
Understanding and Using ServerBench® 3.0

Copyright © 1996 by Ziff-Davis Publishing Company
All rights reserved.

Manual release date: April 1996 with version 3.0 of ServerBench

LICENSE AGREEMENT FOR ZIFF-DAVIS' SERVERBENCH® VERSION 3.0

READ THIS AGREEMENT CAREFULLY BEFORE USING THE SOFTWARE EMBODIED IN THE SERVERBENCH® 3.0 CD-ROM, DISKETTE(S), OR, IF PRELOADED ON YOUR HARD DRIVE, DOWNLOADED OR IF PROVIDED AS PART OF A COLLECTION, THE PRELOADED, DOWNLOADED OR COLLECTED FILE(S) (the "Media"). Embodied in the ServerBench 3.0 Media is the ServerBench version 3.0 computer programs and related documentation (the "Software"). Ziff-Davis Publishing Company, having a place of business at One Park Avenue, New York, New York 10016 ("Ziff-Davis") is the licensor under this Agreement and you are the licensee. By using the Software, in whole or in part, you agree to be bound by the terms of this Agreement. If you do not agree to the terms of this Agreement, promptly return the Software to the Ziff-Davis Benchmark Operation at 1001 Aviation Parkway, Suite 400, Morrisville, North Carolina 27560 (or, if downloaded, delete the Software, or if preloaded on your hard drive or if provided as part of a collection, cease use of the Software). Title to the Software and all copyrights, trade secrets and other proprietary rights therein are owned by Ziff-Davis. All rights therein, except those expressly granted to you in this Agreement, are reserved by Ziff-Davis.

1. Limited License

This Agreement grants you only limited rights to use the Software. Ziff-Davis grants you a non-exclusive, non-transferable license to use the Software on a file server networked with multiple PC computers for the sole purpose of conducting benchmark tests to measure the performance of computer hardware and operating system configurations. You have the right to make a single copy of the Software for archival purposes and the right to transfer a copy of the Software across an internal local area network only to the PC computers attached to such network; provided, however, that all such copies are considered Software hereunder, that all uses of such copies are governed by the terms and conditions of this Agreement and that you shall be responsible for all uses of such copies in violation of the terms and conditions of this Agreement. Ziff-Davis hereby grants you the right to publish, except in any country where a third party claims during the term of this license that such distribution infringes that party's proprietary rights, benchmark test results obtained by you from your use of the Software provided that with the publication of each such result you:

- A. Identify Ziff-Davis, the name and version number of the benchmark Software used and the name of the test suite used in the test (e.g., Ziff-Davis' ServerBench® version 3.0 with the standard system test suite SYS_60.TST);
- B. Identify for the applications server the exact name, number of processors, processor speed (including clock speed) and type(s), amount of RAM, amount of secondary RAM cache (if any), size of hardware hard disk cache, (if any), number and type of hard disk controller(s), type of I/O bus, number and type of hard disks, hard disk size, driver version of the disk controller(s), type of disk organization (e.g., mirrored), number and type of network controllers, driver version of network controller, network operating system name and version, and any relevant modifications to the default network operating system parameters (e.g., WXY Corp. Model 466 with 1 66-MHz Intel® 486DX2-66 CPU, 64MB of RAM, 256KB CPU cache, WXY Integrated Drive Array disk controller, EISA I/O bus, 2 ABC 520MB disk drives, a xxxxx386.dsk 12,621 4/29/92 disk driver, hardware striping, WXY 32-bit ABC controller, a xxxxxxx.lan 40,905 9/11/92 net driver, NetWare® 4.1, and the following NOS parameters: set maximum physical receive packet size = 4202);
- C. Identify for the test bed the network type, the number of clients, the client operating system version (e.g., Windows® 95), the number and type of hubs/concentrators, the number of clients per segment, the client CPU type and speed in percentages, client network software name and version (drivers and protocols), the size of the client network cache, if any (e.g., 10Base-T with 32 clients, 2 XYZ Ethernet Hubs, 16 clients per segment, 75% of the clients are 486/25 and 25% of the clients are 386/20, 75% of the clients have ABC NE2000 network cards and 25% of the clients have GHI network cards, Microsoft® Windows 95, Microsoft TCP/IP using Windows Sockets Version 1.1, enhanced mode 32-bit NDIS driver);

- D. Identify the controller operating system version (e.g., Microsoft Windows 95) and network software and version;
- E. State that all products used in the test were shipping versions available to the general public;
- F. State that the test was performed without independent verification by Ziff-Davis and that Ziff-Davis makes no representations or warranties as to the results of the test; and
- G. Follow proper trademark usage and acknowledge Ziff-Davis' trademark rights (e.g., "[] achieved a ServerBench® overall score of X transactions per second. ServerBench is a registered trademark or trademark of Ziff-Davis Publishing Company in the U.S. and other countries.").

This Agreement and your rights hereunder shall automatically terminate if you fail to comply with any provision of this Agreement. Upon such termination, you agree to cease all use of the Software, cease the transfer of any copies of the Software and cease the publication of benchmark test results obtained by you from use of the Software. Further, you agree to delete the Software and to destroy all tangible copies of the Software and other materials related to the Software in your possession or under your control, or, if downloaded or preloaded on your hard drive or if provided as part of a collection, to cease use of and destroy any and all copies of the Software in your possession or under your control.

2. Additional Restrictions

- A. You shall not (and shall not permit other persons or entities to) rent, lease, sell, sublicense, assign, or otherwise transfer the Software or this Agreement. Any attempt to do so shall be void and of no effect.
- B. You shall not (and shall not permit other persons or entities to) reverse engineer, decompile, disassemble, merge, modify, include in other software or translate the Software, or use the Software for any commercial purposes, except for the publication of test results, as provided above.
- C. You shall not (and shall not permit other persons or entities to) remove or obscure Ziff-Davis' copyright, trademark or other proprietary notices or legends from any of the materials contained in this package or downloaded.
- D. You acknowledge that the Software contains Ziff-Davis' trade secret information and you agree not to disclose or disseminate such information other than as provided herein.

3. Limited Warranty and Limited Liability

THE SOFTWARE IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE RESULTS AND PERFORMANCE OF THE SOFTWARE IS ASSUMED BY YOU, AND ZIFF-DAVIS AND ITS AUTHORIZED DISTRIBUTORS ASSUME NO RESPONSIBILITY FOR THE ACCURACY OR APPLICATION OF OR ERRORS OR OMISSIONS IN THE SOFTWARE. IN NO EVENT SHALL ZIFF-DAVIS OR ITS AUTHORIZED DISTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE SOFTWARE, EVEN IF ZIFF-DAVIS OR ITS AUTHORIZED DISTRIBUTORS HAVE BEEN ADVISED OF THE LIKELIHOOD OF SUCH DAMAGES OCCURRING. ZIFF-DAVIS AND ITS AUTHORIZED DISTRIBUTORS SHALL NOT BE LIABLE FOR ANY LOSS, DAMAGES OR COSTS, ARISING OUT OF, BUT NOT LIMITED TO, LOST PROFITS OR REVENUE, LOSS OF USE OF THE SOFTWARE, LOSS OF DATA OR EQUIPMENT, THE COSTS OF RECOVERING SOFTWARE, DATA OR EQUIPMENT, THE COST OF SUBSTITUTE SOFTWARE OR DATA, CLAIMS BY THIRD PARTIES, OR OTHER SIMILAR COSTS.

THE ONLY WARRANTY MADE BY ZIFF-DAVIS AND ITS AUTHORIZED DISTRIBUTORS IS THAT ANY ORIGINAL PHYSICAL MEDIA IN WHICH THE SOFTWARE IS EMBODIED AND WHICH IS DISTRIBUTED BY ZIFF-DAVIS OR ITS AUTHORIZED DISTRIBUTORS SHALL BE FREE OF DEFECTS

IN MATERIALS AND WORKMANSHIP. ZIFF-DAVIS' AND ITS AUTHORIZED DISTRIBUTORS' ENTIRE LIABILITY AND THE USER'S EXCLUSIVE REMEDY SHALL BE LIMITED TO THE REPLACEMENT OF THE ORIGINAL PHYSICAL MEDIA IF DEFECTIVE. THE WARRANTIES AND REMEDIES SET FORTH HEREIN ARE EXCLUSIVE AND IN LIEU OF ALL OTHERS, ORAL OR WRITTEN, EXPRESS OR IMPLIED. NO ZIFF-DAVIS DISTRIBUTOR, AGENT OR EMPLOYEE, OR THIRD PARTY, IS AUTHORIZED TO MAKE ANY MODIFICATION OR ADDITION TO THIS WARRANTY.

SOME STATES DO NOT ALLOW EXCLUSION OR LIMITATION OF IMPLIED WARRANTIES OR LIMITATION OF LIABILITY FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES; SO THE ABOVE LIMITATIONS OR EXCLUSIONS MAY NOT APPLY TO YOU.

4. U.S. Government Restricted Rights

The Software is licensed subject to RESTRICTED RIGHTS. Use, duplication or disclosure by the Government or any person or entity acting on its behalf is subject to restrictions as set forth in subdivision (c)(1)(ii) of the Rights in Technical Data and Computer Software Clause at DFARS (48 CFR 252.227-7013) for DoD contracts, in paragraphs (c)(1) and (2) of the Commercial Computer Software-Restricted Rights clause in the FAR (48 CFR 52.227-19) for civilian agencies, or in the case of NASA, in Clause 18-52.227-86(d) of the NASA Supplement to the FAR, or in other comparable agency clauses. The contractor/manufacturer is the Ziff-Davis Benchmark Operation, 1001 Aviation Parkway, Suite 400, Morrisville, North Carolina 27560.

5. General Provisions

Nothing in this Agreement constitutes a waiver of Ziff-Davis rights under U.S. copyright laws or any other Federal, state, local or foreign law. You are responsible for installation, management, and operation of the Software. This Agreement shall be construed, interpreted and governed under New York law. If any provision of this Agreement shall be held by a court of competent jurisdiction to be illegal, invalid or unenforceable, the remaining provisions shall remain in full force and effect.

Trademarks

MacBench®, NetBench®, ServerBench®, WinBench®, and Winstone® are registered trademarks and PC Bench™ and ZD Net™ are trademarks of Ziff-Davis Publishing Company.

CompuServe® is a registered trademark of CompuServe, Inc.

Digital™ and Alpha™ are trademarks of Digital Equipment Corporation.

Intel® and Pentium® are registered trademarks of Intel Corporation.

Microsoft®, MS-DOS®, and Windows® are registered trademarks and Windows NT™ is a trademark of Microsoft Corporation.

NetWare® and Novell® are registered trademarks of Novell, Inc.

MIPS® is a registered trademark of MIPS Technologies, Inc.

OS/2® is a registered trademark and PowerPC™ is a trademark of IBM Corporation.

PC/TCP® is a registered trademark of FTP Software, Inc.

SCO® and UnixWare® are registered trademarks and SCO OpenServer™ is a trademark of The Santa Cruz Operation, Inc.

UNIX® is a registered trademark in the United States and other countries, licensed exclusively through X/Open Company, Ltd.

Table of Contents

About ServerBench's Documentation.....

- Here's how we document ServerBench.....
- Conventions ServerBench uses.....
- Conventions this manual uses.....
- Contacting ZDBOp.....

About ServerBench: The Salient Points.....

- Here's what ServerBench is.....
- The ServerBench test setup.....
- Here's what comes with the ServerBench benchmark.....
- Here's what doesn't come with ServerBench.....
- The results ServerBench produces.....
- The secret to the ServerBench test suites.....
- The ServerBench individual tests.....
- Repeatability — isn't it great.....
- Don't forget to license your copy of ServerBench.....
- If you want to run ServerBench, here're two tips.....

What's New in ServerBench 3.0.....

- More server platforms.....
- You can have Windows 95 clients/controller.....
- Support for Winsock-compliant TCP/IP stacks.....
- Unattended mode adds error-handling features.....
- External Notification.....
- You can replace a client that terminates.....
- Enhanced results reporting features.....
- The standard test suites are now even more “stressful”.....
- New features for creating mixes.....
- Other new ServerBench 3.0 features.....

Part 1 Getting to Know ServerBench.....

Chapter 1 Introducing ServerBench.....

ServerBench in brief.....	
The ServerBench basics.....	
Using test suites to measure performance.....	
The ServerBench test environment.....	
A quick look at how ServerBench runs.....	
The results ServerBench produces.....	
Why you should run ServerBench.....	
Forget those results from earlier versions.....	
Using ServerBench with NetBench®.....	

Chapter 2 How ServerBench Works.....

The foundation for ServerBench.....	
The subsystems ServerBench stresses.....	
Here's where ServerBench fits in.....	
The ServerBench programs and how they work.....	
The basic provider model.....	
Sample provider model.....	
A typical test scenario.....	

Chapter 3 The Concepts Behind ServerBench.....

The design goal.....	
The advantage of having a synthetic benchmark.....	
Providing an easy-to-use benchmark for different servers.....	
Here's what client/server means.....	
The limits of testing an applications server.....	
How the subsystems affect each other.....	
The benefits of using stress tests.....	
One client does the work of several users.....	
Different clients put different loads on the server.....	
Creating tests that model real users' work.....	
Measuring the results using TPS.....	
The role cache plays.....	
Factoring in multitasking.....	
Figuring out the knees in the results curves.....	
How to evaluate throughput in terms of your server.....	
System throughput versus client response time.....	
Processor scaling.....	
Standard deviation and variance in scores.....	
Using a weighted harmonic mean.....	
Making repeatability a reality.....	

Chapter 4 The ServerBench/NetBench Connection.....

- Choosing ServerBench over NetBench.....
- Running both ServerBench and NetBench.....
 - Starting and using ServerBench and NetBench.....

Part 2 The Mechanics of Using ServerBench.....

Chapter 5 Using ServerBench's Windows and Menus.....

- The main ServerBench window.....
- The controller window.....
 - The client grid and color legend.....
 - The test suites box and test status information.....
 - The controller window's function buttons and menus.....
 - The Test Suite History window.....
 - The Unattended Mode dialog box.....
 - The External Notification dialog box.....
- The test suite selection windows.....
 - Duplicate results names.....
- The results dialog boxes.....
 - The View Results dialog box.....
- The creating and editing test suites screens.....
 - Create or Edit Test Suites dialog box.....
 - The Mixes in Suite dialog box.....
 - Adding existing mixes to a test suite.....
 - The Reorder Suite Mixes dialog box.....
 - The Mix Definition window.....
 - Duplicating mix fields across a test suite.....
 - A word of caution about duplicating mix fields.....
 - Copying mix information into the Mix Definition window.....
- The Client window.....

Chapter 6 Starting, Stopping, and Interrupting ServerBench

- Starting ServerBench.....
 - Starting ServerBench on the controller.....
 - Starting ServerBench on the server.....
 - Starting ServerBench on the clients.....
- Disconnecting the server and clients.....
- Exiting from ServerBench.....
- Interrupting a running test suite.....
- Using ServerBench's on-line Help.....

Chapter 7 Quickly Running a Sample Test Suite.....

- Running ServerBench's sample test suite.....
- Saving the results.....

Part 3 Executing ServerBench's Test Suites.....

Chapter 8 Running ServerBench's Test Suites.....

- Before you run your test suites.....
- Allowing enough time to run a test suite.....
- Setting up error-handling features.....
 - Setting ServerBench up to notify you of an error.....
- Selecting and running a test suite.....
 - Adding test suites you want to run.....
 - Removing a test suite from the list of test suite files.....
- Using the pause option.....
- Running the test suites.....
- Monitoring a test.....
 - The different client stages.....
 - Checking the status of all the selected test suites.....
- Running additional test suites.....
- Replacing a client that terminates.....
- Disconnecting the server and clients.....

Chapter 9 Getting the Best Scores Possible.....

- Here's how we test ServerBench.....
- Here's what we recommend you do.....
- Some general ServerBench testing tips.....
 - Keep the background activity down.....
 - Use a test network, not a production network.....
 - Other ways to improve ServerBench results.....
- How hardware and software affect results.....
 - A look at what affects each server subsystem.....
 - Factors related to your system's hardware and software.....
- Take a look at your test parameters.....
 - Record your results while the server in a steady state.....
- Determining what to test on your system.....
- The best score isn't always the right score.....
- Creating test mixes to stress your server.....

Chapter 10 Creating Your Own Test Suites.....

- Developing test suite files.....

- Designing a test mix.....
- Creating a test suite: the basic steps.....
- Specifying a file name for your test suite.....
- Creating or editing a mix.....
 - Specifying the Mix Name.....
 - What the Mix Name does.....
 - Entering the duration information for a mix.....
 - What the Ramp up, Ramp down, and Length parameters do...
 - Choosing values for Ramp up, Ramp down, and Length.....
 - Determining how long an iteration takes.....
 - Specifying the timing information.....
 - What the Delay and Think time parameters do.....
 - Defining transactions.....
 - The basic steps for creating a transaction.....
 - How the parameters affect the transactions.....
 - Here's how the I/O range parameter works.....
 - Some different types of transactions.....
 - Providing the basic client information.....
 - How to enter the client information.....
 - Including client groups.....
 - How the client groups feature works.....
- Using another mix to set up the current mix.....
- Copying mix fields across a test suite.....
- Copying mixes from an existing test suite.....
- Inserting new mixes into a test suite.....
- Reordering mixes.....
- Deleting mixes from a test suite.....

Chapter 11 The Details on ServerBench's Tests.....

- Understanding ServerBench's tests.....
- The Processor test.....
- The disk tests.....
- The network tests.....
- The standard ServerBench test suites.....
 - Looking at SAMPLE.TST.....

Part 4 Checking Out ServerBench's Results.....

Chapter 12 Let's Take a Look at Those Results.....

- Viewing test results.....
 - Here's how to display ServerBench's results workbooks.....
- Using the results workbook.....
 - Summary report.....
 - Table 1.....
 - Overall Data.....
 - Table 2.....
 - Client data.....
 - The Disclosure report.....
 - Table 4.....
 - The information in Table 5.....
 - The Suite Definition reports.....
 - Table 6.....
 - Table 7.....
- Creating graphs of multiple results.....
- Using your disclosure database.....
 - Creating your first database snapshot file.....
 - Updating server disclosure information.....
 - Updating client disclosure information.....
 - Selecting a client or clients.....
 - Editing the client information.....
 - Updating or adding clients to a snapshot file.....

Chapter 13 Figuring Out What Your Results Mean.....

- Looking for the main score.....
- The measure of a ServerBench result.....
- Keeping track of ServerBench's results.....
- Making ServerBench's results count.....
- Looking at a graph of server's performance.....
- Determining just what your scores mean.....
- Checking for uneven service.....
- Checking for bottlenecks.....
- The process for testing your system.....
- How to tell if your results are abnormal.....
- Here's some background on the results.....
 - The individual tests ServerBench uses.....
 - The standard ServerBench test suites.....
 - Deciding which result is really the best for you.....

How ServerBench calculates scores.....
Comparing your server's results.....

Chapter 14 Publishing Your ServerBench Results.....

If you want to publish results, read this.....
 Sample ServerBench disclosure sheet.....
 ServerBench terms and measurements.....

Part 5 If You Have a Question.....

Chapter 15 Contacting ZDBOp.....

Look here for possible solutions to problems.....
Here's how you get in touch with ZDBOp.....
 Reporting a problem.....
If you want to get a copy of a benchmark.....
 Getting copies of benchmarks from ZD Net.....
 Requesting the benchmark CD-ROMs from ZDBOp.....

Chapter 16 Frequently Asked ServerBench Questions.....

Glossary.....

Index.....

Problem Report Form for ServerBench

Benchmark Request Form

Acknowledgments

About ServerBench's Documentation

This manual tells you what you need to know to run and understand ServerBench. ServerBench is a portable client/server benchmark you can use to measure applications servers running on multiple server operating system platforms (for a complete list of which server platforms ServerBench runs on, see the **README.TXT** file in the **SB30** directory on the Server Benchmarks CD-ROM and in the ServerBench library on ZD Net™, an on-line service of Ziff-Davis).

While the installation process varies slightly for the different operating systems, the way you use ServerBench doesn't. That's where this manual comes in. It tells you which buttons to click on to start ServerBench, where to look for test results, and all the details in-between. After reading this manual, you should know just about everything there is to know about what makes ServerBench tick. You'll learn:

- What ServerBench is.
- How you execute it.
- What concepts went into designing it.
- How to interpret its results.
- What affects its results.
- How the standard test suites work.
- How to create your own test suites.
- What you need to do if you want to publish your ServerBench results.
- Where to go if you need help.

What you won't find in this manual is a set of installation instructions. See the on-line installation manual that comes with your server platform for information on how to install ServerBench on your system.

NOTE: You'll see the pronoun "we" used throughout the rest of this manual. "We" refers to either the Ziff-Davis Benchmark Operation (ZDBOp) or Ziff-Davis in general.

Here's how we document ServerBench

ServerBench is a big benchmark that works on numerous different server platforms. As a result, there's a lot of documentation that goes with ServerBench. The documentation that comes on the Server Benchmarks CD-ROM consists of:

- **README.TXT** files. We include a **README.TXT** file for each server platform as well as for other directories on the CD-ROM. (We also include a **README.TXT** file on ZD Net.)
- A generic users' reference manual. This manual, which you are currently reading, applies to all server platforms of ServerBench.
- A quick start handbook. This manual is a short manual that focuses on quickly running ServerBench and looking at your test results. It also steps you through the process of creating a test suite.
- Server platform manuals. We include an on-line manual for each server operating system we port ServerBench to. This manual tells you how to install ServerBench on that system and provides port-specific tips for using ServerBench.
- On-line Help files. We include on-line help files that you can view from the ServerBench controller PC.

We also supply an on-line FAQ (frequently asked questions) file for ServerBench on ZD Net.

The next sections give you the details about the documentation we provide for ServerBench.

README.TXT

If you get ServerBench on the Server Benchmarks CD-ROM, you'll actually see several **README.TXT** files. The CD-ROM contains the following **README.TXT** files:

- **\README.TXT**
Describes the contents of the CD-ROM. Since this is the Ziff-Davis Server Benchmarks CD-ROM, it contains all the ServerBench platforms and Ziff-Davis' benchmark for file servers, NetBench®.

- **\SB30\README.TXT**
Explains how to find the files you'll need to install ServerBench on your server platform. This file also points you to the appropriate **README.TXT** file for your server operating system.

These are all text files you can view with any text editor. The **README.TXT** you will find most useful is the one in your server platform's directory. The **\SB30\README.TXT** file contains a directory tree that shows you how to locate the platform specific **README** files. (If you downloaded ServerBench from ZD Net, then you'll need to expand the controller executable to get to the platform-specific **README.TXT** file.)

The **README** files contain the ServerBench License Agreement, instructions on how to locate the ServerBench installation manual for your server platform, and any late-breaking information that did not make it into the ServerBench manuals. You should read this information before you install or run ServerBench.

Also, when you install the ServerBench files on the controller you'll find another copy of the **README.TXT** file in the ServerBench installation directory. This is the **README** file for your particular ServerBench port.

Understanding and Using ServerBench® 3.0

This is an on-line copy of this manual. It is in the **DOCS** subdirectory on the Server Benchmarks CD-ROM. Each server platform has a **DOCS** subdirectory.

This manual is also in the **DOCS** subdirectory that ServerBench creates when you install it on the controller (the controller is the PC that you run ServerBench from and monitor the tests).

On the CD-ROM, we provide the manual in three different formats, so you should be able to print a copy:

- **USING.DOC** is the Microsoft Word for Windows™ 6.0a version of the manual. If you print this file, you'll get a nicely formatted copy of the manual.

NOTE: This is the only format of the manual you'll find in the **DOCS** directory on the controller. That directory does not contain **.RTF** or **.TXT** versions of the manual.

- **USING.RTF** is the Rich Text Format (RTF) version of the manual. If you don't have Word for Windows, you may be able to read this file into a word processor that accepts RTF and print the manual that way. You'll lose some formatting when you import the **.RTF** file, but you'll still get all the text and possibly the screen captures, depending on your application's graphics abilities.
- **USING.TXT** is an unformatted text version of the manual. This one's not very pretty, but if you can't use either of the other versions, you should be able to print this ASCII file from DOS.

