server hardware or software features are used during the execution of the benchmark, they must be documented.

2. The features must be available for general customer shipment within six months of the date of SPEC results publication.

3. The server hardware or software features used when running the benchmark must meet the requirements for SPEC SFS Release 1.1 results being generated, as defined in Section 9.6.1.1.

9.6.1.1 SPEC SFS Results Types

The SPEC SFS Release 1.1 Reporting Rules define which results may may be reported ("reportable results"). SPEC SFS Release 1.1 Reporting Rules form the basis for determining which server hardware and software features are allowed when running the 097.LADDIS benchmark. Furthermore, results for certain types of server configurations may not be reported and are denoted exclusions.

9.6.1.1.1 Reportable Results

The reportable result type is defined to be those benchmark results for which the submitter:

1. Asserts that the server adheres to the NFS Version 2 Protocol Specification, particularly the protocol requirement that for NFS write operations, NFS servers must not reply to the NFS client before any modified file system data and metadata are written to stable storage, as quoted in Section 9.6.1.1.3.

2. Asserts that the server adheres to the SPEC description of stable storage specified in Section 9.6.1.1.3.

3. Asserts that the server passes 097.LADDIS validation.

4. Asserts that the server configuration for running the 097.LADDIS benchmark, adheres to the following description, which applies to all architectures:

   SPEC intends that for every network, all file systems should be accessed by all clients uniformly.

   Once the number of load generating processes has been determined, then load generator mount points should distribute file systems in the following manner.

   Using a round-robin assignment, select the next file system to mount by selecting from the following collection, varying first (a), then (b), then (c), and so on:

   a. next network,

   b. next cluster processor (if clustered system),

   c. other controllers in the path from the network, to the file system,

   d. file system.

   Several examples are useful to illustrate this algorithm.

   I. n-level symmetric multiprocessors (include uniprocessor, i.e. n=1).

      a. Select next load-generating process for a client.

      b. Select next network accessed by that client.

      c. Select next network controller on the network.

      d. Select next disk controller

      e. Select next file system.

   II. Cluster system.

      a. Select next load-generating process for a client.

      b. Select next network accessed by that client.

      c. Select next cluster processor on the selected network.

      d. Select next network controller on cluster controller.

      e. Select next disk controller on cluster controller.

      f. Select next file system on controller.

   III. Functional Multiprocessing.

      a. Select next load-generating process for a client.

      b. Select next network accessed by that client.

      c. Select network processor.

      d. Select next file processor.

      e. Select next storage processor.

      f. Select next file system.

This foundation for this run rule can be found in a paper entitled "Server Configuration Requirements" in the documents directory, filename "ServerConfiguration.ps". This postscript file can be printed using the command:

lp -opostscript ServerConfiguration.ps

9.6.1.1.2 SPEC SFS Excluded Results

SPEC SFS Release 1.1 results may not be reported for NFS servers which incorporate:

1. Volatile memory-based filesystems such as RAM disk.
2. Filesystems that do not survive server reboot.

9.6.1.1.3 SPEC SFS Description Of Stable Storage

The Sun Microsystems, Inc. publication "Network Programming," Revision 50, of December 15, 1987, page 174 states the following concerning the NFS protocol:

> *All of the procedures in the NFS protocol are assumed to be synchronous. When a procedure returns to the client, the client can assume that the operation has completed and any data associated with the request is now on stable storage. For example, a client WRITE request may cause the server to update data blocks, filesystem information blocks (such as indirect blocks), and file attribute information (size and modify times). When the WRITE returns to the client, it can assume that the write is safe, even in case of a server crash, and it can discard the data written. This is a very important part of the statelessness of the server. If the server waited to flush data from remote requests, the client would have to save those requests so that it could resend them in case of a server crash.*

Unfortunately, the above excerpt from the NFS protocol specification does not adequately define "stable storage." To resolve this ambiguity for the purposes of the SFS Benchmark, SPEC defines stable storage in terms of the following operational description:

> *NFS servers must be able to recover without data loss from multiple power failures (including cascading power failures, i.e., several power failures in quick succession), operating system failures, and hardware failure of components (e.g., CPU) other than the storage medium itself (e.g., disk, non-volatile RAM).*

This description, coupled with the NFS Version 2 Protocol's requirement as quoted above, that "for NFS write operations, NFS servers do not reply to the NFS client before any modified file system data and metadata are written to stable storage", lead to the following examples of stable storage:

i. Media commit of data, i.e., the modified data has been successfully written to the disk media, for example, the disk platter.

ii. An immediate reply disk drive with battery-backed on-drive intermediate storage or uninterruptible power system. (UPS)

iii. Server commit of data with battery-backed intermediate storage and recovery software.

iv. Cache commit with uninterruptible power system (UPS) and recovery software.