The ServerBench® Quick Start Handbook

This is an on-line manual that helps you run ServerBench's standard test suites, look at your results, and even create a quick test suite of your own. It focuses on quickly stepping you through some of the basic ServerBench tasks. This manual applies to all ServerBench platforms.

You'll find a copy of this manual in the **DOCS** subdirectory for its server platform on the Server Benchmarks CD-ROM and in the **DOCS** subdirectory that ServerBench creates when you install it on the controller.

On the CD-ROM, we provide the manual in three different formats, so you should be able to print a copy:

Installing ServerBench® 3.0 on your server platform

ServerBench provides an on-line manual for each supported server operating platform. This manual tells you how to install ServerBench on your system. It also includes any port-specific information and tips for using ServerBench.

Each of these manuals is in the **DOCS** subdirectory of the corresponding server platform on the Ziff-Davis Server Benchmarks CD-ROM and in the **DOCS** subdirectory that ServerBench creates when you install it on the controller.

On the CD-ROM, we provide these manuals in three different formats, so you should be able to print a copy:

ServerBench's on-line Help

ServerBench provides on-line Help that contains most of the information in this manual. You can access ServerBench's on-line help from most ServerBench windows on the controller.

If you are at the main ServerBench window and you choose the drop-down Help menu, ServerBench gives you the option of choosing:

- **General.** This starts the primary on-line Help, which contains most of this manual.
- **Port Specific.** This starts another version of Help, which contains most of the information in the on-line ServerBench manual for your server platform.
- **About.** This displays ServerBench's About screen. From this window you can choose to re-read the ServerBench License Agreement or to view the Credits, which list the names of the members of the ServerBench development and documentation teams.

The FAQ file

If you look on ZD Net, you'll notice that we also have an on-line FAQ file for ServerBench. This file contains answers to some of the questions we get and any late-breaking tips we have. This one file has both general ServerBench information and information about the different server platforms. We do not have a separate FAQ file for each server platform ServerBench runs on.

Conventions ServerBench uses

Because you run the ServerBench tests from the controller, which must have Windows, ServerBench takes advantage of common Windows features. You benefit from this approach because you don't have to learn new ways to do things and select items.

The Windows features that ServerBench uses include:

- Clicking and double-clicking with your mouse's left button to select and activate items. (If you are using keyboard commands instead of a mouse, you will need to enter the corresponding keyboard command when you see the instruction to click or double-click on an item.) Generally, if you click on an item, such as an icon or a machine name in a list, you are selecting that item and ServerBench highlights it. If you double-click on an item, you are choosing that item and ServerBench takes an action associated with it. For example, if you double-click on the ServerBench icon in the Ziff-Davis Benchmarks program group on the controller, the ServerBench program starts and the main window appears.
- Using drop-down menus from the Windows main menu bar. ServerBench lists some of its options as menu titles in the Windows menu bar in the ServerBench controller interface. When you select one of these menu titles, such as Help, a drop-down menu appears displaying the menu items available. In many cases, the drop-down menus duplicate actions you can perform in the main window. In some cases, the drop-down menus give you additional options, such as getting information on how to contact ZDBOp.
- Moving windows. You can minimize, maximize, and move some of ServerBench's windows the same way that you work with your windows now.
- Switching between windows on the controller. You can switch windows on the controller by clicking on an inactive window to make it active or by pressing ALT-TAB.
- Using keyboard commands to select some items instead of using the mouse. Press the ALT key and the underlined access (or shortcut) character of the item you want to select. This item can be one of the ServerBench function buttons, a menu title, or a menu item within a drop-down menu. For example, to select the Start Test option in the File menu, press the ALT key, the f key, and then the s key.

NOTE: You can only access some of the ServerBench functions without a mouse. Other functions will require a mouse.

Conventions this manual uses

This manual uses a few general conventions. These are:

- Using the pronoun "we" to refer to members of the Ziff-Davis Benchmark Operation team or members of Ziff-Davis in general.
- Displaying directory names and file names in bold, such as saying **\SB30\CONTROL\SUITES** to refer to a subdirectory that contains test suite files.
- Displaying in monospace or typewriter font information that appears on your screen or that you enter, such as:

```
MAKEDISK A:
```

- Using italics to indicate variables that you must supply a value for. For example:

```
client client_name
```

indicates that you should enter the name of the client after the command `client`.

- Using the term *choose* to indicate when you need to perform some sort of operation on an item, such as double-clicking on an item. This manual uses the term *select* to indicate when you need to highlight an item, which usually occurs when you single-click on it. (You can use either your PC's mouse or keyboard commands to execute ServerBench from the controller.)

For example, when you select the ServerBench icon, you click on it once. This highlights the icon. When you choose the ServerBench icon, you click on it twice. This both highlights and opens the icon.

- Referring to the main ServerBench window, the controller window, and the client window.

The *main ServerBench window* is the window that first appears on the controller when you start ServerBench. This window contains six function buttons.

The *controller window* is the window that appears when you choose the function button Start Test from the main ServerBench window. You use this window to actually start tests on ServerBench and to monitor the progress of the tests. The grid in this window shows the state of all the clients who are connected to the server for the ServerBench test.

The *client window* is the window that appears on each client. This window displays information about the client and the client's current state in the test. You monitor the clients' status while ServerBench is running its test. You do not enter input from this interface.

Contacting ZDBOp

You can use ZD Net, a Ziff-Davis on-line service, to ask questions about or comment on ServerBench. You can also mail or fax your questions and comments to ZDBOp. Chapter 15 "Contacting ZDBOp" contains details on the different ways to contact ZDBOp.

End of "About the Documentation"

About ServerBench: The Salient Points

This is a brief introduction to ServerBench. It contains a summary of the key points about ServerBench. We explain these points in detail later in this manual.

Here's what ServerBench is

ServerBench version 3.0 is a Ziff-Davis benchmark program that lets you measure the performance of application servers in a client/server environment. It provides you with an overall score for your server and individual scores for the clients.

ServerBench lets you test different servers as they provide a variety of services to PCs running Windows 95 or Windows for Workgroups. These PCs are called clients. To accurately represent this client/server interaction, we modeled ServerBench after an application server environment. This means your data and your applications exist on the server and the client PCs are simply a front-end to provide you with an access point into the applications. This is the case with a database transaction processing system, where all the data resides on the server and the clients serve as front ends to access the data.

To execute ServerBench, you must run three different ServerBench programs: a server program that resides on your server, a controller program that resides on the Windows-based PC you designated as the controller, and a client program that resides on each of the Windows-based PCs designated as a client. ServerBench uses only these programs to measure your server; it does not use other server applications.

The ServerBench test setup

You must have at a minimum three machines to run ServerBench:

- A server. ServerBench currently runs on several server operating systems and we're in the process of porting it to more systems. (For a list of ServerBench 3.0's currently supported ports, see the **\SB30\README.TXT** file on the Ziff-Davis Server Benchmark's CD-ROM.)

- A PC running Windows 95 or Windows for Workgroups 3.11. This is the controller. ServerBench does not run tests on the controller.
- Additional PCs running Windows 95 or Windows for Workgroups 3.11. These are the clients who send requests to the server. You can have up to 1,000 clients; however, because ServerBench uses stress tests, you can get accurate results with only a few clients.

In addition, you must also supply the correct network protocol software to allow these machines to talk to one another. The different ServerBench platforms only work with specific network protocols; if you don't have them, you must buy them.

Here's what comes with the ServerBench benchmark

ServerBench comes in three parts. You get an executable for the server, one for the controller, and one for the clients. Each executable contains the ServerBench program for that machine (server, controller, or client) and any files the program needs, such as the seed files used to create the test data files.

Here's what doesn't come with ServerBench

You must provide the correct operating systems for your server, controller, and clients as well as the correct network protocol for these systems. In addition, you must also supply Microsoft Excel 5.0 or later, which ServerBench uses when it displays its results. We do not provide these things.

ServerBench also has other hardware and software requirements. The on-line ServerBench manual for your server platform contains a complete list of what ServerBench needs.

The results ServerBench produces

ServerBench produces an overall score for your server. You'll find the overall score in Table 1 of the ServerBench results spreadsheet. (Because we keep improving ServerBench, you can only compare ServerBench 3.0 scores with other ServerBench 3.0 scores; you cannot compare these scores with scores from earlier versions of ServerBench.)

ServerBench reports its results as TPS, or transactions per second. ServerBench's test suites consist of transactions. These transactions are the basic unit of work the clients uses to stress the server. Each transaction consists of the work requests a client sends to the server and the response the client gets back. Each client measures how long each transaction takes and how many transactions take place. The client then calculates the TPS. ServerBench uses these individual TPS scores to produce the overall server TPS score and individual scores for the clients.

You must have Excel 5.0 or higher if you want to view your results. ServerBench uses special Excel macros to format its result log files into a spreadsheet that you can read.

The secret to the ServerBench test suites

ServerBench uses stress tests to create its test suites. Stress tests let you place large amounts of work on the server. As a result, these tests exercise the server in less time and using fewer clients than it would take for you to monitor your server's day-to-day performance. This is why, even though ServerBench supports up to 1,000 clients, you can get an accurate measure of your server's performance with as few as three or four clients. Each client is doing the work of more than one user.

The ServerBench individual tests

ServerBench contains numerous individual tests that you use when you create the transactions ServerBench sends to the server. (A client never sends a test to the server; it always sends a transaction.) The individual tests include disk tests, a processor test, and network tests. ServerBench packages the transactions in a test mix and places the test mix in a test suite. A test suite can contain more than one mix. By running a test suite, you can combine the different subsystem tests together to create a set of transactions that use your server the same way your users do. The Ziff-Davis publications use the standard test suites located in the controller's SUITES directory when they prepare ServerBench articles.

Repeatability — isn't it great

We designed ServerBench so that you can get accurate and repeatable results. This means that test results from multiple runs of the same test suite with the same parameters are consistent with each other.

Don't forget to license your copy of ServerBench

Even though we give ServerBench away, it is still licensed software. You must license and register your copy of ServerBench before you can use it. The License Agreement appears on your screen the first time you run the benchmark and at the beginning of this manual.

If you want to run ServerBench, here're two tips

You'll get your best results when you run ServerBench on a test network under controlled conditions. If you run ServerBench on your production network, it will slow down your server and annoy users.

You should also make sure you don't have any other non-critical applications running. Running ServerBench while other applications are also making requests of

the server or the clients will distort your results and cause problems for any users who are using those applications.

End of "About ServerBench"

What's New in ServerBench 3.0

We work to make each version of ServerBench better than the last one. Here are some of the changes you'll see in ServerBench 3.0 that give it a leg up on the earlier versions. (The *ServerBench® Quick Start Handbook* contains an expanded list of these new features.)

More server platforms

ServerBench 3.0 has added three new Windows NT™ Server 3.51 RISC platforms and an OS/2 Warp Server platform. The new RISC ports use Digital™ Alpha™, MIPS®, and PowerPC™ microprocessors. In addition, ServerBench continues to run on Windows NT servers using x86-compatible microprocessors.

NOTE: ServerBench 3.0 does not support NetWare 3.1x or SCO Unix 4.0.

For the OS/2 Warp Server platform, ServerBench has switched network protocols. This port uses TCP/IP as the network protocol. ServerBench 2.0 on OS/2 required you to have NetBIOS on the server, controller, and clients.



Tip:

For a complete list of operating systems that ServerBench supports, see the main ServerBench **README.TXT** file. This file is located in the **\SB30** directory on the Ziff-Davis Server Benchmarks CD-ROM.

You can have Windows 95 clients/controller

With ServerBench 3.0, both the clients and controller will work with Windows 95 as well as Windows for Workgroups 3.11. When we added support for Windows 95 clients and controller, we kept the ServerBench client and controller programs as 16-bit Windows applications. Because we didn't upgrade them to 32-bit applications, they'll continue to work on PCs running Windows for Workgroups 3.11 and Windows 3.1.

Support for Winsock-compliant TCP/IP stacks

Most of the ServerBench platforms now accept any third-party TCP/IP stack that is Winsock 1.1 compliant as the network protocol for the controller and clients. For more information on the network protocol your server platform of ServerBench requires, see the installation manual for your ServerBench port.

Unattended mode adds error-handling features

ServerBench's new error handling features let you specify in advance what you want ServerBench to do if it encounters an error. These features work only when you run ServerBench in unattended mode (the default mode for running ServerBench). As a result of these new features, you can tell ServerBench to keep running test suites when an error occurs. Earlier versions of ServerBench stopped when an error occurred.

External Notification

ServerBench's external notification feature means that you can configure the controller to run a program, such as a pager program, if an error occurs. Basically, you enter a command line in the External Notification dialog box. Then, if an error occurs, the controller executes that command line. This command line can start any program, such as an executable, a **.bat** file, a **.pif** file, or an **.exe** file program. If you specify a pager program that dials a specified pager number and that program accepts alphanumeric pagers, you can actually send the error message to the pager.

You can replace a client that terminates

If you accidentally reboot a client during a "Connect Clients" stage and cause the ServerBench client program to terminate, you can continue without that client, replace that client with another client that has the same ID number, or reconnect that client. The controller will accurately reflect the number of connected clients when the test continues. In addition, ServerBench gives you opportunities to reconnect clients that terminate.

Enhanced results reporting features

With ServerBench 3.0 comes enhanced results reporting. Two of the biggest changes are that ServerBench now uses a **.TLG** extension to identify its results files and that ServerBench now places each set of results tables in an Excel workbook.

Here's a summary of some of the results enhancements in ServerBench 3.0:

- **You can display results for 60 or more clients at once.** ServerBench 2.0, because of limitations with the Excel macros, limited its results tables to a maximum of 30 clients at a time.
- **It's easier to name results files.** Once you've selected a test suite, ServerBench gives you the option of entering a name for that test suite's results file as well as an identifying comment to go with the results file. This way you can easily create several sets of results from the same test suite without having to worry about overwriting the previous set of results. By default, ServerBench uses the test suite name as the base name of the results file, gives it a **.TLG** extension, and places the file (and the other log files it uses) in the controller's **RESULTS** subdirectory. When you enter the name of the results file, you can also enter a path name to a different directory. If you enter the same results path name that an existing set of results has, ServerBench warns you and asks whether you want to enter a new name or overwrite the existing results.
- **You only need one session of Excel running.** Previously, each time you selected a set of results to view, ServerBench would launch a new session of Excel, even if you current had a session of Excel running.
- **You can create a TPS graph and a variance graph.** When you're setting up your results, you can easily install ServerBench's new Add-In module for Excel, **SVRBENCH.XLA**. This module lets you create a TPS summary graph and a variance summary graph.

The **SVRBENCH.XLA** module also enables you to easily print the graphs or tables from within the ServerBench workbooks without printing other forms of data.

- **It's easier to edit the database snapshot files.** The database snapshot files contain information about how your server and your clients are set up. ServerBench uses these files when it creates the server and client disclosure tables. Now you can directly edit a database snapshot file and change its contents. You'll find this option in the View Results dialog box.
- **We've improved the results tables.** We've streamlined spreadsheet layouts through Excel Workbooks to make ServerBench's results tables easy to understand and quick to manipulate. In addition, we've reorganized the information to make it easier for you to understand what the results mean and how ServerBench calculated them.

The standard test suites are now even more “stressful”

We've enhanced ServerBench's standard test suites to make them stress the server more.

- **System test suite.** As a result of our research, we've tuned the system test suite to increase the amount of time the network operating system spends in kernel mode. The system test still includes processor, disk, and network tests

structured into nine distinct transactions. This change does not affect the amount of time it takes to run ServerBench's system test suites with 60 clients; they still take about four to four and a half hours.

- **Processor test suite.** We've improved the Processor Test so that it has more control over the CPU load it places on the server. When you create or modify a Processor test in the Mix Definition window, you enter a value in the range of 1 to 2800 iterations for the Total Size parameter. (The value of the Total Size parameter for all the other tests is 1 to 16777215 bytes.)
- **Disk test suite.** We've changed the Disk test suite to modify the sizes of the data files so that none of these files will dominate the buffer cache during the execution of a mix. In addition, the server now aligns its read and write operations on boundaries that are same size as the Request Size parameter. For example, if the Request Size parameter is 2048, then the server will align the read and write operations on 2048-byte boundaries.

And, because you can display results tables containing 60 or more clients, we now supply test suites that go from 1 to 60 clients instead of the paired test suites we supplied with earlier versions of ServerBench. For example, instead of having **SYS_28.TST**, which used a maximum of 28 clients, and **SYS_60.TST** which started at 32 clients and ended with 60 clients, we now have **SYS_60.TST**, which starts at 1 client and ends with 60 clients. This means a single test suite does the same work of two test suites in previous versions of ServerBench. So the standard test suites that ship with ServerBench 3.0 are **SYS_60.TST**, **D_60.TST**, **N_60.TST**, and **P_60.TST**.

New features for creating mixes

To make it easier for you to create or edit a mix, we've added a number of improvements. One of the key improvements is that you can work with multiple mixes and test suites at one time. ServerBench lets you grab multiple test suites when you're selecting the test suites you want to use. Previously, you could only select one test suite at a time.

In addition, you can change the values in the Mix Definition window for one mix and have the changes apply to all the mixes in the test suite.

Another major change from ServerBench 2.0 is that you can insert mixes anywhere within a test suite. You can also reorder the mixes in a test suite.

The following is a summary of some of ServerBench 3.0's new features for working with mixes.

- **An improved Test Suite window.** This window lists the test suites you've selected, lets you add more suites or remove suites, and displays information about the mixes each suite contains.

- **The ability to insert mixes anywhere in a test suite.** You can create a mix and place it anywhere in the existing test suite (except as the first mix). The only caveat is that all the mixes must have a unique name.
- **The ability to reorder mixes within a test suite.** You can reorder the existing mixes in a test suite as long as all the mixes in the suite have a unique name.

When you're at the Mix Definition window, you'll notice that this window has a new look and several new features. One of the biggest changes to this window is that it is now a viewer that lets you move back and forth between all the mixes in that test suite. Use the Previous Mix and Next Mix buttons to scroll through the mixes in this test suite. Because of this feature, we've also modified the Cancel function. If you've edited a mix, then scrolled to another mix and edited it and choose Cancel, ServerBench will display a dialog box that will ask whether you want to cancel the changes to the mix just edited or cancel all the changes to this test suite.

Some of the new features you'll see at the Mix Definition window include:

- **Increased precision for the Disk test parameters.** You can enter values for the Disk test file initial size and Disk test file I/O Range parameters that go to three decimal points – 0.001 MB up to 1024.000 MB.
- **Changes to the Total Size parameter.** The value the Total Size parameter changes based on whether you're setting up a Processor test or one of the other tests. For the Processor test, the value of the Total Size parameter is in iterations (1 to 2800); for all the other tests this value is in bytes (1 to 16777215).
- **Entering client path names.** You can set up sequentially numbered path names for clients simply by entering a path name that ends in an asterisk in the Data File pathnames box. For example, if you've specified 60 clients and you enter the path name **f:\data***, the path names for those 60 clients become **f:\data1**, **f:\data2**, and so on up to **f:\data60**.
- **Inserting and deleting mixes.** Just as you can insert a mix into a specific place in the test suite from the Test Suite window, you can also insert a new mix into a specific place in the test suite from the Mix Definition window. When you have a mix displayed in the Mix Definition window, choose Insert new mix option in the drop-down Mix menu. ServerBench will let you create a mix that will follow the mix you were just working on. In addition, you can delete the mix you're currently editing. Choose the Delete current mix option from the Mix menu.
- **Copying fields to all the mixes in the test suite.** You can copy the fields in the mix you are working on to all other mixes in the test suite.

Other new ServerBench 3.0 features

This is a quick list of some ServerBench 3.0's other enhancements.

- **Controller window icon shows test run status.** ServerBench displays the status of the current test run even when you minimize the controller window. To disable this feature, choose the System menu (the icon in the upper left corner of every Windows application).
- **The controller sounds different tones when it displays certain dialog boxes.** The controller uses audible indicators when it displays various questions or special dialogs. This way you'll know without having to constantly watch the controller when certain events occur. For example, the controller sounds a tone when ServerBench finishes executing the selected test suites.
- **ServerBench displays client's real-time status.** If you click on a client square, the data in the Client Information pop-up box reflects the client's current status. The Status field continuously updates itself. For example, if that client is changing from the Initializing to Running test stage, you'll see the Status field change.
- **You can choose a smaller size for the client grid.** You can specify a client grid size of 77 squares. This grid size is more appropriate when you're running the standard test suites, which use a maximum of 60 clients.
- **Warning box appears if you choose Quit while the controller's in performing a crucial task.** If you try to quit the controller when it's processing critical data, ServerBench displays a warning message. For example, if you've told ServerBench to skip a mix, the controller will wait for the server and clients to clean up. If you press Quit now, the controller won't be able to tell the clients and server to abort. So ServerBench displays a pop-up warning box asking if you really want to quit the controller now.
- **New Test Suite History window lets you track test runs.** You can use this window to review which test suites ServerBench has run and is currently running, the name of the results file for each test suite, the path names for the results files, any special comments you entered, and the status of the test suite, such as whether ServerBench skipped any mixes in a particular suite or if any errors occurred. If you request the status of a running test suite, ServerBench displays the status in real time; i.e., the current status of that test suite at that moment.
- **ServerBench provides a Sticky Suites feature and a Sticky Results feature.** These two features mean that ServerBench 3.0 takes you to the directory where you last selected test suites to run or results to view. If you keep all

your test suites in one directory and all your results files in another, you'll won't have to spend time searching to get to these files.

- **You can grab multiple files.** Wherever possible, ServerBench lets you select multiple files for an operation. In the past, you could only select one file at a time.

End of "What's New in ServerBench 3.0"

Part 1

Getting to Know ServerBench

The goal of this part of the manual is to introduce ServerBench to you. It provides you with the basics about ServerBench. It also tells you something about how ServerBench works and the concepts that we used in developing the benchmark. In addition, this section contains a comparison between ServerBench and NetBench®, Ziff-Davis' file server benchmark.

If you have any questions about the terms we use in these chapters, check the glossary.

Chapter 1

Introducing ServerBench

This chapter gives you an overview of ServerBench. After reading this chapter you should have a general understanding of:

- What ServerBench is.
- How it works.
- Why you would want to use it.



ServerBench

ServerBench defined:

ServerBench is Ziff-Davis' industry-standard benchmark program for measuring the performance of application servers in a client/server environment. Client PCs running either Windows 95 or Windows for Workgroups 3.11 stress the server under test with transaction requests for different types of work that produce different types of load on the server. Each client records how many transactions it completes and how long it takes to complete them. ServerBench uses the client information to produce an overall TPS (transaction per second) score for your server and individual TPS scores for the clients.

ServerBench in brief

ServerBench is a Ziff-Davis benchmark program that lets you measure the performance of an applications server in a client/server environment. When we talk about an applications server, we're referring to a setup where most of your data and your applications execute on the server. The client PCs send requests for work to

the applications on the server. Once the server finishes the work, the application sends a response to the client. The client then performs some work related to the response. An example of an applications server would be a database server.

We created ServerBench so that you can use it to test different servers as they provide a variety of services to clients running Windows 95 or Windows for Workgroups. You can also use ServerBench to test one server and see what effect varying a piece of hardware or software affects your server's performance. ServerBench produces an overall score for your server. It also produces scores for your clients.

ServerBench terminology:

Server:

The applications server running the ServerBench server program. This machine receives transaction requests from each client for service, acts on them, and sends a response to the client. ServerBench runs on several different server operating systems. For a complete list of which server platforms ServerBench runs on, see the **README.TXT** file in the **SB30** directory of the Ziff-Davis Server Benchmarks CD-ROM.

Controller:

A PC running either Windows 95 or Windows for Workgroups and the ServerBench controller program. You execute the test suites, monitor the test run, view your results, and create test suites from the controller. The controller does not contribute to the results.

Clients:

PCs running either Windows 95 or Windows for Workgroups PC and the ServerBench client program. The clients send transaction requests to the server.

The ServerBench basics

ServerBench measures the performance of applications servers by having the Windows-based client PCs, send a variety of requests for work to the server. You start the tests from a PC called the controller, which is also Windows-based.

The requests the clients send are called transactions. Each client records the amount of time it has to wait to receive a response from the server for each transaction and how many transactions the server completes during the entire test mix. ServerBench takes all this data the clients collect produces an overall score for

your server. ServerBench reports these results in terms of TPS (transactions per second).

The ServerBench tests target your server's processor, disk, and network subsystems.

ServerBench works by running tests that produce different types of load on the server. Some of the tests focus on one subsystem while others exercise another subsystem. These tests perform operations similar to the operations your server normally performs. By combining the tests and adjusting the parameters for the tests, you can get an accurate and repeatable measure of how your server performs under normal operating conditions as well as a projection of how your server would work under a heavier load.

You can also limit your tests to specific subsystems. For example, you can run only the disk tests if you want to see how well that subsystem performs on your server.



Tip:

Because ServerBench performs its tests in a client/server environment, its results reflect the time it takes the client's request to reach the server, the time the server spends doing the work required by the request, and the time it takes the server to reply to the client once it has processed the client's request. This means that, even though you execute a test suite that contains tests for only one server subsystem, other subsystems can affect the test. The most obvious subsystem that affects all the test suites is the network subsystem since it serves as the communication mechanism between the server and the client for all the tests.

ServerBench can theoretically handle up to 1,000 clients. Your operating system network hardware and software, or test bed may restrict you to a smaller number of clients. Don't worry, though. Because ServerBench's standard test suites use stress tests, you don't need a lot of clients to get a representative measure of your server's performance. In fact, the standard test suites use only 60 clients. The reason you don't need a lot of clients is because each client in the test is pelting the server with requests as quickly as the server can handle them. As a result, a single client causes as much server work as many real users. Thus, ServerBench is pushing servers to their breaking points with a small number of clients.

ServerBench is a “synthetic” benchmark: It contains specially written tests that allow it to model the kinds of disk, network, and processor interaction clients have with an applications server. ServerBench uses its proprietary programs to provide this measure of applications servers; it does not use any commercial applications.

Using test suites to measure performance

When you run the ServerBench tests, you're actually executing a test suite. Each test suite consists of other layered components. Here's how it works:

- Each test suite you run contains one or more test mixes.
- Each test mix contains one or more transactions.
- Each transaction contains one or more tests bundled together.

So when you tell ServerBench to run a test suite, ServerBench checks the test suite to see which mixes to run. Each mix contains the test specifics the clients need, such as how many times a client should ask ServerBench to execute a transaction and what the parameters for the transaction are. To make the test even more representative of real-world server/client interaction, each client reorders the transactions before executing them. (This way the server is getting a variety of requests from clients.) The clients then start sending the transactions to the server.

By creating multiple mixes that use different transactions, you can create a test suite that exercises your server the way your clients normally exercise it but in a shorter period of time than monitoring actual use would take.



Tip:

ServerBench comes with several standard test suites you can run to get a representative score for how well your server performs overall, as well as how well the server's processor/memory, disk, and network subsystems perform. You'll find these standard test suites in the ServerBench subdirectory **SUITES** on the controller.

ServerBench provides the following individual tests:

- One processor test. This test measures the server's ability to do compute-intensive work without I/O.
- Five disk tests. You can use these tests to evaluate how well your server's disk subsystem works. The disk tests are sequential read and write, random read and write, and the append test.
- Two network tests. You can tell how well your server's network subsystem handles client/server transactions by running these tests. The network tests are client to server and server to client.

When you create your own test suites, you can vary the parameters for the different individual tests, the transactions containing the tests, and mixes containing the

transactions. This provides a lot of flexibility in designing the workload you use to test the server.

For more information about these tests, see Chapter 11 “The Details on ServerBench’s Tests.”

The ServerBench test environment

This means that to execute ServerBench you must have a networked system up and running; in other words, a network applications server connected to at least two PCs. One PC will be your controller and the other PC (or PCs) will be your client (or clients). You also need the hardware and software necessary to allow these machines to talk to one another.

Basically, the ServerBench test environment consists of the:

- **Network applications server under test.** This is the server you are testing. The server executes the transactions issued by the clients.

For each supported server platform, ServerBench uses its own specially written software (not commercial server applications) to exercise that server's three main subsystems (processor, disk, and network) in the same way that a server application does. All the server operating system hardware and software as well as the network hardware and software affect the ServerBench scores. This includes everything from the disk controller to the number of processors to the network protocol.

- **Controller.** The controller is a PC running Windows 95 or Windows for Workgroups 3.11. As the name implies, you use the controller to control ServerBench’s test runs. You start test suites, monitor their progress, and view the results from the controller. You can also use the controller to create test suites.
- **Clients.** These are the PCs running Windows 95 or Windows for Workgroups 3.11. The clients are the ones that actually request the tests ServerBench runs on the server. The clients ask for a mix of processor, disk, and network services. Each client keeps a tally of its completed transactions and how much time each transaction took starting with the moment the client made the request until the client got an answer from the server. You must have at least one client to run ServerBench. ServerBench will accept up to 1,000 clients; however, you can get a reasonable score for your server with as few as three or four clients, providing you set your test parameters to stress the server.

NOTE: Your clients can be on multiple network segments, where each segment is connected to a different network adapter on the server.

ServerBench relies on the networking software on the server to handle communication with the various clients.

- **Network Protocol Software.** The network protocol software is the piece of the ServerBench test environment that connects the different machines used in the benchmark tests. ServerBench does not provide network protocol software for the controller or clients; you must supply that. Each ServerBench platform supports a single network protocol; ServerBench doesn't work with each protocol a server platform supports. (See the on-line installation manuals for details on the network protocol software each port requires.)

To set up ServerBench, you must install programs on the server, the controller, and each of the clients. Thus, for each machine in the test (server, controller, and all the clients) you'll need a ServerBench directory with the correct program in that directory. (The controller **SETUP.EXE** and client **SETUP.EXE** both create these directories for you. The server installation steps vary depending on your server platform. For information on installing ServerBench, see the ServerBench on-line installation manual for your platform.)



Tip:

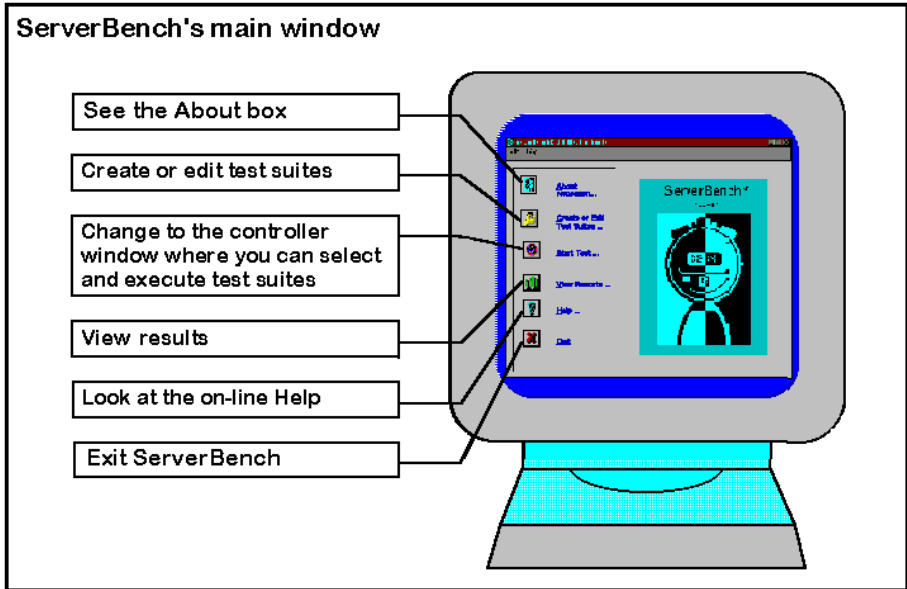
Always set up a test network for running ServerBench. Otherwise the network traffic ServerBench generates will slow down your network and annoy your regular users while the traffic your users generate will lower your ServerBench results.

A quick look at how ServerBench runs

To execute ServerBench's test suites, you must have the controller program running on the controller, the server program running on the server, and the client program running on each of the clients in the test. Once ServerBench is running on the test machines, you will do most of your work at the controller; you won't need to enter any additional commands at the server or the clients.

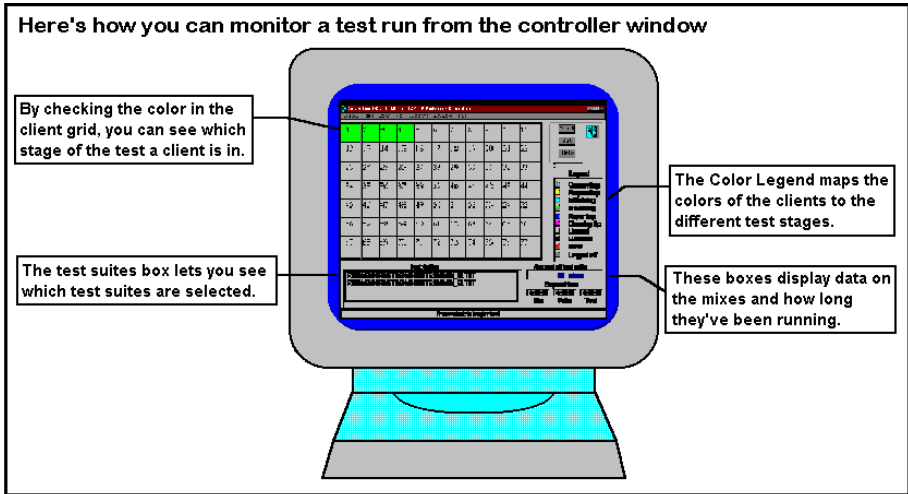
From the controller you can start test suites, create new test suites, or view your test results.

Figure 1-1: The tasks you can perform at ServerBench's main window



When you're at ServerBench's main window on the controller, you can change to the controller window by choosing Start Test. You execute ServerBench's test suites from the controller window. You can also monitor the clients from this window. As the tests run, the controller window changes to let you know how the tests are progressing. You can see which clients are running the test and which (if any) are excluded from the test. You can also see which stage the clients are at in the test (their squares in the client grid change colors), which test mix is running, how long it has been running, and how many other test mixes are left to run.

Figure 1-2: Monitoring a test run from the controller window



After you press Start from the controller window, you'll need to go to the server and enter the ServerBench command line. Once the ServerBench server program is running, you'll need to go to each client and choose the client icon to start the ServerBench client program. When ServerBench is running on the server and clients, you won't have to enter any additional commands on those machines.

Additional information about the clients

The ServerBench client program displays a large block letter on each client that lets you tell at a glance which stage of the test process the client is in. If the client is one of those chosen to run the test, then the client window also displays information on how much of the test has executed.



ServerBench automatically saves the results from each test suite it executes in a set of log files. You supply the name for these files when you choose the test suite.

ServerBench uses these files to generate the results it displays when you choose the View Reports button.

For more information, see Chapter 12 “Let's Take a Look at Those Results.”

The results ServerBench produces

ServerBench calculates its results as the number of completed transactions per second or TPS. ServerBench uses special Excel macros to display these results in spreadsheet format. The results ServerBench produces let you measure the overall performance of a server and the type of service the clients are receiving. The ServerBench results:

- Let you analyze and predict the performance of different applications servers that are tested under the same conditions and using the same testbed.
- Give you the means to see how changing a variable in a test or on your server affect your server's performance.

Why you should run ServerBench

ServerBench gives you a way to compare different applications servers to each other. ServerBench is also a good tool to use if you want to see how changing parameters and software on your server affects its performance.



Tip:

When you're checking the affect that changing a parameter has on a server, make sure you change *only one* parameter at a time.

If you're planning to compare different servers, you need to:

1. Have the servers set up the same way.
2. Use the same testbed.
3. Run identical test suites on each server.

If you don't follow those steps, you won't get a valid comparison. For example, you won't get much information about the true relative speeds of two servers if you run ServerBench on one server with 64 MB of memory and on the other server with 128 MB of memory.

However, you may want to test the effect that changing the amount of memory has on performance by running ServerBench on a single server with everything else the same except the amount of memory. Or you might want to see how well your server performs with one processor and then two processors.

Forget those results from earlier versions

Because we've substantially upgraded ServerBench 3.0, you can't meaningfully compare its results with results you got while running an earlier version. The key change we've made is to beef up our test suites. The load they place on the server more closely reflects the loads of real server applications.

The bottom line? Only compare ServerBench 3.0 results with other 3.0 results.

Using ServerBench with NetBench®

The Ziff-Davis family of benchmarks includes two industry-standard server benchmarks: ServerBench and NetBench. ServerBench focuses on application servers while NetBench works with file servers. Both ServerBench and NetBench give you different information about your server.

ServerBench gives you a gauge of the overall performance of an application server used in a client/server environment. The ServerBench applications reside on both the server and the clients. This means that in the ServerBench world, the client program acts like a "front-end" program; it takes a transaction from the mix and sends that request off to the server. The server program gets the transaction, tells the server what operations to perform, and then returns a response to the clients. This approach reflects how users working at PCs connected via a network to an applications server use that machine (in other words, the server executes the tests).

NetBench measures the file I/O performance of a file server. As a result, it only needs a client program, not a server program. The clients do the work; all they use the server for is a place to create files and move data back and forth. So for NetBench, the client program is essentially the "back-end program" (i.e., the tests are executed on the client). Because NetBench does not place any code on the server, it is independent of the network media type and the underlying network protocols.

NOTE: Keep in mind that, even though NetBench does not require a specific network protocol, the type of protocol you use can affect your test results. Some protocols are faster than others.

Which benchmark is the best one for you to run depends on what type of work you do on your server. If you are using your server as an applications server where you have a database located on your server and users enter data into the database via a database interface running on individual PC clients, ServerBench will give you more appropriate results. However, if users simply log onto the network, get data from the server, work with files on the PC clients, and then store them on the server again, NetBench is the benchmark you want to run.

End of chapter

Chapter 2

How ServerBench Works

This chapter gives you an overview of the inner workings of ServerBench. It provides a look at how the ServerBench programs talk to each other. It also gives you a simplified view of how ServerBench uses provider models for communicating with the server on the different operating systems.

To make this chapter easier to understand, we stick to general explanations of the programs that make up ServerBench.

The foundation for ServerBench

We modeled ServerBench after an applications server environment. By applications server environment, we mean an environment where both your data and your applications exist on the server. The client PC, while it does some work of its own, is mostly a front-end that provides an access point to the application on the server. For example, suppose you have a large database system hosted on the server, which accepts requests from clients connected via a LAN. This is the type of environment that the ServerBench system test suites model.

The subsystems ServerBench stresses

The server subsystems include both hardware and software. This is why the speed of your hard disk alone doesn't tell you how fast your disk subsystem is. When we say "disk subsystem," we're talking about the hard disk (or disks) your server has, the disk controller (or disk controllers) it uses, any disk caching it uses, and software such as disk drivers and the operating system's file system.

By processor subsystem we mean the main processor chip (or chips, if you have a multiprocessor system), your server's main memory, the internal and external cache (if any), cache controller, and memory bus. Scheduling is also an important part of the processor system.

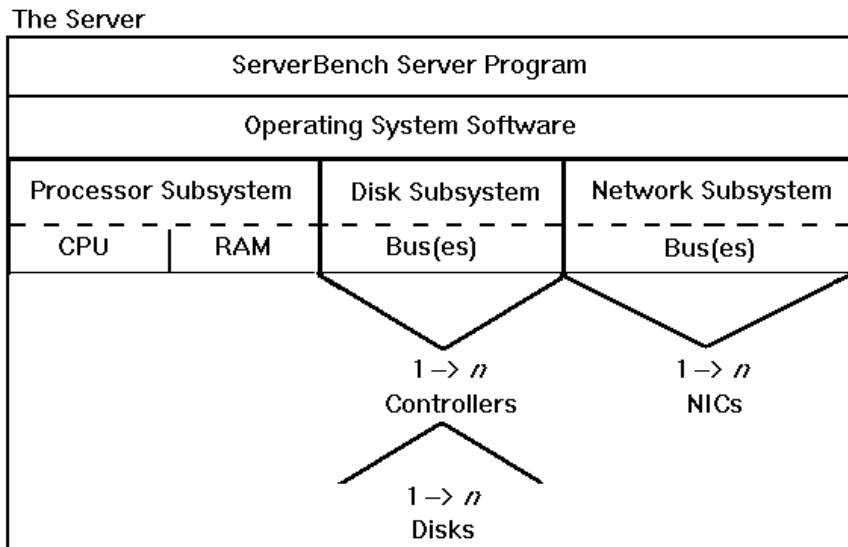
We regard your network subsystem as including the network cabling (for example, Ethernet or token ring), the Network Interface Cards (NIC) and their drivers, and the network protocols (for example, TCP/IP).

Above these subsystems, you have the software for the network operating system. Your ServerBench server program sits on top of the operating system.

Here's where ServerBench fits in

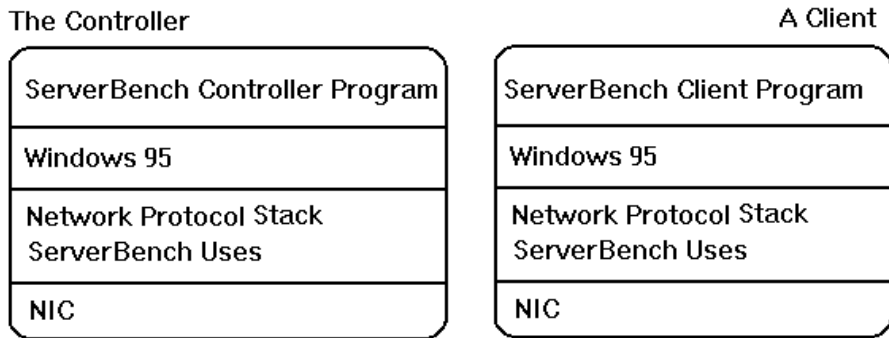
Based on the previous description of subsystems, your server looks something like the diagram in Figure 2-1. It's a machine that has your ServerBench program on top of your network operating system software on top of your three server subsystems. (Keep in mind that there are several different ways to look at a server and Figure 2-1 presents just one of those ways.)

Figure 2-1: One way to look at your server



Now we take a look at the PCs that serve as your controller and clients. The only difference in these PCs is that one is running the controller program while the other is running the client program. These PCs have the ServerBench program sitting on top of a Windows-based operating system (either Windows 95 or Windows for Workgroups 3.11). Because the ServerBench program for both the controller and the clients is communicating with the server, the next layer in these PCs is the network protocol stack ServerBench uses on that server port and the network interface card (NIC). Figure 2-2 illustrates one way to look at these PCs.

Figure 2-2: A look at your controller and client



The ServerBench programs and how they work

ServerBench consists of three proprietary programs — one for the server, one for the controller, and one for the clients. Each program performs a set of tasks that is both crucial to running ServerBench and tailored to the system that program is running on.

The server program contains the code it needs to

- Communicate with the controller and clients.
- Decode the ServerBench requests the clients send.
- Determine which tests to perform.
- Execute the requested tests.
- Encode the test results and send them back to the client.
- Combine the client log files and send them to the controller.

The controller program has to:

- Run the controller interface.
- Communicate with the server (the controller only communicates with the clients via the server; it never communicates directly with the clients).
- Communicate the details of the test suites to the server, which then communicates this information to the clients.
- Tell the server when to tell the clients to start the tests.
- Receive state information from the server and update the controller window based on the progress of the tests and the changes in the clients' status.
- Work with Excel to display the results.

The client program has to:

- Run the client interface.
- Communicate with the server (the clients only communicate with the controller via the server; they never communicate directly with the controller).
- Get the description of the next mix to run that the controller has passed to the server.
- Randomize the order of the transactions in the mix.
- Encode the transaction requests and send them to the server.
- Time how long it takes to get a response from the server once the client issues a transaction request.
- Validate the transaction.
- Create log files containing the results of the mix.
- Send the log files to the server.

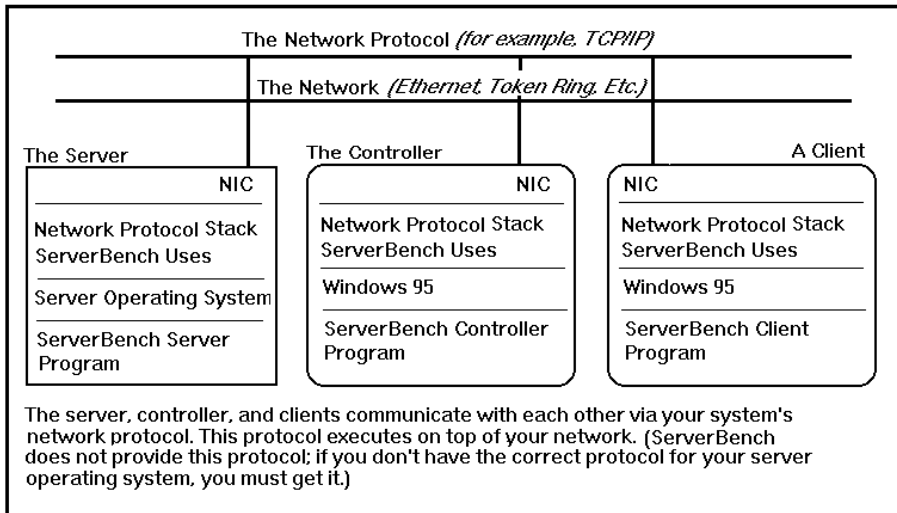
Because we designed the ServerBench programs specifically for the different server platforms, ServerBench only works with certain network protocols (for example, Winsock 1.1 compliant TCP/IP stacks).

NOTE: ServerBench does not supply any network protocols. If you want to run ServerBench, but you do not have the correct network protocol installed on your system, you must get a copy of it.

When you install ServerBench, you place the server program on the server, the controller program on the controller, and the client program on each client you plan to run in the test. Just like most other applications, these programs reside on top of the operating system for each machine. The server program is on top of your server operating system. The controller program executes on top of Windows 95 or Windows for Workgroups 3.11. The client program also runs on top of Windows 95 or Windows for Workgroups.

These three ServerBench programs communicate using the network protocol that ServerBench requires for your server operating system.

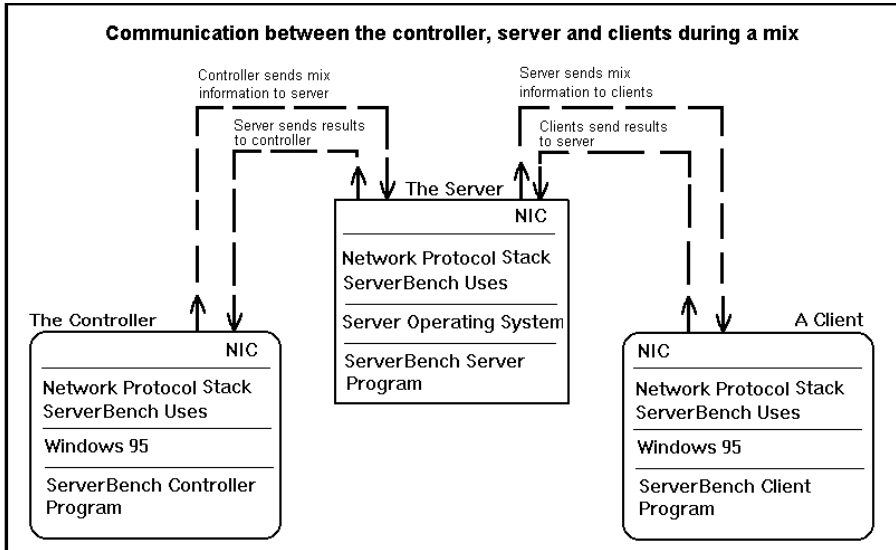
Figure 2-3: The server, controller, and one client



When you run ServerBench, the controller directs the tests (it does not run any tests itself and ServerBench does not count it in the test results).