Conversely, the following are not examples of stable storage:

i. An immediate reply disk drive without battery-backed on-drive intermediate storage or uninterruptible power system. (UPS)

ii. Cache commit without both uninterruptible power system (UPS) and recovery software.

iii. Server commit of data without battery-backed intermediate storage & memory.

9.6.2 Hardware Features

The following are the requirements for the use of general or special server hardware features:

1. The use of volatile memory-based filesystems such as RAM disk on the server are prohibited.
2. The use of filesystems that do not survive reboot of the server are prohibited.

9.6.3 Software Features The following are the requirements for the use of general or special server software features:

1. User Datagram Protocol (UDP) checksums must be calculated for NFS request and reply messages, hence checksums must be enabled on clients.

9.7 Preparing the System for Benchmark Run

9.7.1 General Hardware Requirements

The 097.LADDIS benchmark exercises the CPU, memory, disk, filesystem, and network subsystems of the server being tested. Thus, experimentation may be necessary to determine the amount of memory, number of disks and disk controllers, and number of networks and network controllers required to achieve "best" performance.

9.7.2 General Software Requirements

In addition to the base UNIX operating system you will require the C compiler; the awk, id, make, nroff, ps, and sed system utilities; the SVR5 version (sh5 on BSD- or OSF-based systems) of the Bourne shell; and the ed line editor. You will also require the software necessary to run the NFS Version 2 protocol on the server being benchmarked and the NFS client systems used as LADDIS load generators.

9.7.3 Disk Space Requirements For 097.LADDIS

The SPEC SFS Release 1.1 benchmark uses a collection of Network File System (NFS) clients to generate a load on the NFS server being tested. The NFS clients are called "LADDIS load generators." One LADDIS load generator, denoted the "prime LADDIS load generator," is selected to control the benchmark and consolidate the results and run logs of the individual LADDIS load generators.

The 097.LADDIS benchmark requires disk space on the server being tested and the NFS clients used as LADDIS load generators. The benchmark requires a variable amount of disk space on the server for the target file data generated by the LADDIS load generators. On the LADDIS load generators, the benchmark requires a fixed amount of disk space for benchmark programs and binaries, and a variable amount of disk space for capturing benchmark results and run logs. The prime LADDIS load generator requires additional disk space for the consolidated results of all of the individual LADDIS load generators.

Disk space requirements for the 097.LADDIS benchmark can be summarized as follows:

1. Server Requirements: The server must be configured to provide an amount of disk space equal to five megabytes (5 Mbytes = 5 * 1,048,576 bytes) per NFS operation per second generated by the collection of LADDIS load generators, plus an additional 20% for fileset growth during benchmark execution. For example, at a load level of 100 NFS operations per second, the server must provide 600 Mbytes of disk space for the files generated by the benchmark. The 600 Mbyte disk space requirement for this example can be calculated as follows:

   [600 Mbytes = ((5 Mbytes)*(100 NFS operations per second)) +
   ((0.20)*(5 Mbytes)*(100 NFS operations per second))]

2. LADDIS Load Generator Requirements: Each LADDIS load generator requires approximately 4 Mbytes of disk space for the benchmark source code and binaries. Further, each load generator requires approximately 6 Kbytes (6 Kbytes = 6 * 1024 bytes) of disk space for benchmark results and log files for each load level (test point).

3. Prime LADDIS Load Generator Requirements: In addition to the LADDIS load generator disk space requirements described in item 2 above, the prime LADDIS load generator has a further disk space requirement for consolidated benchmark results and log files from each load generator of 30 Kbytes + ((6 Kbytes)*(the number of 097.LADDIS load generators)) for each load level (test point).