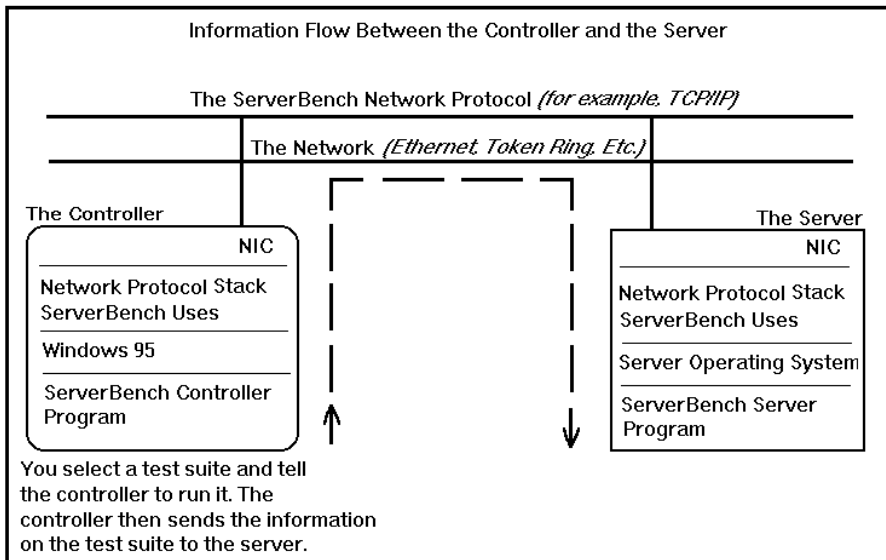
When you run ServerBench, the server acts as a communications interface between the controller and clients. The controller and clients never communicate directly. So when you run a test suite, all the communication flows through the server, as shown in Figure 2-4.

Figure 2-4: The flow of information during a ServerBench session



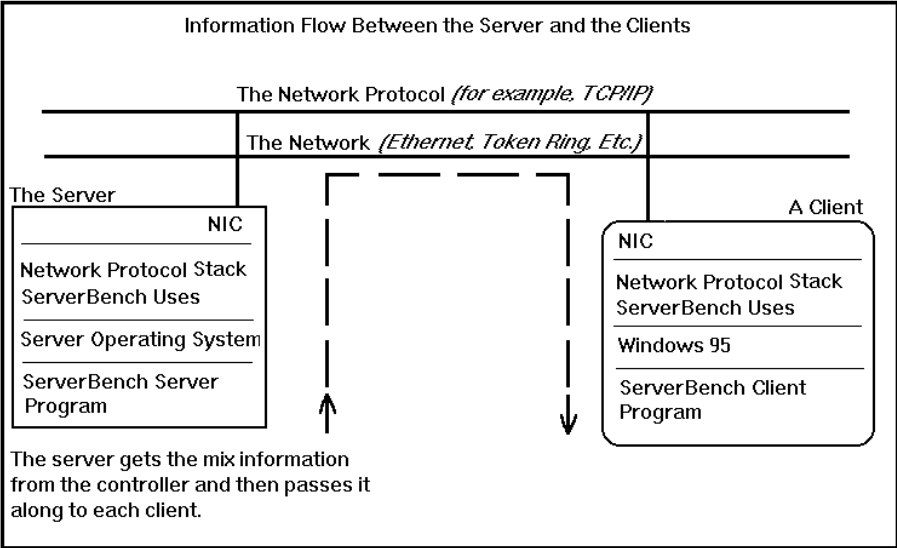
Here's a overview of how the controller, server, and clients interact during a test run. At the controller, you select the test suites you want to execute from the controller. When you tell the controller to start the test run, the controller communicates this information to the server.

Figure 2-5: The controller communicates with the server



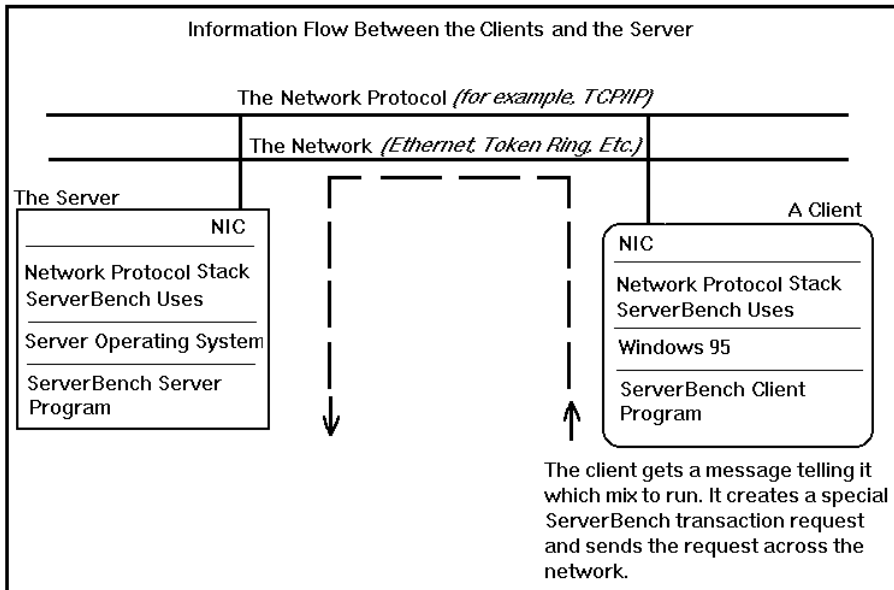
The server then sends the mix to each of the clients as they request the mix. When all the clients have received the mix, the server begins the mix initialization stage. It creates the necessary resources for each client, such as a data file for the disk test. The server also communicates with the controller to update the clients' status in the controller's client grid. When the server has completed the initialization for all the clients, it tells the clients to start executing the mix.

Figure 2-6: The server sends mix information to the client



When a client starts the mix, it encodes a transaction into a ServerBench request and sends the request across the network to the server.

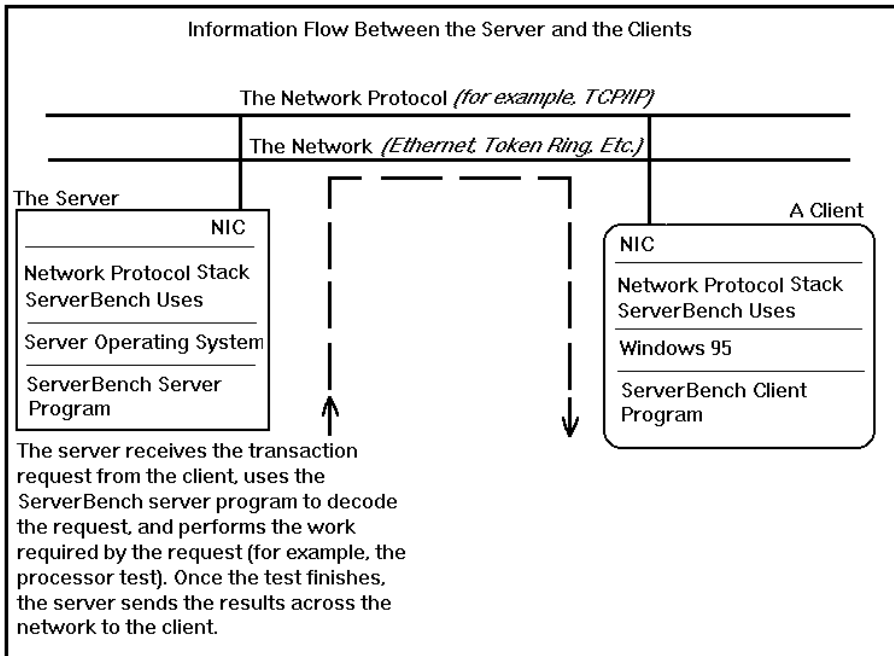
Figure 2-7: The client sends transaction requests to the server



The ServerBench server program decodes the request and performs the transactions specified by the request. (For example, if the client is requesting the Processor test, then the server begins executing the Processor test.)

Once the server performs the transactions, it encodes its response and sends the response to the client. For most of the tests, this reply is simply an acknowledgment that the test is complete. For the network tests, the server actually sends data back to the client.

Figure 2-8: The server sends a reply to the client



When the client receives the response from the server, it does the following:

- Adds the amount of time that elapsed from the moment it sent the request to the server until it received a reply back from the server to the accumulated total for that transaction.
- Increments the count representing the number of times the server successfully executed that transaction.

When the mix ends, all the clients send their results and error logs to the server. The server combines the logs and transfers them to the controller. Later, when you choose View Results, ServerBench combines the client logs and calculates the overall ServerBench score for the server as well as individual scores for each client.

The basic provider model

The term provider model refers to the platform-specific implementation of the ServerBench application's architecture. While this layer of implementation is transparent to you as you run ServerBench, it's important to know that ServerBench changes its model to support the different server platforms.

NOTE: Because ServerBench works with models that use processes as well as models that use threads, we use the generic term *provider model* in this manual to describe the ServerBench architecture for a specific platform.

Each server operating system implements a different provider model. When ServerBench runs, it uses the model that works best for your server platform. We work with the different server platform vendors to find the models that are appropriate for ServerBench on their server operating systems. We then port ServerBench to the server operating systems using those recommendations.

Provider models all contain the following key pieces:

- A master provider. This provider acts as a kind of overseer (in a very broad sense) of the ServerBench transaction process. The master provider sets up the initial communication between clients and service providers. The master provider does not take part in the tests.
- Multiple service providers. These providers actually handle the client/server communication. You can have one service provider per client or one service provider for multiple clients.

There are a lot of variations that can occur with provider models. As we mentioned, the server operating system may use either threads or processes. The most important difference in provider models, though, is the procedure the server uses to assign providers to clients. Some operating systems use a detailed scheduling procedure to allow one provider to handle multiple clients. Others use a one-to-one ratio of providers to clients. The goal of all the models is to balance the overhead of managing the providers against the clients' need for resources. You don't want your system to start thrashing.

From our standpoint, using the correct provider model is important because it can affect:

- The speed at which the server responds to the client requests.
- How long the request spends waiting in queue.

These two factors affect how even and efficient the service is on your server. So when we port ServerBench to an operating system, we try to use the most appropriate provider model for your server platform.

For information on how ServerBench implements this model for your server operating system, see the on-line installation manual.

The next section gives you an example of a very basic provider model.

Sample provider model

This example shows a provider model that uses a 1-to-1 provider/client ratio. This means the master provider associates each client with a single service provider. That service provider handles the client's requests to the server and returns the server's response to the client. This model uses the following format:

1. Once you've started the controller, you start the master provider on the server by starting the ServerBench server program. The master provider listens for connections from clients.
2. You start ServerBench on a client. That client connects with the master provider so that it can communicate with the server.
3. The master provider creates a service provider for that client. The client is now connected to the service provider, not the master provider. The client will use its service provider for all its communication with the server.
4. Steps 2 and 3 repeat until all the clients are running the ServerBench client program and each one has its own service provider.
5. The client makes a request of the service provider.
6. The client's service provider executes the request on the server and then returns the reply to the client. As long as the client is connected to the server, the same service provider handles its requests to the server.

A typical test scenario

The following list steps you through what happens when you run a test suite.

1. Execute ServerBench on the controller by choosing the ServerBench icon from the Ziff-Davis Benchmarks program group. The main ServerBench window appears.
2. Choose the Start Test button. This takes you to the controller window.
3. At the controller window, choose the Start button. The controller will display a pop-up box prompting you to start the server program.
4. Go to the server and enter the ServerBench command line. This starts the master provider. The server program communicates with the controller and creates a "Connect Clients" dialog box in the controller window. Now you can connect your clients.
5. Go to each client and start the ServerBench client program. The client program will connect to the server and the server will relay that information to the controller. (The clients never talk directly to the controller.)

NOTE: When you connect a client, you'll notice that client's square in the client grid of the controller window changes to the connect color shown in the Legend on the controller window.

6. Now, return to the controller and select OK in the "Connect clients" pop-up box once you've started ServerBench on all the clients.
7. A dialog box appears asking "Do you want to select a test suite." Choose Yes.
8. Select the test suite (or suites) you want to run and choose OK.
9. The controller passes information on the first test mix in the test suite to the master provider.
10. The server sends information on the first test mix in the test suite to the clients. During this initialization phase, ServerBench creates any data files and allocates any buffers it will need during the test.
11. Each client then sends the transactions defined in the test mix to the server and stores its results in a client log. After all the clients complete the mix, each client sends its log to its server. The server then sends the logs to the controller.
12. ServerBench repeats steps 9 through 11 for all the mixes in the test suite.
13. ServerBench saves your server's results each time a mix completes.
14. When the test ends, you can view your results from the controller. To do this, switch from the controller window to the main ServerBench window.
15. Choose View Results from the main ServerBench window.

NOTE: ServerBench displays the results using Excel. Because ServerBench uses several special Excel macros to format the results, it can take up to a couple of minutes for the results to appear. How long it takes depends on factors such as the size of the mix, the number of clients, and so on.

The steps above describe what ServerBench does during a single test suite. You can run multiple test suites during a single ServerBench session. As each mix completes, ServerBench stores the results from that mix. This way you never risk losing results from a mix that has run if something happens later in your ServerBench session.

End of chapter

Chapter 3

The Concepts Behind ServerBench

This chapter presents a general explanation of some of the concepts that we considered in developing ServerBench. This chapter also describes some of the elements that help ServerBench to accurately measure your server's performance. These are concepts you may want to keep in mind as you use ServerBench.

The design goal

Our goal in designing ServerBench has always been to create a benchmark program you can use to realistically measure a server's performance in a client/server environment without having to spend 24 hours a day for a week monitoring the server.

Because we modeled ServerBench after an applications server, ServerBench's server and client programs simulate the interaction between PC users and a server. To make it easier for you to set up and monitor the tests without affecting their results, ServerBench also provides a controller program. The controller program, like the client program, runs on a Windows-based PC, but it doesn't execute any tests and it isn't included in the results.

These proprietary programs contain synthetic tests we created that mimic actual server/client activity. When running a ServerBench test suite, the clients pound the server with more requests than a server would normally get in a real-world server/client environment. As a result, ServerBench can produce a realistic score for your system in a shorter amount of time and with fewer clients than if you were actually monitoring your system.

The following sections explain some of the factors we considered as we developed ServerBench.

The advantage of having a synthetic benchmark

We created ServerBench as a synthetic benchmark to allow you to get a reasonable measure of your server in a short period of time. In general, you can run a synthetic benchmark in less time than it would take to run each application and still get an accurate reflection how well the machines under test perform. Or, in ServerBench's case, how well a server handles client requests.

A synthetic benchmark such as ServerBench succeeds because it runs specially written tests that model the way applications behave. So you get the benefit of having a test that performs application operations without having to take the time to run each application separately.

Another advantage of using a synthetic benchmark is that you can run tests that focus on specific operations, such as disk operations, without having to perform all the other operations an application performs. ServerBench provides tests you can use if you want to target the performance of a specific subsystem: processor, disk, or network.

Providing an easy-to-use benchmark for different servers

We've ported ServerBench to several different server platforms and we're continuing to port ServerBench to even more platforms.

The user interface and the steps to run test suites are the same regardless of which server you use. This makes it easier for you to test different machines. In addition, the requirements for the controller and the clients are essentially the same for each ServerBench platform. (As with the server, you'll need to make sure these PCs also have the correct network protocol for your server platform.)

ServerBench comes with a program for the server, the controller, and the clients. It also comes with suggestions for how you set up your server to run ServerBench on your server platform. You can install and start executing ServerBench fairly easily.

See the on-line ServerBench manual for your server platform for information on setting up ServerBench on your server.

Here's what client/server means

By now you know that ServerBench is a client/server benchmark. But do you know what that means?

When we talk about a client/server environment or testing an applications server, we're referring to a system that consists of PCs (the clients) connected to a server via a local area network (LAN). In this system, the server handles the back-end processing while the client handles the front-end processing. This means that most

of the actual work occurs on the server. That's where the user applications and the user databases reside. If a client needs to search for something in the database, the search takes place on the server, not on the client. The client generally is running a smaller program. So, while the client does work with the data and is running some programs of its own, its primary purpose is to provide the interface the user needs to communicate with the server and access the applications and database there.

ServerBench models this work environment. It provides a program that the client runs and a program that the server runs.

When you execute a ServerBench test suite, each client in a test sends a transaction request to the server. Each request asks for different types of work from the server. The server gets the request and begins doing the work in it. Then the server sends a response to the client. When the client gets the reply to one transaction, it sends the next transaction. The server handles transactions from multiple clients concurrently.

While the server is performing the work requested in the transaction, the client waits for the server's response. When the response arrives, the client calculates how long it waited after it issued the transaction to get response from the server (i.e., how long the transaction took) and it updates its log of transactions.

In a file server environment, the server setup tends to be almost the opposite of an applications server setup. The majority of the applications and databases reside on the individual clients. The client could work on the server; however, the server is primarily a repository for files. A user gets the files off the server, moves them to the client PC, and works on them there. As a result, the server's CPU power doesn't have too much of an impact on the performance users want. However, the network resources can make a big difference.

The limits of testing an applications server

Testing an applications server imposes certain constraints on how you can tailor the tests and evaluate your results. You need to remember that the key components of applications server consists of an operating system, a network system, and the hardware and software that makes everything work.

This definition of a server defines the level at which ServerBench operates. It measures the overall server and server subsystems. It does not measure individual components of the server, such as the speed of your hard disk; everything about the server affects your ServerBench results. This is different from the way other Ziff-Davis benchmarks measure PCs.

For example, if you run the disk test for Ziff-Davis' PC Bench™, you will learn how fast your disk is. The PC Bench disk test focuses solely on the disk and does

not worry about the other components that make up your disk subsystem. However, if you run the disk test for Ziff-Davis' WinBench®, you will learn how fast your disk is *as you use it in a Windows environment*. And, because almost everyone uses SmartDrive on Windows, your disk score will also indicate how well SmartDrive manages your disk. Thus, unlike the PC Bench test, the WinBench disk test is not just a strict measure of how fast a disk is but a measure of your disk, the Windows environment's effect on your disk, SmartDrive's effect, and so on. You've started adding layers to what affects your test results.

When you run the ServerBench disk tests, you add even more layers. Now you're measuring how fast a disk is in a server environment that involves networking, processor scheduling, and probably other features such as an intelligent disk controller, a RAID array, and so on. In other words, a server's disk subsystem involves a lot of hardware and software. You could strip all of this hardware and software away and get a burst rate for your disk, but you would not be getting a meaningful measure of your disk subsystem.

All of this means that there is a limit to how low-level ServerBench tests can be. Even when you run the disk test, you're getting more than a measure of your disk. You're getting a measure of your disk in an environment involving a server and where the server and clients interact across a network.

How the subsystems affect each other

The most obvious subsystem that affects other subsystems is the network. Each transaction a client sends the server must travel across the network. This travel time is part of the total time for that transaction.

If you have a saturated LAN, your clients won't be able to send transactions. As a result, the lack of effectiveness in your network will dominate all your results. So, even if the transactions only contain disk tests, your scores will reflect the speed of your network subsystem.

Knowing how the subsystems affect each other can help you spot bottlenecks in your server subsystems. For more information on how to spot a bottleneck, see the section "Checking for bottlenecks" in Chapter 13 "Figuring Out What Your Results Mean."

The benefits of using stress tests

One of the main reasons ServerBench can successfully measure your server's performance is that it uses stress tests.

Using a stress test is like going to a cardiologist to have your heart evaluated. To get the best measure of how well your heart is functioning, your cardiologist would simply follow you around for a week and monitor your heart. However, your cardiologist doesn't have the time or the resources to do that, so he has you do a treadmill test instead. This causes your heart rate to increase significantly and gives your doctor an indication of how well your heart is functioning.

The ServerBench stress tests use the same principle. Most people don't have enough time and enough clients to adequately test a server as it runs. Thus, the stress tests let you measure your server's performance in less time and using fewer resources.

The differences between a stress test and a typical server workload are:

- A stress test lets you impose an arbitrarily large amount of work on the server.
- With a stress test you can target a specific server subsystem of the server for testing.
- Clients within a stress test turn around requests much faster than they would in a real-world scenario. They don't pause the way actual users would.

One client does the work of several users

Because clients within a stress test pound the server with requests, each client can equal more than one actual user in terms of the work it does. The advantage to you is that you don't need as many resources to run ServerBench and measure your server as you would if you were actually monitoring your server. You can push your server to the breaking point with just a few clients.

The reason for this difference between clients and users is that, when you run ServerBench's standard test suites, each client immediately fires off a new request to the server as soon as it gets a reply from the server. An actual user would probably take the server's response, do something with the data, and then send another request. In essence, the server would normally get some breathing room between user requests.



Additional Information:

There's no magic formula to determine how many users one client equals. However, the smaller the value you enter for the Think time parameter, the more a single client will stress the server. Think time is the mix parameter that governs how long a client waits after getting a response from the server before it sends another request to the server. For more information about Think Time, see Chapter 10 "Creating Your Own Test Suites."

Different clients put different loads on the server

Unless each client in your testbed is identical in every way possible, you'll notice that different clients put different loads on the server. For example, if one client has a more powerful processor and/or a faster network adapter or driver, it will issue transaction requests faster than a client with less powerful components. The upshot of the first client's additional power is that it will place a greater load on the server and get a higher score for a mix.

Because of these differences in clients, you can't fairly compare the results of a test suite using one testbed with the results of a test suite using another testbed.

To see these differences in clients, look at your client disclosure information. ServerBench displays information about your client configurations when you view the test results. The client information is in Table 3.

Creating tests that model real users' work

The foundation to ServerBench's success as a benchmark is the way it tests the server. ServerBench lets you create tests that exercise the server the way users would. These tests don't simply perform one operation repeatedly and then perform the next operation repeatedly. That kind of monotonous repetition doesn't accurately measure a server's performance. In fact, it may give you an overly high score for your server. When a server performs the same test several times in a row, the overhead of executing the test drops off.

ServerBench avoids this problem by:

- Using transactions that request a variety of server work. For example, our standard transactions contain processor, disk, and network operations.
- Mixing up the transactions within a mix. Each client running in a mix takes the list of transactions and randomizes their order. This way the clients give

the server a true "mix" of requests, which means the server is handling all types of operations at any given moment. In addition, if all the clients requested the same tests at the same time, then one subsystem would be overloaded while another sat idle.

The process we use for creating tests is another way we model a real-world environment. Each test suite is a carefully combined layer of tests:

1. The lowest layer in a ServerBench test suite consists of the processor, disk, and network tests. Each of these tests by itself exercises one of the server's three main subsystems. Before creating the tests, we decided which subsystems we should measure and then designed the tests to measure them. We also made sure that you could vary tests as needed. So we include several test parameters, such as Request Size and Total Size, that affect how that test runs.
2. Next is the transaction layer. While it may look like we're just complicating the process of creating a test, we're actually improving ServerBench's ability to simulate an applications server environment. By bundling tests together we're again giving the server a variety of operations to perform. In a real-world database situation, most operations involve work from multiple subsystems. The server has to read some data, do some processing, and so on. By using transactions, the ServerBench test suites can more closely reflect the kinds of requests for work an applications server might normally get. In addition, by sending several tests across the network as a unit, we're cutting down on the effect that the network has on the test results.
3. Now we come to the mix layer. By creating a test mix that contains several transactions, we've added even more variety to the operations the server performs. Remember that each client will reorder the transactions in the mix. So the server will stay very busy handling one transaction from client 1 and a different transaction from client 2, and so on. In addition, by changing the mix parameters (such as Think time, Delay, number of clients, and so on), we can adjust how much stress each mix places on the server.
4. The final layer is the test suite. The test suite is the container for mixes. You use test suites to group mixes together. Doing this lets you see the effect the tests are having on the server. For example, if you want to plot a results curve, you can create a test suite containing several mixes where each mix increases the number of clients. By looking at your results from this test suite, you can see how adding clients affected your throughput.

Measuring the results using TPS

Once we figured out how to create meaningful tests, we had to decide how to measure their results. Because the basic means of communication between the client and the server is the transaction request, it seemed logical to make that the primary metric to measure.

During a test mix, each participating client keeps track of the totals for each type of transaction it requests and how long each transaction takes. Once the test completes, the client divides the total number of completed transactions by the time the transactions took to complete. The client then sends this information to the server, which passes the information on to the controller. The controller calculates an overall ServerBench score for the mix using the data from the client.

Here're the steps each client performs during a test mix:

1. The client starts a timer and sends a transaction request to the server.
2. The server replies to the request.
3. The client stops the timer.
4. The client validates the transaction.
5. If the transaction passes the validation, the client updates it transaction time log and its transaction count.

After going through the above steps, the client starts the same process all over again.

So, the time the client records for a completed transaction is the time the transaction spends:

- Traveling along the network both to the server and then back to the client.
- Waiting in a queue for a service, such as disk service.
- Receiving the service.

The role cache plays

The amount of server file cache you have has an effect on your test results. A large disk cache can reduce the number of disk accesses your server makes. Because your processor is faster than your disk, you will probably see higher scores in cases where the ServerBench test data file fits in cache.

NOTE: There're different levels of disk cache. When we talk about disk cache, we're usually talking about software cache that the operating system manipulates.

Here's how cache works. Cache is a storage area made up of random access memory (i.e., RAM), which is the main working memory for your server. It lets your server access the information it needs extremely quickly. Cache is not a storage area sitting out on a hard disk. Instead, cache acts as a buffer between the CPU and the disk drive.

The advantage to having a program or test data file fit in cache is that then your server can get all the information it needs without accessing the disk. This means your server does not have to slow down to match the speed of your disk. When your server writes the data to the disk, it can be only as fast as your disk is because it has to wait for the disk-writing operations to complete. Cache, on the other hand, accepts all the data that needs to be written to disk just as fast as the CPU can send it.

An example of how cache works occurs when you only have one client running disk tests. As the tests run, you may not see the disk light on your server flashing. This is because all the test data fits in cache.

Factoring in multitasking

Multitasking is at the crux of the ServerBench tests. In an multitasking environment, many tasks share processing time. The processor swaps the tasks on and off.

NOTE: Many of the machines ServerBench runs on use multiple processors. The following, however, is a single processor example.

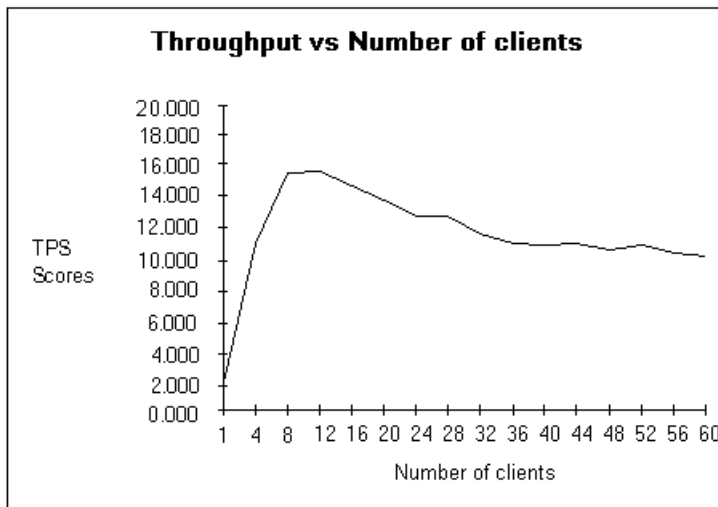
ServerBench uses multitasking this way. Each mix contains a fixed number of clients. The mix tells the clients which transactions to send to the server. The clients begin the test and start sending transaction requests to the server. The server, instead of telling the clients to wait while it handles one client's request, takes all the transactions from all the clients. Because a processor can only respond to one request at a time, the server lines up the other requests and puts them in a queue to wait for the processor time they need. If there are a lot clients sending requests, the queues can get pretty long. Meanwhile, as the clients wait for the server to reply to their tests, they're running a clock and calculating how much time the server is taking.

By knowing how many clients are competing for processing time and how much time it takes for them to get service, you can calculate the throughput for the tests and discover at which point the throughput drops off.

Figuring out the knees in the results curves

One of the key results that ServerBench returns is a curve that plots a server's total TPS throughput against the number of active clients. The point at which the number of clients is increasing but the throughput starts to level off is called the knee of the curve.

Figure 3-1: A results curve containing a single knee



As you increase the number of clients sending requests to a server, you expect the total throughput on the server to go up. This is because at low client counts, the server is not being utilized to its full potential, just as a bank with eight tellers to serve four customers isn't being utilized to its full potential. In the bank's case, some of the tellers, like components of a computer system, remain idle.

With a server, as you continue to increase the number of client requests, the overhead of managing the requests also increases. The additional requests take more memory, there is more work for the scheduler, there is only a limited bandwidth for the network, and so on. Eventually, the throughput starts decreasing, indicating the server has reached the knee of the curve. This is the point at which the contention for resources and the overhead of managing the resources causes throughput to go down.

Your server's configuration can affect where the knee of the curve occurs. You can often improve your peak throughput by:

- Adding another CPU.

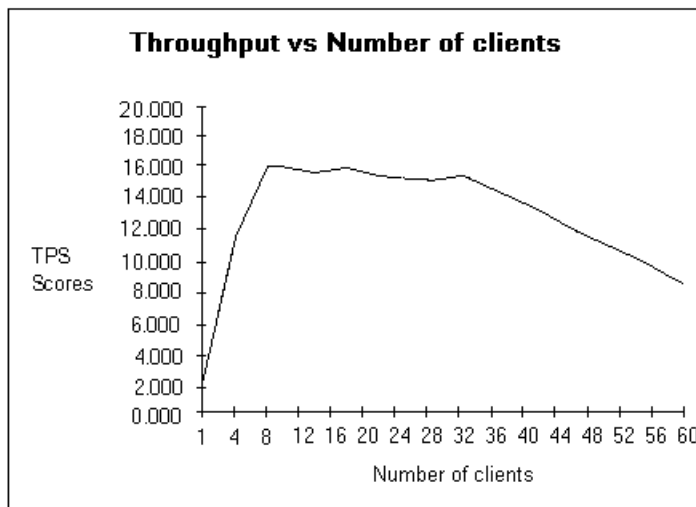
- Increasing the amount of server memory used for the disk cache.
- Using an intelligent network card or disk controller.
- Adding RAM.
- Changing your processor's scheduling.

NOTE: For more information on different components that influence your server's performance, see Chapter 9 “Getting the Best Scores Possible.”

There are three general types of curves that you will see with ServerBench.

- A curve with a single knee that shows a very steep drop in throughput as you increase the client load.
- A curve with a single knee that shows a gradual, sloping drop in throughput as you increase the client load.
- A curve with a double knee. In this type of curve, the throughput reaches its knee and then levels out, just as a plateau does, as you continue to increase the client load. Then, after you've added a certain number of clients, the throughput drops off sharply. A double knee often means there are physical bottlenecks that affect the server's performance. These include the amount of traffic the network can handle, the speed of the disk, the amount of main memory, and so on. Because of these bottlenecks, the server throughput quits going up, but the server maintains a fairly level throughput as it continues to add clients. Eventually, though, after enough clients are added, the throughput will begin a rapid decline. This is the second knee of this curve.

Figure 3-2: A results curve with a double knee



To get a valid measure of your server's performance, you will want to run ServerBench until you see the knee in your server's performance curve. This means you need to keep increasing the client load until the TPS scores begins to level off. In our standard test suites, we add clients incrementally as the tests run (i.e., we start with 1 client, then run the test with 4 clients, next 8 clients, and so on).

If you run the ServerBench tests and don't hit a knee, then you need to increase the intensity of the tests. You can increase the intensity of the tests by:

- Making the tests bigger; for example, increasing the initial size of the test data file from 5 MB to 10 MB. Or, if you have a large data file already, increase the I/O range parameter. As a result, your server will break out of disk cache and hit the disk more often.
- Increasing the Request size parameter so that you read more data off the disk will also cause you to hit the disk more often. In general, the random read disk tests are more likely to make your server break out of cache than the sequential read tests.
- Increasing the Total size parameter also adds pressure to the disk subsystem. For example, if you have a 90 MB cache with 60 clients where each client reads 3 MB, then your server will probably break out of cache with each disk operation.
- Making the mix longer. You can do this by adding more transactions to the mix or just simply adding more tests to the individual transactions.

NOTE: If you make your test bigger, it usually makes it longer as well.

- Reducing the amount of Think time. In general, the smaller the value for Think time, the more the test will stress the server.
- Adding clients.

How to evaluate throughput in terms of your server

As you look at the throughput graphs for different servers, keep in mind that the server with the best peak throughput number may not be the best server for your needs. You need to also consider how the server handles intense loads. ServerBench lets you measure how much your throughput degrades as you increase the load on your server.

Here's an example. Suppose you're comparing servers and Server A reaches a knee at 30 clients and then experiences a sharp drop in throughput. Server B, however, reaches a knee at 25 clients and then begins a very gradual drop in throughput. If you only have 25 to 30 users, then Server A is probably the server you want. But, if you have a workload equivalent to 40 clients, Server B may be the better server for you. This is because, while the throughput on Server B drops off sooner than on Server A, it doesn't degenerate as sharply. This tells you that probably more of your

users will get reasonable service. Since Server A's throughput shows such a sharp drop, you know that if you have a workload equivalent to more than 30 clients, some of those clients will get really bad service.

System throughput versus client response time

When you look at your results, you'll notice that the system throughput as shown by the overall score is different from the throughput for the individual clients. In other words, your system throughput may increase even as your client response time decreases. Don't worry. This is normal.

To understand how this happens, consider the following example. Imagine you are at a bank where each transaction takes ten minutes. There are four tellers and you. From your point of view, the process took you ten minutes (i.e., that is your response time). From the bank's point of view, one transaction occurred in ten minutes; thus, the transaction throughput was one in ten minutes or .1 per minute. Now, go through the same process but this time you are one of four customers. You still have to wait ten minutes, but the throughput is now four in ten minutes or .4 per minute.

Consider this scenario one more time but with eight customers instead of four. The tellers are working harder (probably, they're not asking people how they are any more), so the transaction time is now eight minutes. This means that, if you were at the end of the line, you had to wait sixteen minutes. This response time seems bad to you; however, throughput from the bank's point of view is now eight in sixteen minutes or .5 per minute. In other words, the bank's throughput improved even though your response time increased.

Similarly, suppose you are running ServerBench on a four-processor totally symmetric, multiprocessing system and you execute a test mix that only does processor tests. It takes the first client 150 milliseconds to run the test. It takes the same amount of time for the second client and also for the third client. Thus, from the client's point of view, nothing has changed — the test took the same amount of time for each client. From the server's point of view, it has tripled its throughput. This is a case where the overall system throughput improves but the client response times stay the same.

In evaluating your server, you need to consider both the system throughput and the client response time. You want to determine how evenly your server handles your clients. To check how even the service is on your server, look at the individual client scores in Table 3 of the results workbook.

Processor scaling

If you have a multiprocessor system, then you also need to consider the effect that the number of processors has on your scores. Within a certain range, the more

processors your system has, the better its scores. You will reach a point, though, at which the overhead required to manage more processors outweighs the advantage of having them. Determining how much throughput improves in relation to the number of processors is called processor scaling.

The multiprocessor systems you are likely to use with ServerBench are tightly coupled processors. This means all the processors have an equal access to memory. While access to memory is an advantage, it also results in overhead. The system has to coordinate access to memory. This means the hardware and software spend more time making this processing system work.

In addition, processor scheduling is another source of overhead that all types of multiprocessing systems experience.

Taken together, these factors means there is a point at which the overhead of managing one more processor outweighs the advantage of having another processor.

For example, if you have two processors, you would expect to see your throughput double. However, because of the overhead of managing an additional processor, your throughput may only increase by a rate of something like 1.8, not 2. Even at that rate, there's a definite advantage to having two processors. Eventually, though, as you add processors, your server's throughput rate will start dropping instead of increasing. At that point, you will have passed the peak of processor scaling and are better off not adding another processor. (See Chapter 13 “Figuring Out What Your Results Mean.”)

While bad processor scaling or even a lack of processor scaling is not a problem you are likely to encounter often as you run ServerBench, it is something you should think about as you evaluate your scores and even as you run the ServerBench tests. Processor scaling is especially important when you are comparing servers with different numbers of processors; for example, if you're thinking of buying a new server or upgrading your current server.

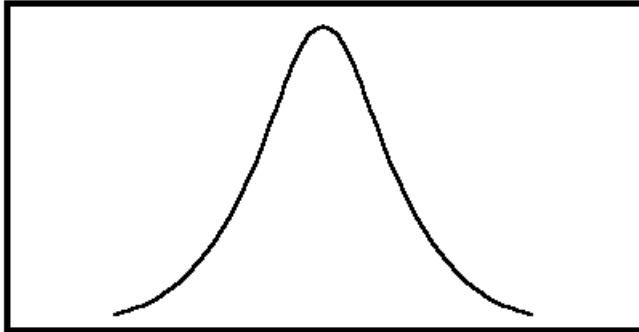
Standard deviation and variance in scores

In reporting your scores, ServerBench evaluates all the test results and then calculates the standard deviation from the mean (or average) of the tests as well the variance (or deviation) from the mean of individual clients. This information tells you how far out clients are from the mean. You can use this information to determine how even the service is on your system in general and what type service the individual clients are receiving.

The standard deviation tells you the range at which most of the clients are getting service; i.e., the normal distribution. Knowing this range gives you an indication of how even the service is on your system. The larger the standard deviation, the more

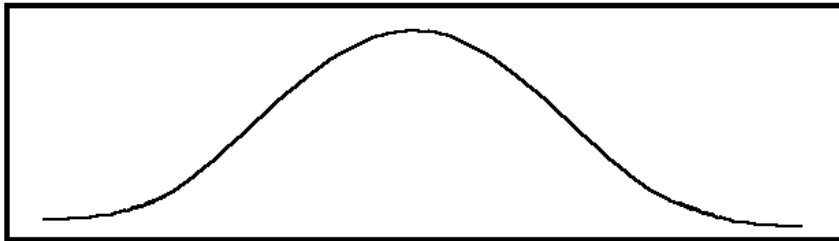
uneven the service is for the clients. For example, if ServerBench plots a graph of the test results and they display as a high bell curve, you know your server is providing even service. This is because the standard deviation is narrow, which means the majority of your clients are getting the same kind of service. Figure 3-3 shows a bell curve with a narrow standard deviation.

Figure 3-3: An example of a narrow standard deviation



If the bell curve is low and flat like a sand dune (as shown in Figure 3-4), then the service on your server is not as even. This is a case where the standard deviation is wide, thus indicating that the service for the clients is varied.

Figure 3-4: An example of a wide standard deviation



To determine how well the service was for a single client, look at that client's variance. This shows how far out from the mean of a test a particular client was. The greater the variance from the mean, then the worse the service for that client was.

ServerBench reports the standard deviation and variance because all clients do not get even service. In fact, the heavier the load, the more likely it is that clients will receive uneven service. In some cases, uneven service causes a client to be starved out and unable to get service. In other cases, the client may have had an incomplete transaction. Since the client never gets a response from the server, it never issues another request. ServerBench does not measure incomplete transactions.

Using a weighted harmonic mean

A harmonic mean is a way of combining several different scores to create a single representative score for that test group. We use a harmonic mean in calculating the overall ServerBench score for your server because it allows us to account for the fact that the transactions contain different tests — the transactions aren't all alike. By using a harmonic mean, ServerBench can combine each client's TPS scores to create one score that reflects how well your server performed.

NOTE: The overall ServerBench score for your server appears in Table 1 of ServerBench's results.

ServerBench weights the different transactions based on how often the clients request that transaction in one iteration of a mix.

Here's a very simple example of how that works. Suppose a mix contains three transactions — A_TRANS, B_TRANS, and C_TRANS. The mix tells the clients to request A_TRANS and C_TRANS one time each during a mix iteration, but it tells them to request B_TRANS three times during a single iteration. This means that B_TRANS has a higher weight than the other two transactions. In other words, the TPS score for B_TRANS will count more heavily when ServerBench computes the overall score using a harmonic mean.

Making repeatability a reality

In a test environment, repeatability of test scores is important. ServerBench's design enables it to ensure a high degree of repeatability. This means that test results from multiple runs of the same test with the same parameters are consistent with each other.

The steps ServerBench takes to promote repeatability include:

- Making each phase of the ServerBench testing distinct. A client completes one aspect of a test and then waits for the other clients to finish. Once all the clients have finished, they all move to the next stage of the test. This ensures that all the clients are at the same state in the testing process. You don't have one client cleaning up while another one is running a test.
- Using a well-defined initialization that is separate from the test. ServerBench does not include the time it spends in the initialization phase as part of the test time.
- Giving you the option of returning the clients and server to as clean a state as possible at the end of each test. You can use a command line option to tell ServerBench to delete all the data files when the test ends. This way, when the next test starts, that test recreates the data files. As a result, the state of the system should be consistent for successive runs of the tests. (By default,

ServerBench uses the same data file for the next disk test. This reduces the amount of time ServerBench spends initializing data files. The system test suites include a large enough ramp up time to compensate for any file caching effects.)

A step you can take to increase the repeatability is to reboot your server between test runs. Doing this returns your system to consistent power-up state and ensures that you get the lowest amount of variation between test runs. This is the method we use at ZD Labs.

End of chapter

Chapter 4

The ServerBench/NetBench Connection

This chapter is for people who find themselves using both ServerBench and NetBench. It notes some of the differences and similarities between working with the two benchmarks. In this chapter you'll find information on:

- When to use ServerBench and when to use NetBench.
- The differences between running ServerBench and NetBench.

Choosing ServerBench over NetBench

Both ServerBench and NetBench give you valuable information about your server. However, they each measure different aspects of a server's performance. ServerBench's tests target applications servers while NetBench's tests zero in on file servers. Whether ServerBench or NetBench is a better tool for evaluating your system depends on the type of work you do on your server.

Table 4-1 gives you a brief comparison of the two benchmarks. The next section gives you more details about the benchmarks.

Table 4-1 A comparison of ServerBench and NetBench

ServerBench	NetBench
Measures application servers.	Measures file servers.
Provides an overall TPS score for your server and individual scores for clients. The scores show how well servers handle client requests for a variety of operations.	Provides an overall I/O throughput score for your server and individual scores for clients. The scores show how well servers handle client requests for network file operations.
Accepts Windows 95 and Windows for Workgroups clients.	Accepts DOS, Windows for Workgroups, and Mac OS clients. The NetBench Windows for Workgroups clients will run under Windows 95 as 16-bit applications.
Requires a server program, a controller program, and a client program.	Requires only a controller program and a client program.
Requires you to provide a specific network protocol.	Runs on top of whatever protocol your system uses.
Uses proprietary programs to communicate only with itself.	Accesses data through a publicly available API.
Runs only on specific server platforms.	Runs on any server that provides shared file access to the controller and clients.
CPU power plays a very large role in how well your server performs.	The server's disk I/O speed and the network I/O speed have the most influence on test scores
Reports scores as TPS (transactions per second).	Reports scores as bytes per second.

Running both ServerBench and NetBench

Because we know that some of you are running both the ServerBench and NetBench tests, we've tried to do what we could to make this dual testing easier.

NOTE: If you want to test both ServerBench and NetBench on your system, you may need to change some settings on your server before you can switch from one benchmark to the other. For example, ServerBench requires that your server be configured as an applications server while NetBench requires that it be a file server. If you use a server operating system (such as Windows NT) that uses different configurations for applications servers and file servers, you'll need to reconfigure your server before you switch between these benchmark programs.

You'll notice a lot of similarities as you look at the two benchmarks. Here's a brief summary of what you'll see:

- **User interface.** Both ServerBench and NetBench use the same basic user interface. A minor difference you might notice as you monitor a test run is that ServerBench waits and updates all the colors in the client grid at the same time. NetBench, however, goes ahead and changes a client's color in the client grid when that client is ready for the next test stage,.
- **Directory structure.** If you run the standard test suites, you must set up a directory structure where NetBench can create its test files. ServerBench's standard test suites create the test files in the ServerBench directory on the server.
- **Standard test suites.** Both ServerBench and NetBench provide standard test suites you can use in testing your system. You'll find these test suites in the subdirectory **SUITES** in controller's installation directory.

The key ServerBench suite is **SYS_60.TST**. (ServerBench also provides other test suites that focus on the disk, processor, and network server subsystems.) The key NetBench test suites are **NBDM_28.TST** and **NBDM_60.TST**. In addition, both ServerBench and NetBench provide a short, demonstration test suite. ServerBench's is **SAMPLE.TST** and NetBench's is **VERIFY.TST**.

- **Testbed.** ServerBench only works with PCs running Windows 95 or Windows for Workgroups. NetBench works with DOS or Windows for Workgroups clients and Mac™ OS clients. (NetBench has client executables for DOS, Windows, and Mac OS clients. If you run Windows client on Windows 95, it will run as a 16-bit Windows application.)
- **Number of clients.** The standard test suites for both ServerBench and NetBench include mixes that add clients incrementally up to a total of 60 clients. ServerBench does this with one test suite. NetBench pairs its standard test suites so that each suite runs eight mixes. The only difference in NetBench's mixes is that for the first test suite, the number of clients

per mix increases from 1 to 28 and for the second test suite, the number of clients increases from 32 to 60.

- Files you provide. Both ServerBench and NetBench require you to provide a client configuration file for each client. These files use the same three parameters: client name, client ID number, client group number.
- Creating test suites. You follow the same basic steps to create a test suite for ServerBench as you do for NetBench.
- Mix parameters. ServerBench and NetBench use the same mix parameters (Ramp up, Ramp down, Length, Think time, Delay, Total Clients, Data pathnames, and Group numbers). However, with ServerBench the only unit of measure for Ramp up, Ramp down, and Length is seconds; NetBench lets you specify whether you want to use seconds or iterations.
- Test parameters. Because ServerBench and NetBench have different tests, they obviously have different test parameters. However, the test parameter Request Size works the same on both benchmarks. Also, ServerBench's test parameter Total Size is equivalent to NetBench's test parameter File size.
- Results. Both benchmarks use Excel to display their test results. If you don't have Excel installed, you cannot view the results. ServerBench uses Excel 5.0 or higher while NetBench uses Excel 4.0 or higher.
- Command lines to start the benchmark programs. Both benchmarks create an icon on the controller that lets you start the controller program. ServerBench also creates icons on the clients that you can use for starting the client program on them. NetBench lets you set up a batch file you can use for starting the client program on the DOS clients and create icons for starting the Windows for Workgroups clients. ServerBench also requires you to enter a command line on the server. NetBench does not run a server program, so there is no server command line to enter. See the section "Starting and using ServerBench and NetBench" for more information about their command lines.
- Running a test suite. The process for selecting, running, and monitoring a test suite is essentially the same for both benchmarks.

Starting and using ServerBench and NetBench

The procedure you follow to start these two benchmarks, run a test suite, and then view your results is essentially the same, as you can see from Table 4-2.