9.8 Running the SPEC SFS Release 1.1 Benchmark

9.8.1 Requirements

This section details the requirements governing how the SPEC SFS Release 1.1 benchmark is to be run for the purpose of generating reportable SPEC SFS Release 1.1 results.

9.8.1.1 097.LADDIS Benchmark Execution Requirements

The following 097.LADDIS benchmark execution requirement must be followed when generating reportable SPEC SFS Release 1.1 results:

1. NFS servers must perform write operations in a manner that is consistent with the requirements for the Reportable SPEC SFS result type, as described in Section 9.6.1.1.1.

Each benchmark run consists of a set of requested NFS load levels for which an average response time measurement is made. The benchmark measures the actual load level achieved and the associated average response time for each of the requested NFS load levels.

At the start of each benchmark run, i.e., before the first in a series of requested NFS load levels is generated, the target filesystems on the NFS file server being benchmarked must be initialized to the state of a newly-created, empty filesystem. For UNIX-based systems, the "mkfs" (make filesystem) or "newfs" (new filesystem) command would be used for each server filesystem to be used by the benchmark. For non-UNIX-based systems, a semantic equivalent to the "mkfs" or "newfs" command must be used.

The measurement of all data points used to define a performance curve is made within a single benchmark run, starting with the lowest requested NFS load level and proceeding to the highest requested NFS load level. The requested load levels are specified in a list, from lowest to highest, from left to right, respectively, in the LADDIS parameter file laddis_rc.

The "runsfs" script described in Section 8 allows the user to enter the information required for a benchmark run, prompting the user for any missing information. When all pertinent information has been specified by the user, the runsfs script generates a 097.LADDIS parameter file for a benchmark run.

Please refer to "SPEC SFS Release 1.1 Reporting Rules", Section 10, for the required measurement points necessary to report a complete server response/throughput curve.

No server or testbed configuration changes, server reboots, or file system initializations (e.g., "newfs") are allowed between requested NFS load levels.

If any requested NFS load level must be rerun for any reason, the entire benchmark run must be restarted, i.e., the server's filesystems must be initialized and the series of requested NFS load levels repeated.

9.8.1.2 097.LADDIS Benchmark Parameter Requirements

All 097.LADDIS benchmark default parameter values must be used when generating reportable SPEC SFS results, except as noted in the following list:

1. **Load:** A collection of NFS clients called LADDIS load generators is used to generate an aggregate NFS load on the server being tested. For a testbed composed of 'N' LADDIS load generators, each generator produces 1 'N'-th of the load given the design of the 097.LADDIS benchmark.

2. **Server Fileset:** The size of the fileset generated on the server by the 097.LADDIS benchmark is established as a function of requested NFS throughput. Thus, fileset size is dependent on NFS throughput across the entire results curve. This provides a more realistic server load since more files are being manipulated on the server as the NFS load on the server is increased. This reflects typical server use in actual computing environments.

   The total fileset created by the 097.LADDIS benchmark is sized at 5 MBytes per NFS operation per second. The number of files within the total fileset is established based on the relationship of 8 files per megabyte of fileset size.

   The 097.LADDIS benchmark accesses a subset of the total fileset during load generation. The subset is denoted the working fileset or simply, working set. The working set is sized at 20% of the total fileset size, i.e., 1 Mbyte per NFS operation per second. The files in the working set are selected according to a Poisson distribution of 20% of files in the total fileset.

   The default parameters of the 097.LADDIS benchmark allow the benchmark to automatically create a total fileset and working fileset on the server that adhere to these rules.

3. **Number of Server Filesystems:** The number of server filesystems used to contain the total fileset is determined by the individual executing the benchmark. The selected number of filesystems must allow the total fileset size to be achieved for the requested NFS load levels. The number of server filesystems is specified by the MNT_POINTS parameter in the laddis_rc parameter file.

4. **LADDIS Load Generators:** At least two LADDIS load generators must be used per network supported by the server. For example, if a results submitter claims that a given server supports four Ethernets, then at least eight load generators (two load generators on each of the four Ethernets) must be used in the testbed configuration.

   Note: Bridged Ethernets require that two load generators be on each physical segment, rather than logical segment.

   It is recommended that "diskful" NFS clients (systems with a local disk supporting the operating system, swap space, and user files) be used as LADDIS load generators. This recommendation is made to ensure that performance measurements made by the 097.LADDIS benchmark are not skewed by network traffic not generated by the 097.LADDIS benchmark. For the same reason, it is also recommended that load generator log files be created on the local disk of the diskful NFS client.

   The NFS clients used as LADDIS load generators are defined by the CLIENTS parameter in the laddis_rc parameter file.

5. **Number of Processes per Load Generator:** The number of LADDIS subprocesses per LADDIS load generator is determined by the individual running the benchmark according to the following guideline. The number of subprocesses per load generator is specified by the PROCS parameter in the laddis_rc LADDIS parameter file.

A minimum of eight LADDIS subprocesses per network supported by the server must be used. For example, for a testbed consisting of two LADDIS load generators targeting a server on one Ethernet, at least eight LADDIS subprocesses must be used on the load generators, hence at least four subprocesses per LADDIS load generator.

6. **Biod Emulation Parameters:** The 097.LADDIS benchmark emulates NFS client block-I/O daemon ("biod") read-ahead and write-behind functionality. A minimum of two outstanding read operations, established by the BIOD_MAX_READS parameter in the laddis_rc file, and a minimum of two outstanding write operations, established by the BIOD_MAX_WRITES parameter in the laddis_rc file, must be used per LADDIS load generator subprocess. This is the default behavior of the benchmark established by the default benchmark parameters.

7. **Time parameters:** RUNTIME, the time of measurement for which results are reported, must be the default 600 seconds for reportable results. The WARMUP_TIME must be set to the default of 60 seconds for reportable results.

8. **Isolated LADDIS network:** LADDIS should be run in an isolated network setup in order to obtain valid results. Results obtained on "production" networks are invalid as they will most likely not be reproducible. Such results are probably meaningless. Further, LADDIS may fail to converge to the requested load rate and behave erratically due to varying ambient load on the network.

9.8.2 Method

The method of using the benchmark tools and its menu driven user interface is described in detail in Section 8, "SFS Tools Interface".

The recommended method for running the SFS suite is described below. The benchmark may be run by any other mechanism that preserves the workload, its required parameters, concurrency, correctness and accuracy verification, and throughput measurements performed by the supplied makefiles. This is most easily accomplished using the supplied scripts.

9.8.2.1 Running 097.LADDIS from the $SPEC Directory

To run the SPEC SFS Release 1.1 suite you can use the shell script "runsfs", at $SPEC directory level. It calls an interactive menu-driven shell script, $SPEC/bin/runsfs, with the proper environment and shell. Using your responses, runsfs calls $SPEC/bin/laddis_mgr. The runsfs script guides you through the execution of the 097.LADDIS benchmark. The script prompts you for the information necessary to run the benchmark. Further, the script allows you to change only those benchmark parameters that are permitted to be modified by the SPEC Run and Reporting Rules for SFS Release 1.1. The runsfs script allows you to:

1. Display, create, or modify a vendor wrapper file.
2. Display, create, or modify a LADDIS parameter file (laddis_rc).
3. Compile the 097.LADDIS benchmark.
4. Remotely install LADDIS on the clients and mount/unmount target file systems.
5. Run the 097.LADDIS benchmark.
6. Display or archive benchmark results.

Runsfs may be aborted at any time. For more information refer to the runsfs script and the vendor wrapper files inside the 097.LADDIS benchmark directory.

9.8.2.2 Running 097.LADDIS Via the laddis_mgr Script

Once users gain a high degree of familiarity with the 097.LADDIS benchmark, a second method of running the benchmark may be used. The second method is to run the benchmark via the laddis_mgr script. If the results are to be reported, the only parameters that can be changed, as shown in Section 8.5, are LOAD, BIOD_MAX_WRITES,

BIOD_MAX_READS, NUM_RUNS, INCR_LOAD, CLIENTS, MNT_POINTS, SUT_FS_NAMES, SUTNAME, PRIME_SLEEP, and PRIME_MON_SCRIPT.