Table 4-2 A look at how ServerBench and NetBench work

Starting ServerBench:	Starting NetBench:
<ol style="list-style-type: none"> 1. Start the ServerBench programs. <ol style="list-style-type: none"> a. Choose the controller icon on the controller to start the controller program. b. Choose the Start Test button at the main ServerBench window. c. Choose the Start button at the controller window. d. Start the server program on the server. (The command line you use depends on your server operating system.) e. Choose the client icon on each client to start the client program. f. Choose OK in the Connect Clients dialog box. 2. At the controller window, answer Yes to the dialog box asking if you want to select a test suite. 3. Choose Suites in the Directory box on the Select Test Suites dialog box. 5. Choose SAMPLE.TST. This is the ServerBench demo test and takes two to five minutes to run. 6. Choose OK. 	<ol style="list-style-type: none"> 1. Start the NetBench programs. <ol style="list-style-type: none"> a. Choose the controller icon on the controller to start the controller program. b. Choose the Start Test button at the main NetBench window. c. Choose the Start button at the controller window. d. Start the client program from each client you want to include in the test. You can set up a batch file to do this. e. Choose OK in the Connect Clients dialog box. 2. At the controller window, answer Yes to the dialog box asking if you want to select a test suite. <ol style="list-style-type: none"> 2a. Choose Add at the Select Test Suites window. 4. Choose Suites in the Directory box on the Test Suite File dialog box. 5. Choose VERIFY.TST. This is the NetBench demo test and takes two to five minutes to run. 6. Choose OK.

<p>7. At the Selected Test Suites window, enter a name for the results file (the default is SAMPLE). Now choose OK. ServerBench will start running the test suite. You can monitor the test run from the controller window.</p>	<p>7. At the Select Test Suites window, choose OK. NetBench will start running the test suite. You can monitor the test run from the controller window.</p>
---	---

To view your ServerBench results:	To view your NetBench results:
<ol style="list-style-type: none"> 1. Go to the main ServerBench window. (You can do this by minimizing the controller window or just clicking on the main window.) 2. Choose View Results. 3. At the Select Results dialog box choose Suites in the Directory box on the Test Suite File dialog box. 4. Choose SAMPLE.TLG. 5. Choose OK. 6. At the View Results window, choose all the Worksheets. (If you don't see the Workbook and Worksheet options, click on the More button to display the rest of the dialog box.) Now click on View OK. 7. Go to Table 1 to see the overall ServerBench score for your server. (ServerBench should automatically open its results spreadsheets to Table 1.) 	<ol style="list-style-type: none"> 1. Go to the main NetBench window. (You can do this by minimizing the controller window or just clicking on the main window.) 2. Choose View Reports. 3. Choose Suites in the Directory box on the Test Suite File dialog box. 4. Choose VERIFY.TST. 5. Choose OK. 6. Make sure you select all the tables in the dialog box that appears and choose OK. 7. Go to Table 3 to see the overall NetBench score for your server. (NetBench should automatically open its results spreadsheets to Table 3.)

End of chapter

Part 2

The Mechanics of Using ServerBench

This is the reference section of your ServerBench manual. It lets you look at all the windows and dialog boxes that ServerBench uses. Then it gives you base information on how you run ServerBench. And finally, it steps you through the process of running a short test suite.

Remember, if you have any questions about the terms we use in these chapters, check the glossary.

Chapter 5

Using ServerBench's Windows and Menus

This chapter explains the screens you see when you run ServerBench on the controller and the clients. Both the controller and each client display graphical ServerBench screens. You use the screens that appear on the controller window to start ServerBench, run tests, monitor the tests as they run on each client, view results, create test suites, and perform other ServerBench tasks. The screens that appear on each client are for information purposes. They let you know which phase of the test each client is in (for example, running).

The main ServerBench window

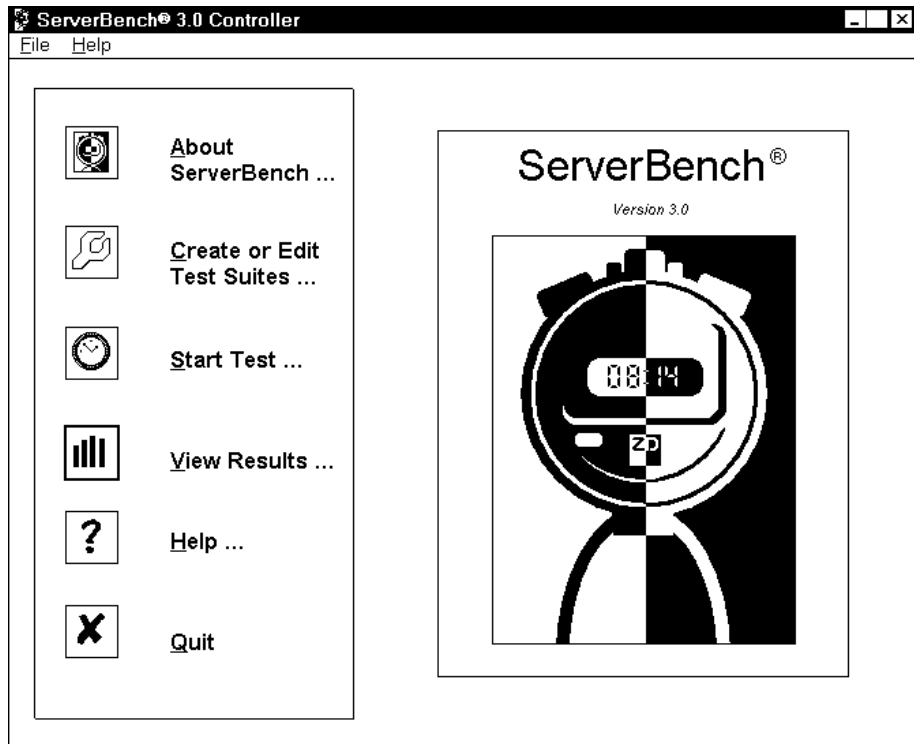
When you start ServerBench on the controller, the main ServerBench window appears. This window contains the ServerBench function buttons. From the ServerBench main window, you can:

- Get information about the ServerBench development team and re-read the ServerBench License Agreement.
- Create and edit test suite files.
- Start a ServerBench test.
- View test results.
- Use the on-line help facility.
- Exit ServerBench.

If you want to get to the controller window, choose the Start Test button. This brings up the controller window on top of the main ServerBench window.

NOTE: The main ServerBench window stays on your screen even when other windows appear.

Figure 5-1: The main ServerBench window



The controller window

The controller window is probably the window where you will spend most of your time. This is the window you use to run the ServerBench test suites and to monitor the clients.

You get to the controller window by choosing the Start Test button (or selecting Start Test from the drop-down File menu) on the ServerBench main window. The controller window then appears on top of the main ServerBench window.

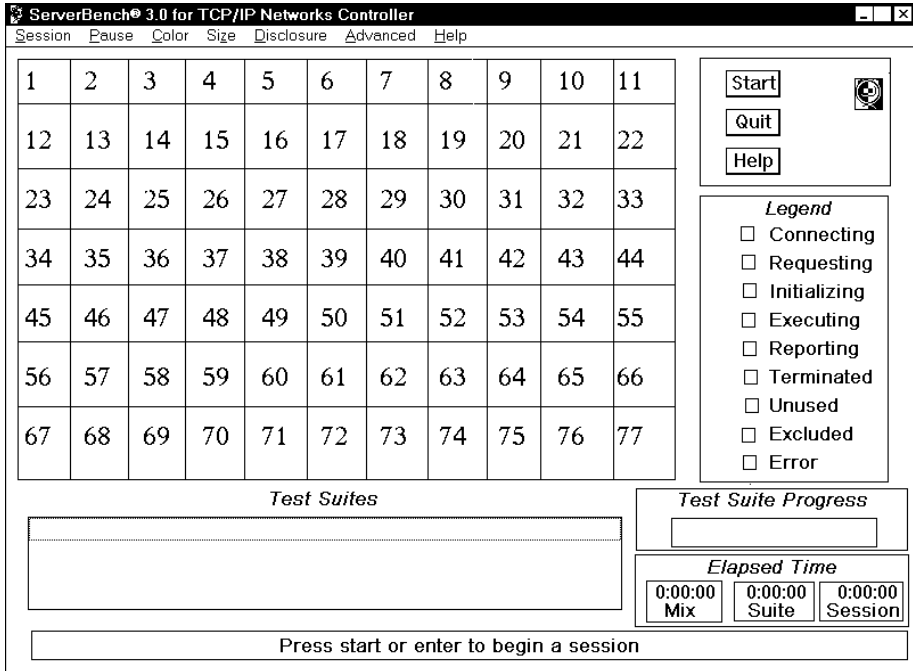
NOTE: If you want to return to the main ServerBench window from the controller window you can:

- Enter the key sequence ALT-TAB. This key sequence switches you between the controller window and the main ServerBench window.
- If any tests are running, choose Quit and select the Abort a test option. This brings you back to the "Do you want to select a test suite" dialog box. Say no.
If no tests are running and you have the "Do you want to select a test suite" dialog box, say no.
- Minimize the controller window by clicking on the down caret in the upper right corner of the window. When you minimize the controller window, ServerBench saves its current state and continues to run any tests that were executing. When you maximize the window, you can view your test suites' status.
- If part of the main ServerBench window is visible, click on it to make it active.
- Use the right mouse button to click on the ServerBench icon in the upper right corner of the controller window.

From the controller window, you can:

- Select a test suite.
- Start a test.
- Halt a running test.
- Add a test suite.
- Specify whether ServerBench pauses between test mixes.
- Monitor the different stages the clients are at in the tests and get other client information.
- Monitor the progress of the test suite you are running.
- Check the status of the test suites you've selected during this test session.

Figure 4-2: The ServerBench controller window



This window contains several key parts:

- The client grid.
- The Legend of client activities.
- The Test Suites box.
- The status information.
- The function buttons.
- The menu options.

NOTE: If your controller's running Windows for Workgroups, you can run your test suites with the controller window minimized. The controller icon displays information about the running tests.

While the controller window is minimized, you can perform other tasks and look at results for mixes that have completed. However, if you try to view the results of a currently running mix while it is reading or writing to a file, ServerBench will abort the running test.

To avoid creating a problem with a running test, you can use the Pause between mixes option in the drop-down Pause menu and view the test results while ServerBench is displaying the Connecting clients box. (Make sure, though, that you *do not* choose the Delete log files option in the View Results dialog box if you plan to run the rest of the mixes in that test suite.)

The client grid and color legend

Probably the first thing you will notice in the controller window is the client grid. This grid contains a square (i.e., button) for each client that is running ServerBench. The number in the grid square corresponds to the client identification number. By clicking on a grid square, you can get information on that client.

Figure 5-3: The client grid in the controller window

1	2	3	4	5	6	7	8	9	10	11
12	13	14	15	16	17	18	19	20	21	22
23	24	25	26	27	28	29	30	31	32	33
34	35	36	37	38	39	40	41	42	43	44
45	46	47	48	49	50	51	52	53	54	55
56	57	58	59	60	61	62	63	64	65	66
67	68	69	70	71	72	73	74	75	76	77

You may also see numbers for clients that are not running ServerBench. There are two reasons for this:

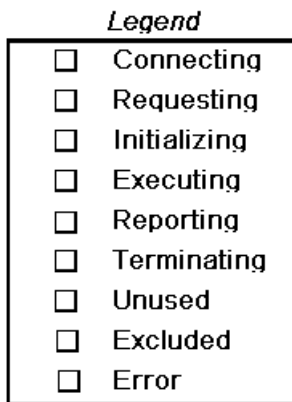
- ServerBench displays all numbers up to the highest identification number of connected client. This means that, if you only connect one client with an identification number of 160, ServerBench also displays numbers 1 through 159 in the grid.
- The client grid has only five sizes – 77, 160, 260, 400, and 1,000 – and displays all the numbers in one of these size ranges, except for 1,000. By default ServerBench displays the grid size that includes the highest

identification number of a connected client. This means that if you only connect one client with an identification number of 165, ServerBench displays a grid with numbered squares that go up to 260.

NOTE: ServerBench does not display the client numbers when you display the grid with 1,000 clients because the grid squares are too small.

Another feature of the grid is that it is color-coded. You can tell at a glance which stage a client is in by looking at the color of that client's square. The Legend of colors located to the right of the client grid tells you which color is associated with which client stage.

Figure 5-4: The color legend in the ServerBench controller window



The legend displays a color for each of the following client stages:

- Connecting. Default color is white.
- Requesting. Default color is yellow.
- Initializing. Default color is light blue.
- Executing. Default color is green.
- Reporting. Default color is dark blue.
- Terminating. Default color is magenta.
- Unused. Default color is gray.
- Excluded. Default color is black.
- Error. Default color is red.

When a client is at one of these stages, its square turns the color associated with that stage. So, if client 65 is running a test, its square is green (or whatever color you assigned to the Executing stage for this test run).

NOTE: Unless a client is connecting or issuing an error message, ServerBench updates the colors for all clients at once. As a result, when you monitor ServerBench's status from the controller, it will probably look like all the clients complete a stage at the same time even though this is not what is really happening. ServerBench updates all the clients at one time to avoid loading the server down with messages during a test.

You can use the drop-down Color menu to change colors associated with the stages. You can also change colors by clicking in the Legend box next to one of the stages. ServerBench will save any changes you make to the colors and then use those colors the next time you start a ServerBench session. (You can return to the default ServerBench colors by selecting the Load Default Colors option from the Color menu.)

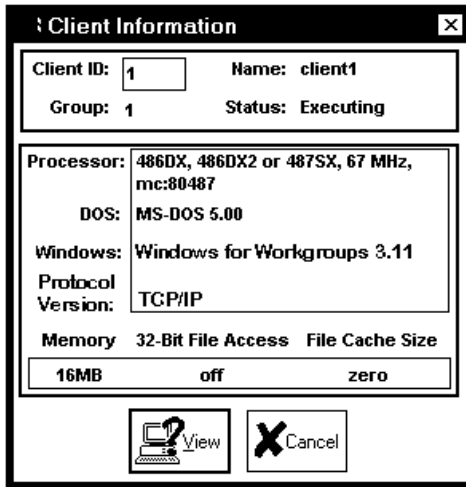
NOTE: ServerBench lets you change the colors in the Legend even when the test is running. This does not affect your server's score.

You can also use the client grid to get specific information on an individual client. When you click on the client's square, ServerBench displays a pop-up box that tells you the client's ID number, its name, its group number, and its status in real time (i.e., at that moment you select it). ServerBench updates the client information each time you select a client's square.

This box also includes the same information that is in the disclosure database about the client's processor, the version of DOS it is using, the version of Windows it is using, the amount of memory, it has, whether its using 32-bit file access, and what its file cache size is.

If you'd like to see the status of another client, enter that client's number in the Client ID portion of this box or click on the appropriate client square.

Figure 5-5: Client information in the controller window

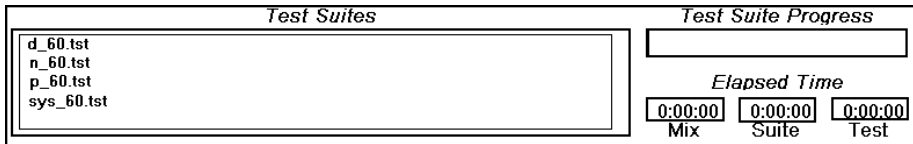


The test suites box and test status information

You can use the information at the bottom of the controller window to monitor the status of the test you are running.

In the box in the bottom left of this window, ServerBench displays the names of the test suites you are running.

Figure 5-6: Information on the test status



In the bottom right of the controller window, ServerBench displays information on the status of the tests you are running. ServerBench tells you:

- How many of the mixes in the test suite have completed. (This is the first number in the box under the heading "Test Suite Progress.")
- The total number of mixes in the test suite. (This is the second number in the box under the heading "Test Suite Progress.")
- The time spent on the current mix.

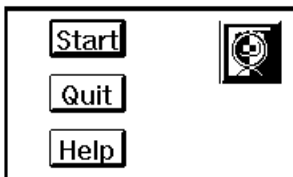
- The time spent on the current test suite. This is the elapsed time spent on all the mixes, not just the time that has passed since you started this test suite. It does not include the amount of time ServerBench may have spent pausing between mixes.
- The total time spent on the test. This is the elapsed time spent on all the tests; it is not just the time that has passed since you started a test (i.e., ServerBench does not include the amount of time it spent pausing between mixes or test suites here). For example, suppose the test started at 2 p.m. and it's now 4 p.m. During this time, ServerBench spent 30 minutes pausing between mixes and between test suites. So the total time on the test is 1.5 hours, not 2 hours.

The controller window's function buttons and menus

The controller window also contains function buttons and drop-down menus you can use.

There are three function buttons in the upper right corner of the controller window. These buttons let you:

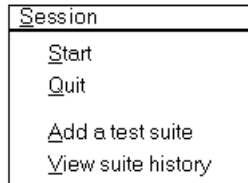
Figure 5-7: Function buttons in the controller window



- **Start.** Use this button to start a test.
- **Quit.** This button lets you interrupt or abort a running test. You can also use this button to return to the main ServerBench window. If any tests are running when you choose Quit and select the Abort a test option, ServerBench displays the "Do you want to select a test suite" dialog box. If no tests are running, choosing Quit causes ServerBench to close the controller window and display the main ServerBench window.
- **Help.** Clicking on this button brings up ServerBench's on-line Help.

There are seven drop-down menus that you can use from the controller window. They are:

- **Session.** From the Session menu, you can:



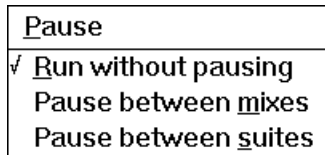
Start a ServerBench test session.

Quit (or abort) the controller window or abort a test.

Add a test suite.

View the test suite history window, which shows you the status of all the test suites you selected during this test session.

- **Pause.** You can select the Pause menu options at any point during a test run. From the Pause menu, you can:



When you specify a pause option, ServerBench runs the test mix or the test suite (whichever you specify) and then waits for you to tell it to continue with the next test mix or test suite. While ServerBench is pausing, you can connect additional clients, add another test suite to the list of selected test suites, or check on any clients that may be hung or displaying an error message.

NOTE: If you are planning on running ServerBench without having someone there to monitor the tests (for example, overnight) *do not* select the pause option. Once ServerBench pauses, it will not continue until you choose OK in the Connect more clients dialog box. (ServerBench will warn you if you try to choose a pause option when you have unattended mode enabled.)

- **Color.** You can use the options in the Color menu to change the colors in the Legend located next to the client grid for your current ServerBench session. ServerBench will then use the colors you set until you change them. If you want to return to the ServerBench default colors, you can select the option Load Default Colors. Here are your options in the Color menu:

<u>C</u> olor
Change <u>c</u> onnecting color
Change <u>r</u> equesting color
Change <u>i</u> nitializing color
Change <u>e</u> xecuting color
Change <u>r</u> eporting color
Change <u>t</u> erminating color
Change <u>e</u> xcluded color
Change <u>e</u> rro <u>r</u> color
Load <u>D</u> efault Colors

- **Size.** The Size menu lets you specify the number of squares ServerBench displays in the client grid. Your options are 77, 160, 260, 400, or 1,000. By default, ServerBench displays the grid that includes the highest client identification number.

NOTE: When you display the grid with 1,000 squares, ServerBench does not display the client numbers; the grid squares are too small.

<u>S</u> ize
Display <u>1</u> 000 clients
Display <u>4</u> 00 clients
Display <u>2</u> 60 clients
Display <u>1</u> 60 clients
<input checked="" type="checkbox"/> Display <u>7</u> 7 clients

- **Disclosure.** If you run a test suite without having "Generate client info each suite" checked, then ServerBench won't save any client disclosure information. In addition, ServerBench won't display Table 5 of the results workbook because it won't have any disclosure information.

NOTE: We recommend that you always have this option selected when you run ServerBench.

The Get Client Information gives you the same information you get when you click on a client's square.

<u>D</u> isclosure
Get Client Information
<input checked="" type="checkbox"/> <u>G</u> enerate client info each suite

- **Help.** You can use the Help menu to start ServerBench's Help.

<u>H</u> elp
<u>G</u> eneral <u>P</u> ort Specific
<u>A</u> bout

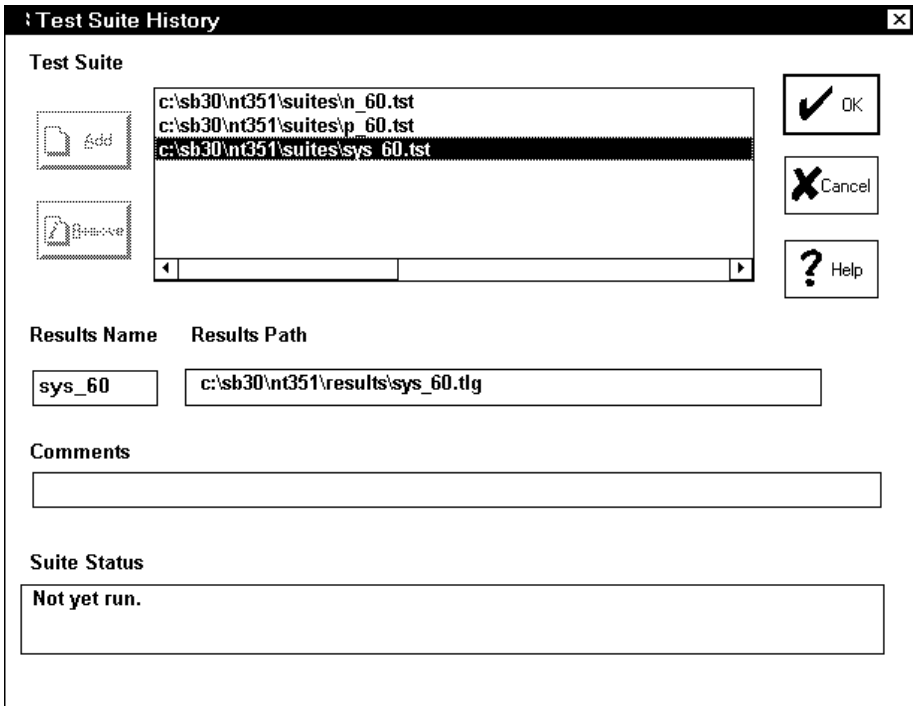
The General help contains most of the information you'll find in this manual. The Port Specific help contains most of the information you'll find in the on-line ServerBench manual for your server operating system. The About option takes you to the ServerBench About screen.

The Test Suite History window

You can use the Test Suite History window to check on the status of all the test suites you've selected during the current test session. When you click on a test suite name in this window, ServerBench displays the path name for that suite's results file. It also displays any comments you entered about that test suite as well as information about whether ServerBench skipped any mixes in that suite. If you request the status of a running test suite, ServerBench displays the status in real time; i.e., the current status of that test suite at that moment.

You can get to this window by choosing View Suite History from the drop-down Sessions menu in the controller window or by clicking on a test suite in the Test Suites box on the controller window.

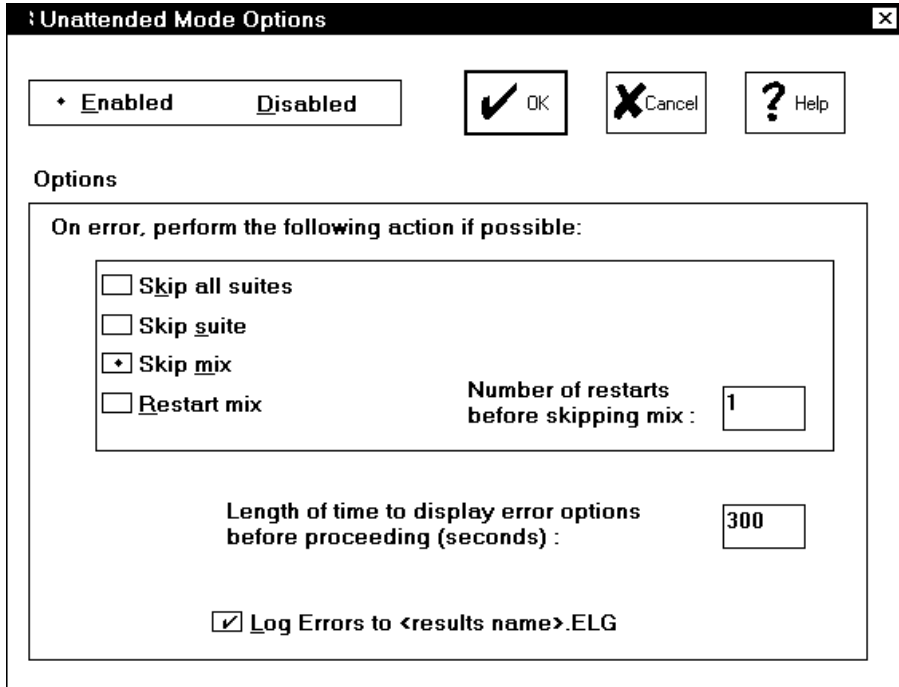
Figure 5-8: Test Suite History window



The Unattended Mode dialog box

ServerBench displays the Unattended Mode dialog box when you choose the option Unattended Mode from the Advanced menu in the controller window.