9.8.2.3 Location of Benchmark Results

When the runsfs or laddis_mgr script is used at the $SPEC level to run the 097.LADDIS benchmark, then benchmark results are located in $RESULTSDIR.

9.9 How to Handle and Report Problems

If you run the benchmark from the top level directory using the runsfs script, all the messages issued are saved in the $SPEC/output directory under the name of laddislog.id where <id> is a run identifier that you are asked to specify. This is the first place to check for possible causes for errors.

In addition, please refer to Section 11, "General Debug Information", for a list of common pitfalls and how to circumvent those.

If you run into errors, please contact SPEC using the phone number provided at the end of this document. You will be referred to a SPEC technical representative who will assist you in resolving the problem(s).

9.10 Recommendations for Reproducibility of Results

When attempting to reproduce the results reported by a specific results submitter, please ensure that your configurations, parameters, and makefiles are set exactly as specified in the submitter’s reported results. These details may be found in the the SPEC newsletter or requested from the submitter.

10. SPEC SFS RELEASE 1.1 REPORTING RULES

This section describes the rules for reporting the results of SPEC SFS Release 1.1 suite. This is a companion to "SPEC SFS Release 1.1 Run Rules", Section 9, which describes the process of running the benchmark contained in the SPEC Benchmark.

SPEC SFS Release 1.1 is a maintenance update for the 097.LADDIS benchmark. Following the release of SPEC SFS 1.1, all future SFS testing will use SPEC SFS Release 1.1.

The 097.LADDIS benchmark progressively stresses an NFS file server by increasing the NFS operation request rate (NFS load) of the NFS clients used to generate load on the server.

The performance metric is defined as the average NFS operation response time measured at a specified NFS load (NFS operations per second).

The NFS server’s performance is characterized in terms of a complete average NFS operation response time versus NFS throughput curve for a given server configuration. Also, the NFS capacity of the server in terms of NFS operations per second is reported at no higher than an average NFS operation response time of 50 milliseconds.

The reporting rules detailed in the following sections, as stipulated by the SPEC Open Systems Steering Committee (OSSC), are mandatory.

10.1 Reporting Guidelines

This section describes the standard SPEC reporting format that must be used when reporting SPEC SFS Release 1.1 results.

10.1.1 Metrics And Reference Format

The performance metric is the average NFS operation response time, in terms of milliseconds, for a given NFS load, in terms of NFS operations per second. The results of a benchmark run, comprised of several NFS load levels, are plotted on a performance curve on the results reporting page. The data values for the points on the curve are also enumerated in a table.

When referencing any point on the performance curve, the format "XXX SPECnfs_A93 NFS operations per second at YY Milliseconds average response time" must be used. If an abbreviated format is required, the format "XXX SPECnfs_A93 NFS ops./sec. @ YY Msec." must be used.

While all SPEC members agree that a full performance curve best describes a server's performance, the need for a single figure of merit is recognized.

The SPEC SFS single figure of merit is a triple which specifies:

1. NFS throughput at an average NFS response time no greater than 50 milliseconds.
2. The number of "SPECnfs_A93 USERS" determined by the algorithm:

$$\text{Number of SPECnfs\_A93 USERS} = \frac{\text{Maximum NFS Throughput @ <= 50 milliseconds}}{\text{10 NFS Operations/Second per SPECnfs\_A93 USER}}$$

The SPEC SFS single figure of merit is reported as a boxed item on the LADDIS performance graph on the reporting page:

<NFS Throughput> @ <Response Time>	==> XXX SPECnfs_A93 USERS
where:	
<NFS Throughput>	- is stated as "ZZZ SPECnfs_A93 NFS Ops./Sec."
<Response Time>	- is stated as "YY Msec."
==>	- is the logical implication operator symbol
XXX SPECnfs_A93 USERS	- is printed in a bold typeface

10.1.2 Reporting Format

10.1.2.1 Table Format

A table, from which the server performance graph is constructed, consists of a number of data points which are the result of a single run of the benchmark. The table consists of two columns, NFS Throughput in terms of SPECnfs_A93 NFS operations per second rounded to the nearest whole number on the left, and Average NFS Response Time in terms of Milliseconds rounded to the nearest tenth on the right. The data points are selected based on the criteria described in Section 10.1.2.2.