Figure 5-9: The Unattended Mode dialog box



At this dialog box, you can:

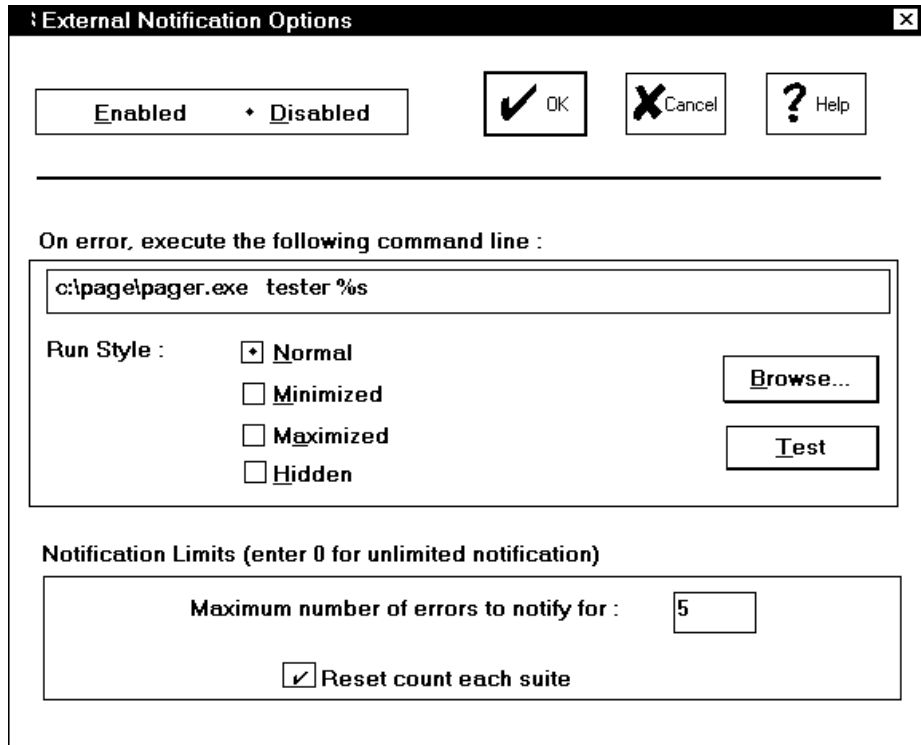
- Enable or disable unattended mode. (By default, unattended mode is enabled.)
- Skip all test suites if an error occurs. This will end that test run.
- Skip the current test suite. ServerBench will start running the next test suite, if there is one.
- Skip the mix that is reporting the error. This is ServerBench's default.
- Restart the mix reporting the error. If you select this option, you need to specify how many times you want ServerBench to try to restart the mix before skipping the mix. ServerBench resets the restart counter each time it starts a mix.
- The amount of time you want ServerBench to wait before taking the action you specified. (We recommend that you set this to at least 60 seconds if you have external notification enabled. The paging program needs time to connect and disconnect.)
- Write the error message to an error log (.ELG) file. By default, ServerBench writes the message to a file that uses the name <results-name>.ELG and saves this file in the controller's RESULTS subdirectory (or the results directory you

specified when you selected the test suite). ServerBench only writes error messages to this log file when you have it running in unattended mode.

The External Notification dialog box

The External Notification dialog box appears if you choose that option from the Advanced menu.

Figure 5-10: The External Notification dialog box



When you enable this dialog box, you can enter a command that the controller will execute if an error occurs. This command line can start any program, such as an executable, a **.bat** file, a **.pif** file, or an **.exe** file program. If you specify a pager program that dials a specified pager number and that program accepts alphanumeric pagers, you can actually send the error message to the pager.

If you include **%s** argument in this command line, the controller will replace that argument with the error message. For example, if you're executing a pager program that supports alphanumeric pagers, you can get it to display the error message when it pages you by entering a command line similar to:

```
c:\pager.exe tester %s
```

This command line tells the controller to execute the pager program and display the error message.

To see what will happen when the controller executes the command line, choose the Test option. This option lets you try the command line before starting the test suite. If you include %s in the command line, the string you'll see when you choose Test is:

```
SeverBench 3.0: Testing external notification  
feature... Success!
```

You can also specify the run style you want the controller to use when it executes the program you specified. You can select any of the standard Windows run styles: Normal, Minimized, Maximized, or Hidden.

In addition, you can tell the controller the maximum number of times you want it to execute this command line. If you enter 0 (i.e., unlimited), the controller will execute the command line you entered each time an error occurs. However, if you enter a number greater than 0, then, once the error count exceeds the notification count, the controller will no longer execute the command line.

If you like, you can also tell the controller to reset the notification count when it starts a new test suite.

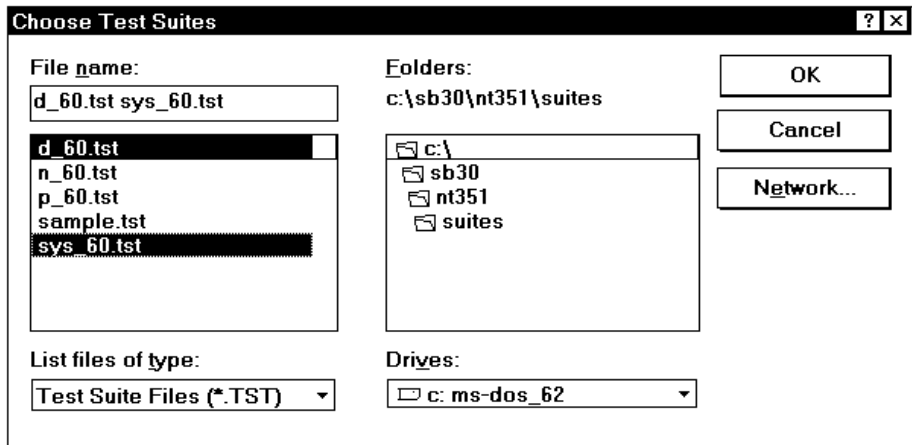
The test suite selection windows

Before you actually run a test suite, you must select it. ServerBench displays several windows to aid you in selecting the test suites you want to run.

When you tell ServerBench you want to choose a test suite to run, it displays the Choose Test Suite dialog box. Go to the directory that holds the test suite (or suites) that you want and highlight the names of the test suites you want to run. You can choose as many test suites from this window as you like. To select a range of test suites, click on the first suite and then, holding down the shift key, click on the last suite in the range. To randomly select test suites, click on the first test suite and then hold down the control (Ctrl) key each time you click on another test suite. Once you've selected the test suites you want, click on OK.

NOTE: When you select several tests, Windows arranges their names in the File name box in alphabetical order, not the order that you selected them.

Figure 5-11: The Choose Test Suites dialog box

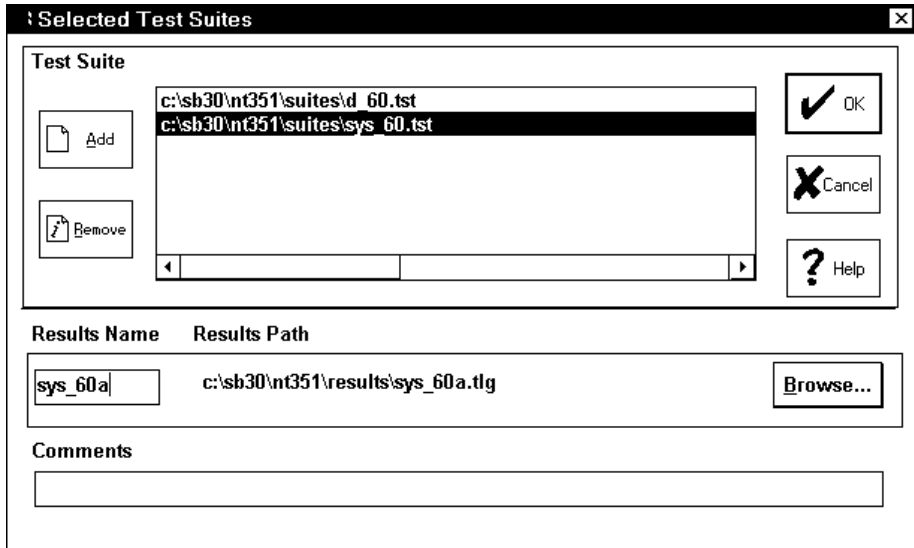


NOTE: You'll find ServerBench's standard test suite files in ServerBench's **SUITES** subdirectory in the controller's installation directory.

ServerBench takes you to the Selected Test Suites dialog box, which lists the suites you've already selected. Two special features in this window are that it lets you enter a path name for your results file and any comments you want to make about running this test suite. (If you don't enter a path name, ServerBench by default uses the test suite name with a **.TLG** extension. It places the path name in the **RESULTS** directory on the controller.)

This dialog box also provides you with an easy way to change the name of the results file so that ServerBench doesn't overwrite your previous results. You can type in a name for the results file in the Results Name box. If you want to change the path name to the results file, you can choose the Browse button. Each time you choose Browse, ServerBench displays the Select Results Path dialog box (see Figure 5-14). When you choose a path name in this box, ServerBench enters that path as the path name to your results file.

Figure 5-12: The Selected Test Suites dialog box



From this window you can also:

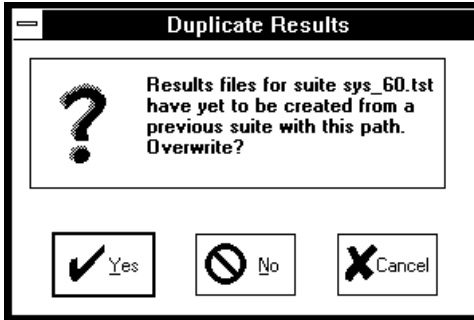
- **Add** more test suites to your current list of test suites. When you choose the Add button, ServerBench displays the Choose Test Suites dialog box. ServerBench always places the test suite (or suites) you're adding after the test suite that is currently highlighted.
- **Remove** one of the currently selected test suites. If you decide you don't want to run all the test suites you've chosen, you can use the Remove button to take the suites you don't want out of the list.

Once you have the suites listed that you want to include in this test run, click on OK.

Duplicate results names

ServerBench warns you if you've entered a name for your results file that another file is already using. For example you might see the warning that appears in Figure 5-13.

Figure 5-13: Duplicate Results warning box



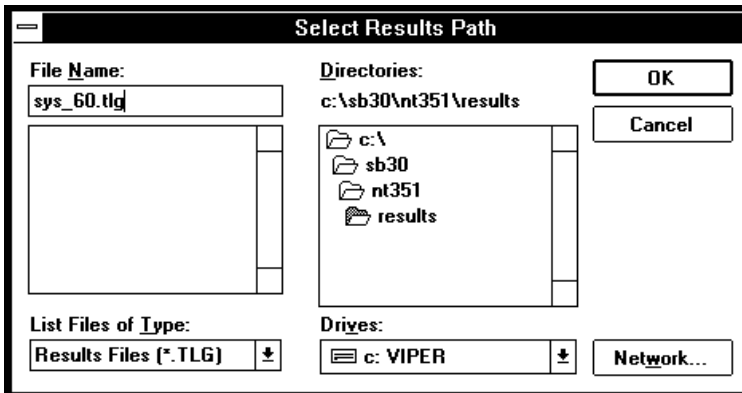
This warning means that you've selected two test suites (in this example, **sys_60.tst**) that have the same results path name. However, neither test suite has run yet. ServerBench also displays warnings if you enter the name for an existing results file.

Each time ServerBench displays a warning box, it gives you the option of choosing:

- Yes, which means you want to overwrite the first set of results with the second.
- No, which means you want to enter a new path name for the results file. (This is the same as choosing Browse in the Selected Suites dialog box).
- Cancel, which simply returns you to the Selected Suites dialog box.

When you choose NO, ServerBench displays the Select Results Path dialog box.

Figure 5-14: Select Results Path dialog box



Under file name, ServerBench displays the current name of the results file. You can use this dialog box to specify a different directory for your results file. Because

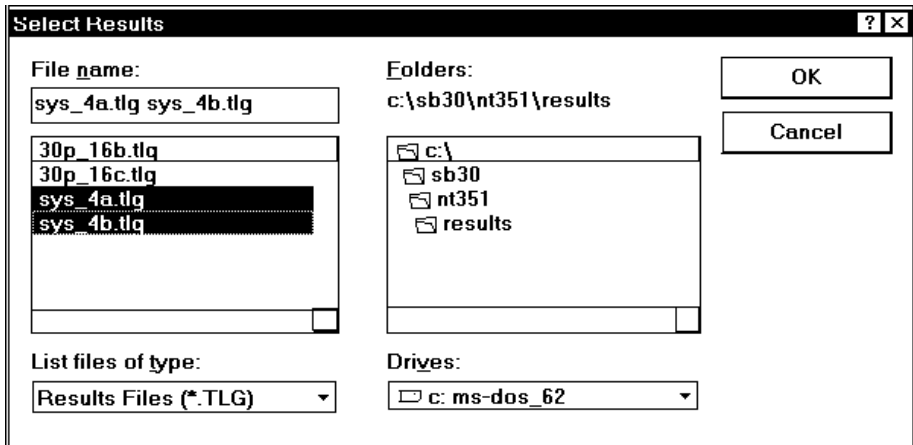
ServerBench uses sticky directories, it'll place the next results file you specify in this directory as well (unless you use the Browse button to specify a different directory).

The results dialog boxes

When you choose the View Results button from the main ServerBench window, ServerBench displays several dialog boxes.

The Select Results dialog box appears first. Go to the directory that holds your results and highlight the names of the results files you want to see. Each results file ends with a **.TLG** extension. You can choose as many results files from this window as you like. To select a range of files, click on the first file and then, holding down the shift key, click on the last file in the range. To randomly select results files, click on the first test suite and then hold down the control (Ctrl) key each time you click on another file. Once you've selected the files you want, click on OK.

Figure 5-15: The Select Results dialog box



NOTE: By default, ServerBench places all the results files in the **RESULTS** subdirectory on the controller.

ServerBench now displays the Selected Results dialog box.

The View Results dialog box

You have lots of options in the View Results dialog box. For example, you can:

- Work with your disclosure database. From this dialog box you can select and edit a disclosure database snapshot file.
- Edit the disclosure logs ServerBench created and tell ServerBench to use them as the basis for the Sever and Client Disclosure tables in the Results workbooks.
- Select which results tables you want to look at. ServerBench will only create results tables for the categories you select.

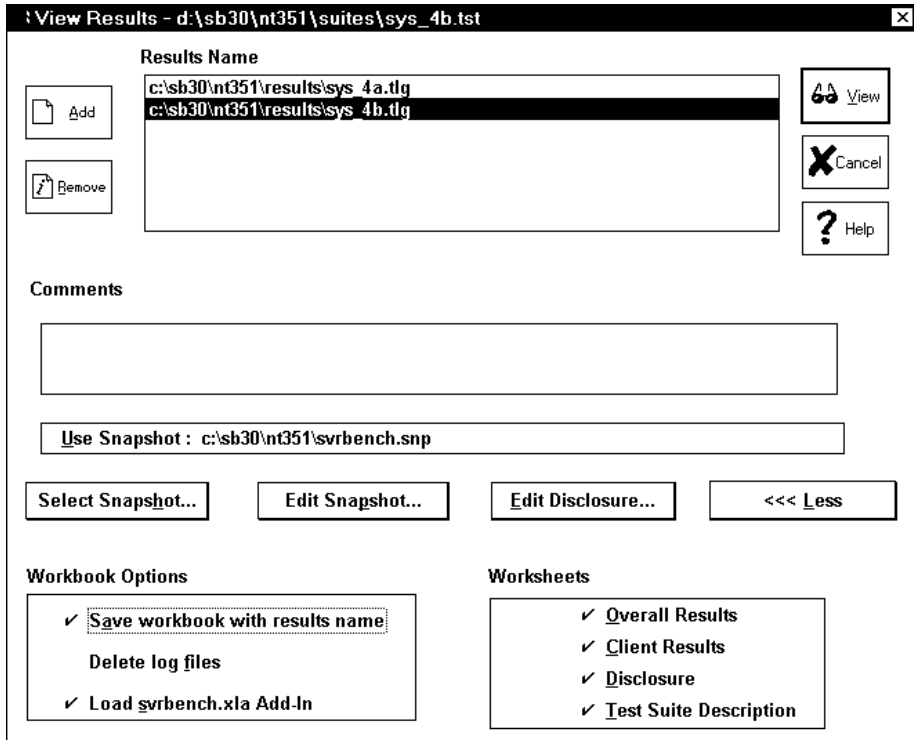


Tip:

You can easily see which test suite you ran to get your results. Highlight a set of results in the Results Name box. ServerBench displays the name of the test suite associated with that set of results in the dialog box title bar.

Each option you select in this dialog box applies to all the results files listed under Results Name.

Figure 5-16: View Results dialog box



Here's what you can do with the buttons in this dialog box:

- **Add** more results files to your current list of files. When you choose the Add button, ServerBench displays the Select Results dialog box.
- **Remove** one of the currently selected results files. If you decide you don't want to view all the results files you've chosen, you can use the Remove button to take the files you don't want out of the list.
- **Use Snapshot**, if checked, tells ServerBench to use the snapshot file displayed in this box as the basis for creating the Server Disclosure and Client Disclosure results tables. If you don't check this option, ServerBench uses the information in the **.DLG** logs to create those tables.
- **Select Snapshot** lets you choose a different database snapshot file. ServerBench uses the information in the database snapshot file to create the Server Disclosure and Client Disclosure results tables. You need to supply the information in these tables if you want to publish your results. You can create

one snapshot file that contains all the information on your test setup and then use it for each set of results as long as your test setup stays the same.

- **Edit Snapshot** lets you modify the server and client disclosure information stored in the snapshot file specified by the Use Snapshot option.
- **Edit Disclosure** lets you modify the information that ServerBench captured about your server and clients. When you run a test suite, ServerBench places information it captures about the server and the client in the **.DLG** log files that it creates for that set of results. ServerBench displays this information in the Server and Client Disclosure forms when you choose Edit Disclosure. If you also have the Use snapshot option checked, ServerBench includes the information from that snapshot file in these forms as well as the **.DLG** information. You can add the **.DLG** information to your current snapshot file by checking option Update snapshot file... at the bottom of the Server and Client Disclosure forms. If you don't check this option, ServerBench won't save the information on the disclosure forms to the snapshot file.
- **Less/More** lets you choose how much of the View Results dialog box ServerBench displays. By default, ServerBench only displays the top portion of this dialog box; it doesn't display the Workbook and Worksheet options. If you choose More, ServerBench displays the entire dialog box and the name on the button switches to Less. Then, when you choose Less, ServerBench displays the abbreviated dialog box.

In addition to setting up your disclosure information in this dialog box, you can also tell ServerBench how to set up your results workbooks. You can choose:

- Which workbook options you want. Under Workbook Options, ServerBench lets you specify:
 - a. **Save Workbook with results name.** When checked, this option tells ServerBench to automatically save each set of a results to an Excel workbook that has the same name as the results file. For example, if your results are called **SYS_4A.TLG**, Excel would save them to a workbook called **SYS_4A.XLS**.
 - b. **Delete Log files.** When you check this option, ServerBench will delete the results log files it uses to generate the results tables after it creates the results spreadsheets. Then, if you don't save the spreadsheet using Excel (or didn't tell ServerBench to automatically save the spreadsheet), you won't be able to view those results again. In addition, if you didn't tell ServerBench to create all the results tables for that set of results, you won't be able to generate them later.
 - c. **Load svrbench.xla Add-in.** When you check this option, ServerBench automatically loads its **SVRBENCH.XLA** Add-In module for Excel. This module lets you create a TPS summary graph and a variance summary graph. It also enables you to easily print the graphs or tables from within

the ServerBench workbooks without printing other forms of data. When you select this option, ServerBench adds a ServerBench option under Excel's Data menu.



Tip:

While you can permanently install the **SVRBENCH.XLA** module through Excel itself, we recommend that you install this module from the View Results window. This way, it won't affect your daily Excel work. If you install the module permanently in Excel, you'll notice that your Excel start-up time is slower even when you're not generating ServerBench results.

- Which results tables you want to view. When you check the category, ServerBench displays those tables when it creates the results workbooks. Your options here are:

Summary	Table 1 – ServerBench Summary. With this table, you'll also find the Test Information box and the TPS graph of your server's results.
Overall Data	Table 2 – Overall ServerBench Data
Client Data	Table 3 – Client Data
Disclosure	Table 4 – Server Disclosure Table 5 – Client Disclosure
Suite Definition	Table 6 – Test Suite Information Table 7 – Transaction Information.

Once you have all the results files you want to view and have set up database and workbook options, click on View and ServerBench will display your results.

NOTE: If you have a Excel session currently running, ServerBench will use that session of Excel to display the results workbooks. Otherwise, it will start a session of Excel.

The creating and editing test suites screens

When you choose the Create or Edit Test Suites button from the main ServerBench window, ServerBench displays a series of dialog boxes.

Create or Edit Test Suites dialog box

The first dialog box to appear is the Create or Edit Test Suites dialog box.

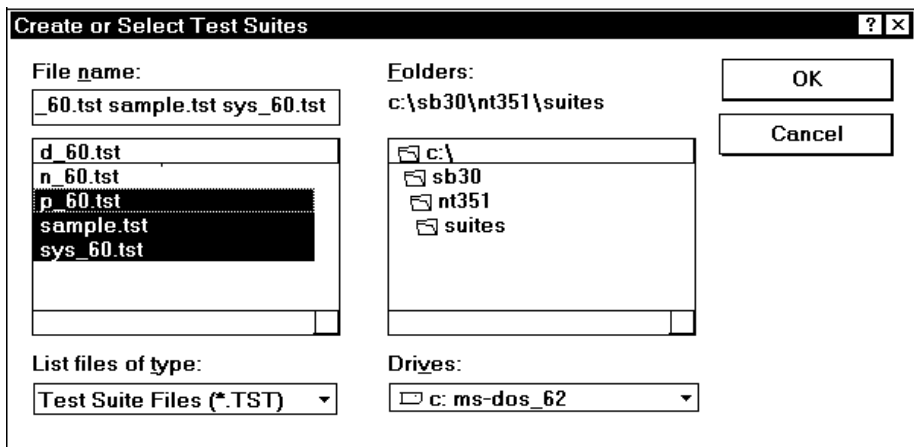
If you want to modify an existing test suite, go to the directory that holds the test suite (or suites) that you want and highlight the names of the test suites you want to edit. You can choose as many test suites from this window as you like. To select a range of test suites, click on the first suite and then, holding down the shift key, click on the last suite in the range. To randomly select test suites, click on the first test suite and then hold down the control (Ctrl) key each time you click on another test suite.

If you want to create a new test suite, enter the name of the test suite in the File name box. ServerBench will ask you if you want to create a test suite file.

Once you've selected the test suites you want, click on OK. ServerBench displays the Mixes in Suite dialog box.

NOTE: When you select several tests, Windows arranges their names in the File name box in alphabetical order, not the order that you selected them.