10.1.2.2 Graphical Format

NFS server performance is depicted in a plot with the following format:

1. Average Server Response Time in units of milliseconds is plotted on the Y-axis with a range from 0 to x milliseconds, where x obeys the run rule in 10.1.1, item 1.
2. The plot must consist of a minimum of 10 data points uniformly distributed across the range of the maximum server load. Additional points beyond these 10 uniformly distributed points also can be reported.
3. All data points of the plot must be enumerated in the table described in Section 10.1.2.1.
4. No data point within 25% of the maximum reported throughput may be reported whose "Actual NFS Mix Pcnt" versus "Target NFS Mix Pcnt" differs by more than 10% for any operation.

10.1.3 System Configuration

The system configuration information that is required to duplicate published performance results must be reported.

This list is not intended to be all-inclusive, nor is each feature in the list required to be described. The rule of thumb is: if it affects performance or the feature is required to duplicate the results, describe it.

10.1.3.1 Hardware

10.1.3.1.1 Server

The following server hardware components must be reported:

1. Vendor’s (Benchmark User’s) name
2. System model number, main memory size, number of CPUs
3. Critical customer-identifiable firmware or option versions such as network and disk controllers, write caches, or other accelerators
4. Number, type, and model of disk controllers
5. Number, type, model, and capacity of disk drives
6. Relationship among disk controllers and disk drives
7. Relationship among disk drives and filesystems
8. Number, type, and model of filesystem/NFS accelerators
9. Number, type, and model of network (Ethernet/FDDI) controllers
10. Number of networks and type
11. Number, type, model, and relationship of external network components to support server (e.g., external routers)
12. Alternate sources of stable storage including un-interruptible power supply systems (UPS), battery-backed caches, etc.

10.1.3.1.2 Load Generators

The following load generator hardware components must be reported:

1. System model number, main memory size, number of CPUs
2. Compiler used to compile benchmark
3. Number, type, model, and relationship of external network components

10.1.3.2 Testbed Configuration

A brief description of the system configuration used to achieve the benchmark results is required. The minimum information to be supplied is:

1. Relationship of load generators, load generator type, network, filesystem, and filesystem mount point options.
2. If the configuration is large and complex, added information should be supplied either by a separate drawing of the configuration or by a detailed written description which is adequate to describe the system to a person who did not originally configure it.

10.1.3.3 Software

The following software components must be reported:

1. Shipping OS version or pre-release OS version, deliverable within six months
2. Other clarifying information as required to reproduce benchmark results (e.g. number of NFS daemons, server buffer cache size, disk striping, non-default kernel parameters, etc.)
3. Number of load generators, number of processes per load generator, server filesystems targeted by each load generator
4. Number of BIOD_MAX_READ and BIOD_MAX_WRITEs used.

10.1.3.4 Notes/Summary of Tuning Parameters

This section is used to document:

1. Single or multi-user state
2. System tuning parameters other than default
3. Process tuning parameters other than default
4. Background load, if any
5. ANY changes made to the individual benchmark source code including module name, line number of the change.
6. Additional information such as compiler options may be listed here.
7. Additional important information required to reproduce the results, which do not fit in the space allocated above must be listed here.
8. A full description of the definition of tuning parameters used should be included as an auxiliary document similar to the tuning notes included with SPECint and SPECfp CPU benchmark results.

10.1.3.5 Other Required Information

The following additional information is also required to appear on the results reporting page for SPEC SFS Release 1.1 results:

1. General Availability of the System Under Test. All the system, hardware and software features are required to be available within 6 months of the date of test.
2. The date (month/year) that the benchmark were run
3. The name and location of the organization that ran the benchmark
4. The SPEC license number

11. GENERAL DEBUG INFORMATION

This section contains available information on how to debug at a very cursory level when problems occur while running the benchmark.

1. The frequently occurring errors are due to improper or insufficient benchmark setup. Please follow the setup procedures given in Section 5 closely.
2. The most common failure might give the following ambiguous error message:

   "rpc: program not available"

   This could be due to any of the following reasons:

   — The number of processes per client have reached a maximum beyond which the client cannot accommodate anymore. Reduce the number of processes per client and try running the benchmark again. Usually 2 - 4 processes per client, should be able to saturate the net.

   — The number of clients per net might have exceeded what is optimal. The ideal is from 2 - 4 clients per net.

   — One or more kernel network parameters on the server and or on client machines might have been set too low for proper network protocol functionality (refer Section 12, "Tuning for Better Performance").

   — The connectivity to the server might have been lost. Check the physical connectivity of the network and then cycle through a network stop and start.
3. Benchmark failures due to insufficient disk space on the shared server partition.

12. TUNING FOR BETTER PERFORMANCE

This section contains some hints that might be useful in obtaining the most appropriate 097.LADDIS results for the tested configuration. These are generally only suggestions, the reader is directed to the Run and Reporting Rules for specific details on how the benchmark must be run, and to the vendor documentation for information on how to obtain the best performance from the particular configuration.

Remember, our experience has shown that most of the effort is in getting client-server set up working. Section 7.4, "Failed Makefiles", and Section 11, "General Debug Information", may have some suggestions on making this go smoothly for you.

12.1 Basic Layout

You must supply sufficient network capacity in order to be able to obtain the maximum utilization of the server. A single Ethernet, for example, is only good for about 300 SPECnfs_A93 Operations per Second (we will refers to these as IOPS) approximately. Hence for example, if your server is expected to be capable of 750 IOPS then at least 3 Ethernets should be used. The same sort of rules apply to FDDI (estimated up to 3000 IOPS/net) and Token Ring has a limit too.

The run rules require at least two load generators per network. More may be desirable. Generally, you want to have fairly powerful clients. A variety of vendors offer suitable machines.

Setting up:

1. Remember to verify that your LADDIS client machines are configured with UDP checksums *enabled*. For some vendors workstations, this is not the default. This is required by the run rules. Check the server too.

2. You might find it easier to use some other tool than laddis to test your connectivity. Ping and rsh work well. A few scripts can go a long way towards testing connectivity. One suggested sequence is to ping the server, rsh a command, mount a filesystem, and create a file on the server from each client.

3. Use of a non-standard NFS operations mixfile with 100% "Null" operations can help to debug your test network and to help identify any clients with problematic transceivers, cables, interfaces, etc.

4. Perform some preliminary testing with a single client, requesting the same load but varying parameters such as PROCS and MAX_BIOD_READS and MAX_BIOD_WRITES. The load should be towards the high end that you expect such a client to be able to do. It may be the case that the context-switching overhead on a client might make it advisable to limit PROCS to a smaller value. It might also be the case that slower clients are disproportionately affected negatively by larger values of MAX_BIOD_{READS,WRITES}.

5. Based on the results obtained above, determine how many clients per network will saturate that network. The theoretical maximum number of SPECnfs_A93 NFS operations per second on a single Ethernet is approximately 330, but collisions and other factors typically result in a maximal *attainable* throughput of 300 IOPS on a single Ethernet. So, for example, if your clients produce optimal response time result at 150 IOPS Requested Load, then you can use the minimum of two clients per network as required by SPEC Run Rules. If your clients can only produce 50 IOPS optimally, then you will want at least five or six per network. N.B. Adding more than 2 clients per Ethernet may cause problems!

Getting the most from your server:

Once you have things pretty much set up and configured as discussed above, it’s time to focus on the server. A variety of tuning opportunities exist for realizing the optimum server performance.

1. Check your vendor documentation. Tuning tips are available from most vendors. Check on available patches.
2. Often the vendor’s own efforts are the best source for LADDIS tuning information. If they exist, obtain the SPEC SFS 1.1 results pages for your server. The notes section will usually contain details on what configuration parameters, etc. work well.
3. Some vendors offer servers with multiple bus configurations, check vendor recommendations for board placement to ensure the buses are well balanced.
4. Generally, the usual sort of file system I/O tuning is effective: balanced disk utilization, use of buffer caches for both data and metadata, etc. This is a very I/O intensive benchmark.

5. Don’t forget to check out your networks. Sometimes increasing the size of some network buffers is necessary.
6. And NFS: you may want to increase the number of NFS daemons (nfsd) that are running on your system.
7. Review previously submitted results by other vendors to determine configurations and tunings they have made before submitting results.

   In particular, look at the amount of memory, number of disks and disk controllers, and number and type of networks. This information is invaluable in helping you specify an initial configuration to test.
8. Pay attention to the number of ops/sec per net. Remember Ethernet cannot support more than about 300 ops/sec per net or about 3200 ops per FDDI. Depending on your Ethernet hardware, this number may be lower.
9. There are usually three important caches to consider when tuning your server:

   — the name lookup cache

   — the inode (metadata) cache

   — the buffer (data) cache

   Large name caches and inode caches will increase your hit rate and reduce disk I/O. A large buffer data cache will help reduce disk accesses needed to service read requests.

   By far, the most inexpensive improvement to be made is in increasing your name and inode cache.
10. There are typically three potential bottlenecks in your system: number of networks, CPU, and disk (disks or controllers).

    If your CPU utilization is not 100% while under heaviest requested load, there are several things that may be wrong. You may not have enough load generating processes on the clients, and you may be starving your server. Increase the number of load generators. If on increasing the number of load generators, your throughput remains the same or drops and your response time increases, then the number of load generators was not the problem.

    If you are requesting more than 300 ops/sec per Ethernet or 3200 ops/sec per FDDI (for the standard mix and default parameters), then you may be network bound. Depending on your server networking hardware and server architecture, you may actually service less than 300 (or 3200) ops/sec per net.

    Do not rule out that your clients may be underpowered and that they may not be adequate to generating the requested load. Run a single client test on an isolated network segment to determine your client’s load generating capacity.

    If your CPU utilization is still low, then you may have an insufficient number of disks to service the requested load. Check your cache sizes. Even modest servers can approach 100% hit rates on name lookup and inode caches, reducing disk requirements. Adding memory to your server, and increasing the size of the buffer cache (if this is a tunable parameter on your server) will further reduce the load on your disk subsystem in many cases.

    Add enough disk drives to your system to allow a suitable number of parallel paths to your file systems. For example, if a disk can handle about 40 disk operations per second, then you need from 15 - 20 disks per 1000 NFS ops/sec.

    Ensure that the clients are correctly and uniformly accessing the file systems, and the file systems are uniformly distributing the load over the disks.

    If your CPU utilization is 100%, add another CPU.

13. READING DOCUMENTS

The principal documents for SPEC SFS Release 1.1 are three: this manual, the laddis(1) manpage and the USENIX White Paper entitled "LADDIS: The Next Generation In NFS File Server Benchmarking".

The USENIX White Paper is include on the SPEC SFS Release 1.1 distribution tape. The White Paper provides an excellent discussion of the technical issues involved in developing 097.LADDIS and the rationale used to establish default 097.LADDIS workload parameter values. It is in postscript format and must be processed by a postscript printer.

The online manpage for 097.LADDIS, "$BENCH/src/laddis.1", gives a very thorough description of the laddis command and its options.

This manual is found in postscript format in the subdirectory labeled **documents**. If you wish to obtain a typeset copy using the local copies of nroff/troff/?roff, then follow this guideline:

```
tbl USER_MANL.mm | nroff -mm -Tlp | col -b -x > USER_MANL
```

This preprocesses the table commands embedded in the document, runs the output through nroff using the System V -mm macros (Memorandum Macros) and directs nroff to format the output for a dumb line printer that can understand reverse column motions (but not ANSI escape sequences), this is then passed to the col command which strips out all reverse line feeds and backspaces and converts all tabs to white space.

14. CONTACTING SPEC

Questions, suggestions or comments, can be directed to SPEC via:

Systems Performance Evaluation Corporation (SPEC)
c/o NCGA
2722 Merrilee Drive, Suite 200
Fairfax, VA 22301

Phone: 703-698-9600 x318
FAX: 703-560-2752

Inquiries will be directed to an appropriate SPEC representative. SPEC encourages and appreciates feedback concerning the SFS Release 1.1 Benchmark and other SPEC benchmark suites.

(NOTE: Documentation changes are possible and probably desirable. Please send in your comments on the documentation included in this suite to the above address.)

CONTENTS

--   --