Figure 5-17: Create or Edit Test Suites dialog box

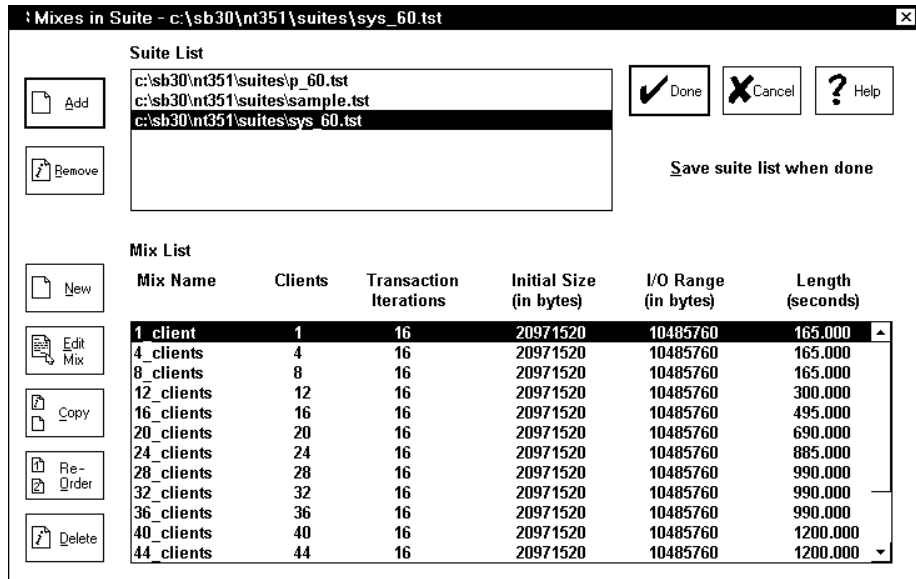


The Mixes in Suite dialog box

The Mixes in Suite dialog box offers you a number of options. It displays information on each mix in a test suite. It also lets you reorder the mixes and add mixes from other test suites. In addition, it lets you get to the Mix Definition window where you can create new mixes or modify existing mixes..

When you highlight a test suite in the Suite List, ServerBench displays all the mixes in that test suite in the Mix List. ServerBench tells you the values of some of the parameters that are set for that mix, such as the mix name, how many clients it has, how many transaction iterations the mix performs, the initial size of each client's data file, the value of the I/O range parameter, and the value of the Length parameter.

Figure 5-18: Mixes in Suite dialog box



NOTE: If you are creating a new test suite file, the list of test mixes in this window will be empty.

This dialog box also includes numerous function buttons. Here's what you can do with the buttons in this dialog box:

- **Add** more test suites to this list. When you choose the Add button, ServerBench displays the Create or Edit Test Suites dialog box.

- **Remove** one of the currently selected test suites. If you decide you don't want to edit all the test suites you've chosen, you can use the Remove button to take the suites you don't want out of the list.
- **New** takes you to the Mix Definition window and lets you create a mix. ServerBench places this new mix after the currently highlighted mix in the test suite.
- **Edit Mix** displays the highlighted mix in the Mix Definition window. You can then change the values for that mix.
- **Copy** lets you copy one or more mixes from another test suite into the currently highlighted test suite. It inserts the mixes you're adding after the highlighted mix in the Mix List. When you choose the Copy button, ServerBench displays the Select Test Suites dialog box. Once you select a test suite and click on OK, ServerBench displays the Copy Mixes from... dialog box. Choose the mixes you want to copy by highlighting them. Now click on OK. ServerBench inserts the mixes you've highlighted into your current test suite. For more information, see the section "Copying mixes from an existing test suite" later in this chapter.
- **Reorder** lets you change the order of the mixes in the current test suite. When you choose the Reorder button, ServerBench displays the Reorder Suite Mixes dialog box. For more information, see the section "Reordering mixes" later in this chapter.
- **Delete** removes the mix you have highlighted from the test suite file.

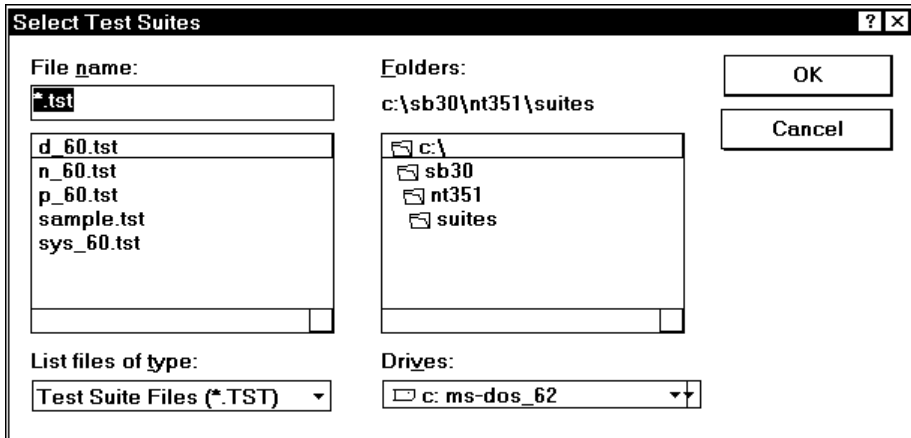
Adding existing mixes to a test suite

When you choose the Copy button from the Mixes in Suite dialog box, ServerBench lets you specify an existing test suite. It then displays all the mixes in that suite. You can highlight the mixes you want to add to your current test suite in the Mixes in Suite dialog box, and ServerBench will place those mixes after the current mix in the Mix List.

NOTE: If you just want to copy some of the information from one mix directly to the Mix Definition window for you to edit, use the Copy fields... option in the drop-down Advanced menu in the Mix Definition window. See the section "Copying mix information into the Mix Definition window" later in this chapter.

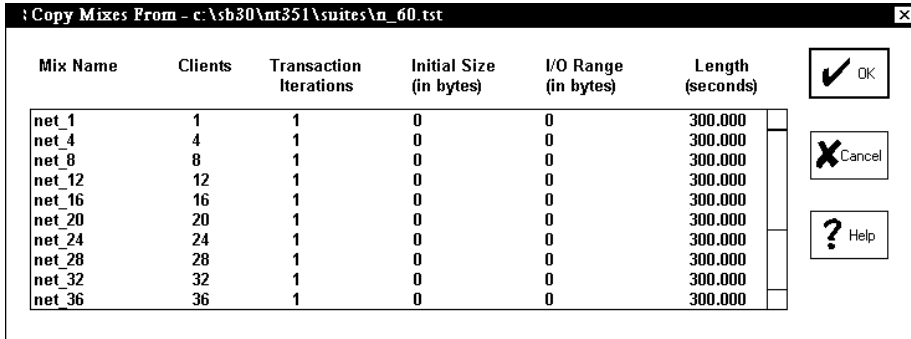
ServerBench displays the Select Test Suites dialog box when you choose the Copy button.

Figure 5-19: The Select Test Suites



Go to the directory containing the test suite you want and select that suite. When you press OK, ServerBench displays the Copy Mixes From dialog box.

Figure 5-20 The Copy Mixes From... dialog box



This dialog box displays information about the mixes in that test suite. It gives you the mix names, the number of clients in each mix, the number of transactions each mix performs, the value for the Disk test file initial size parameter, the value for the I/O range parameter, and the value of the Length parameter.

Highlight the mix or mixes you want to copy and then click on OK.

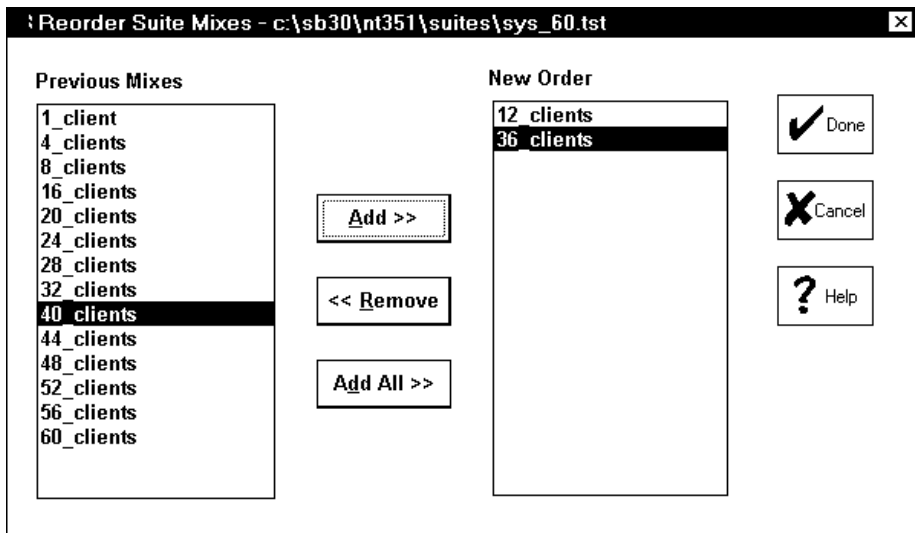
ServerBench returns you to the Mixes in Suite dialog box and places the mixes you've chosen in the Mix List after the highlighted mix.

The Reorder Suite Mixes dialog box

When you choose the Reorder button in the Mixes in Suite dialog box, ServerBench displays the Reorder Suite Mixes dialog box. This dialog box contains a list of all the mixes in the currently selected test suite. You can move these mixes one at a time or in groups to the New Order box. Highlight the mix (or mixes) you want to move and then click on the Add button. ServerBench inserts the mixes in the New Order box. If you already have some mixes in the New Order box, ServerBench places the ones you're adding after the highlighted mix. You can repeat this process as often as you need to.

To remove a mix from the New Order list, simply highlight it and then click on the Remove button. ServerBench places the mix back in the list of previous mixes.

Figure 5-21: Reorder Suite Mixes dialog box



The Mix Definition window

You get to the Mix Definition window from the Mixes in Suite dialog box. To bring up the Mix Definition window, you can choose either the New button or the Edit Mix button. You can also get to the Mix Definition window by double-clicking on

a test suite name in the Suite List or on a mix name in the Mix List.

From the Mix Definition window, you can create mixes or modify existing mixes. There are multiple parts to this window.

Figure 5-22: The Mix Definition window

The screenshot shows the 'Mix Definition' window with the following sections:

- Mix Name:** 'Mix 2 of 16' and a text field containing '4 clients'.
- Duration Information (sec.):** Ramp Up: 50.000, Ramp Down: 15.000, Length: 165.000.
- Timing Information (sec.):** Delay: 0.000, Think: 0.000.
- Transaction Definitions:** A table with columns: Transaction Iterations, Transaction Name, Test Iterations, Test Type, Request Size (bytes), Total Size.

Transaction Iterations	Transaction Name	Test Iterations	Test Type	Request Size (bytes)	Total Size
1	TR1	1	CS	500	500
		15	P	N/A	50
		2	RR	2048	2048
		15	P	N/A	50
		2	RW	2048	2048
3	TR2	1	SC	2048	2048
		1	CS	500	500
		7	P	N/A	50
		2	RR	2048	2048
		7	P	N/A	50
2	RW	2048	2048		
1	SC	2048	2048		
- Client Information:** Total Number of Clients: 4. Client# Data file pathnames: data1, data2, data3, data4.
- Buttons:** OK, Cancel, Help, Previous Mix, Next Mix.
- Footer:** Enter in a name for the mix up to 15 characters in length.

At this screen you enter the following mix information:



Tip:

We explain all these parameters in Chapter 10 "Creating Your Own Test Suites."

- **The Mix name.** This is the name ServerBench displays when it lists the mixes in a test suite. Enter an alphanumeric name that is up to 15 characters long.

- **Duration Information.** Define the following parameters:

Ramp up. Enter the amount of time in seconds for the ramp up period. ServerBench ignores test results from mix iterations that start during the Ramp up period (which occurs at the beginning of a mix). This way you avoid having your test results skewed because the server load at the beginning of the test was very light.

Ramp down. Enter the amount of time in seconds for the ramp down period. ServerBench ignores test results from mix iterations that start during the Ramp down period (which occurs at the end of a mix). This way you avoid having your test results skewed because the server load at the end of the test was very light.

Length. Enter the total amount of time in seconds you want ServerBench to run. The value for length needs to be greater than the sum of the values for Ramp up and Ramp down and all the transaction iterations; otherwise, ServerBench will never record any results. In general, the value for Length is the minimum amount of time the test will run. If ServerBench is in the middle of a mix iteration when the Length value expires, ServerBench will continue the test until it finishes the current iteration. (One iteration of a mix means that each client executes each transaction in the mix.)

- **Timing Information.** Define the following parameters:

Delay. Enter the maximum amount of time in seconds (or fractions of seconds) that you want a client to wait before starting a test once the controller tells the clients to start. When you set the Delay time parameter, each client takes that value and generates a pseudo-random number. This way the Delay time can vary for each client. As a result, when you set Delay time, you stagger the clients' initial requests to the server instead of overloading the server with a burst of requests at the beginning of each mix. The default value for Delay is 0, which is the value ServerBench's standard test suites use.

Think time. Enter the number of seconds or fractions of a second a client will wait after receiving a transaction response before it sends another request. The default value for Think time is 0, which is the value ServerBench's standard test suites use. The smaller the value for Think time, the more the test stresses the server.

- **Transaction Definitions.** Define the following parameters:

Figure 5-23: The transaction portion of the Mix Definition window

Transaction Definitions					
Transaction Iterations	Transaction Name	Test Iterations	Test Type	Request Size (bytes)	Total Size
1	TR1	1	CS	500	500
		15	P	N/A	50
		2	RR	2048	2048
		15	P	N/A	50
		2	RW	2048	2048
		1	SC	2048	2048
3	TR2	1	CS	500	500
		7	P	N/A	50
		2	RR	2048	2048
		7	P	N/A	50
		2	RW	2048	2048
		1	SC	2048	2048

Disk test file initial size (MB):

Disk test file I/O Range (MB):

Transaction iterations. Enter the number of times you want ServerBench to execute that transaction during one iteration of the test mix.

Transaction Name. Enter the name of the transaction in the box under this heading.

Test Iterations. Enter the number of times you want ServerBench to repeat the test during the transaction.

Test Type. Enter the mnemonic for the test you want to include in this transaction in the box under this heading. You can see a list of mnemonics by choosing the down arrow next to this box.

Request size. For the disk and network tests, you must enter the amount of data in bytes that you want ServerBench to move as a unit at one time.

Total size. For the disk and network tests, enter the total amount of data in bytes that you want ServerBench to move during that test. For the processor, enter the total number of iterations you want that test to perform.

Disk Test file initial size (MB). Enter the size you want each client's Disk test data file to be when ServerBench creates it on the server.

Disk Test file I/O Range (MB). Enter the size that you want ServerBench to use for the I/O range. This value specifies how much of the disk file ServerBench will use when performing disk operations.

New. Use the New button to create a transaction or to add another test to the current transaction. If you want to add a test to an existing transaction, you'll need to tab over to the Test Iterations field. If you enter any information in the Transaction Iteration or Transaction Name fields, ServerBench will create a new transaction.

Delete. Use the Delete button to remove a highlighted test.

- **Client Information.** Define the following parameters:

Total number of clients. Enter the number of clients that you want to participate in this test mix.

Client # and Data file path names. ServerBench uses the data file path name to determine where to create the test data files for the disk read and write tests. You can use these boxes to edit the data file path name. If you enter a path name that ends in an asterisk (*), ServerBench will automatically append a number to that path name and enter it as the path name for each client up to the maximum number of clients. So, if you specified 60 clients, and you entered the path name **test***, ServerBench would create the path names **test1**, **test2**... up to **test60**.

Figure 5-24: The client portion of the Mix Definition window

Client Information

Total Number of Clients:

Client# Data file pathnames:

<input type="text"/>	<input type="text"/>
1	data1
2	data2
3	data3
4	data4

◀ ▶

A special feature of the Mix Definition window is that you can scroll through all the mixes in the test suite. In the Mix Name section of this window, ServerBench tells you which number this mix is in the test suite out of the total number of mixes. To move between the mixes, just choose the Previous or Next buttons.

The Mix Definition window also provides several drop-down menus that you can use in working with mixes. These menus are:

- **Mix.** From drop-down menu you can:

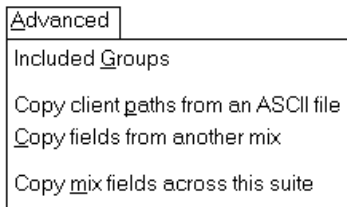
Mix

- Insert new mix
- Delete current mix

Insert new mix. This option lets you create a new mix for this test suite. The new mix always comes after the currently displayed mix.

Delete current mix. This removes the highlighted mix from the test suite.

- **Advanced.** From drop-down menu you can:



Included Groups. When you choose this button, a pop-up dialog box appears that lets you select the group numbers you want included in this test mix. You can use the buttons in this dialog box to select all the clients or to clear all the group specifications you currently have entered.

Copy paths from an ASCII file. With this option, the Copy Paths From A File dialog box appears. At the dialog box, enter the name of the ASCII file where you list the path names for your clients. When you create this file, you should give it a **.PTH** extension. In the file, list the path name for each client on a separate line. For example, depending on the delimiters your operating system uses, you might enter something like:

```
C : \CLIENTS\CLIENT1
C : \CLIENTS\CLIENT2
C : \CLIENTS\CLIENT3
. . .
C : \CLIENTS\CLIENT60
```

When you specify a directory in the path name, make sure that directory exists before you execute the test.

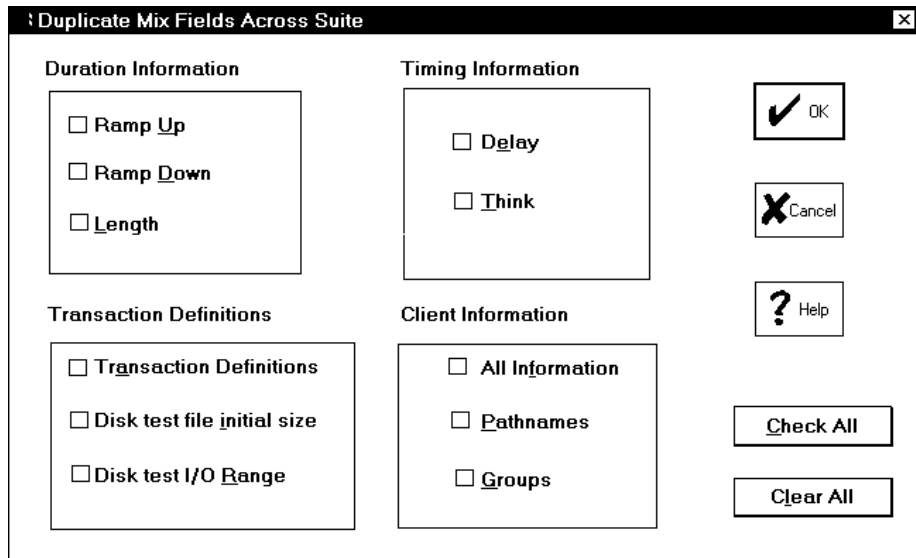
Copy fields from another mix. You can use this option to copy all or part of the information from an existing mix into the Mix Definition window. This option first displays the Copy Test Suite File dialog box and then the Copy a Mix dialog box. For more information on using this option, see the section “Copying mix information between test suites” later in this chapter.

Copy mix fields across this suite. When you choose this option, ServerBench displays the Duplicate Mix Fields Across Suite dialog box. For more information, see the section “Duplicating mix fields across a test suite” later in this chapter.

Duplicating mix fields across a test suite

The drop-down Advanced menu in the Mix Definition window contains the option Copy mix fields across this suite. When you choose this option, ServerBench displays the Duplicate Mix Fields Across Suite dialog box. This dialog box lets you copy the parameter values in the current mix to all the mixes in that test suite. You can copy the value for one parameter or multiple parameters.

Figure 5-25: Duplicate Mix Fields Across Suite dialog box



The options you can choose at this dialog box you are:

- **Duration information.** You can specify the Ramp up, Ramp down, and/or Length parameters.
- **Timing information.** You can choose the Think and/or Delay parameters.
- **Transaction definitions.** If you choose the Transaction Definitions option, ServerBench replaces the transactions in the other mixes with the transactions in the current mix. However, it does not affect the client information in the mixes. Make sure you want all the transactions to be the same in all the mixes before you choose this option.

If you choose the Disk test file initial size and/or the Disk test I/O range, ServerBench updates all the mixes so that they have the same value as the parameter (or parameters) you've chosen.

- **Client information.** If you choose All information, ServerBench makes the client information in all the mixes identical. You will now have the same number of clients in each mix with the same path names. Suppose you have three mixes in a test suite. Mix1 has 8 clients, Mix2 has 16 clients, and Mix3 has 32 clients. You're currently editing Mix2. If you choose All information, then each mix in this test suite will have 16 clients. All the path names and group designations in the three mixes will match the path names and group designations you specified in Mix2.

To change only one part of the client information without affecting the number of clients in the mix, choose either Pathnames or Groups. If you choose the Pathnames option, ServerBench will change only the client path names in the other mixes and only for as many clients as the current mix has. In other words, if the current mix is Mix2 with 16 clients, ServerBench would only change the first 16 path names in Mix3. The path names for clients 17 through 32 would remain untouched.

A word of caution about duplicating mix fields

ServerBench's Duplicate Mix Fields Across Suite feature can be very handy, but you need to keep in mind how the settings you copy can affect the other mixes in the test suites. Here're some things to watch out for:

- **Null pathnames and blank Disk test parameters.** If you're copying information from a Processor test or a Network test to mixes that include Disk tests, make sure you don't copy the client path names, the value for the Disk Test File Initial Size parameter, or the value for the Disk test I/O Range parameter. Because the Processor and Network tests don't use those parameters, you'd be copying null pathnames and blank values for the two Disk test parameters. As a result, if you copy this information to a mix containing Disk tests, you'll be creating an invalid mix.

The opposite is also true. If you choose only the option Transaction Definitions when you're at a mix containing Disk tests and you're copying this information to mixes containing Processor or Network tests, you'll again be creating invalid mixes. This is because ServerBench will add the transaction information (i.e., the Disk tests) to the mix containing the Processor and Network tests, but it won't copy any of the client information. So, once more, you'll have disk tests with no client path names or values for either the Disk Test File Initial Size parameter or the Disk test I/O Range parameter.

- **Dependent fields.** If you're only changing one parameter, you need to consider how that parameter works with other parameters. It's possible that, if you globally update only one parameter across all the mixes, you may cause an error condition for another mix. For example, suppose you change one field, such as Length, without changing the other fields that work with Length — Ramp up and Ramp down. Your current mix (Mix 1) uses Ramp up and Ramp

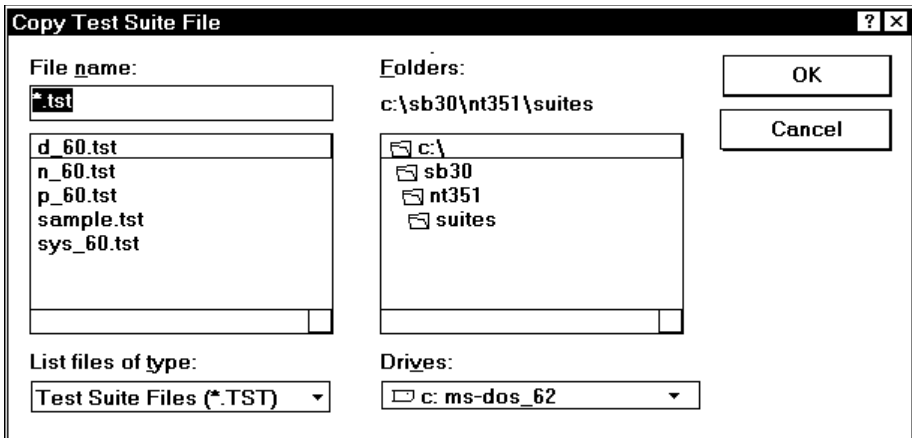
down periods of 30 seconds and a Length of 90 seconds. Mix2 uses Ramp Up and Ramp Down values of 1 minute each. If you tell ServerBench to change the Length value for the mixes to match the value for Mix1, you'll create an error condition for Mix2 because the Length of the test will in Mix2 no longer exceed the combined values of Ramp Up and Ramp Down.

Copying mix information into the Mix Definition window

You can use the Copy fields from another mix option in the Advanced menu to replace the current values in the Mix Definition window with the values in an existing mix. When you choose this option, ServerBench lets you copy either one entire mix or the values you specify.

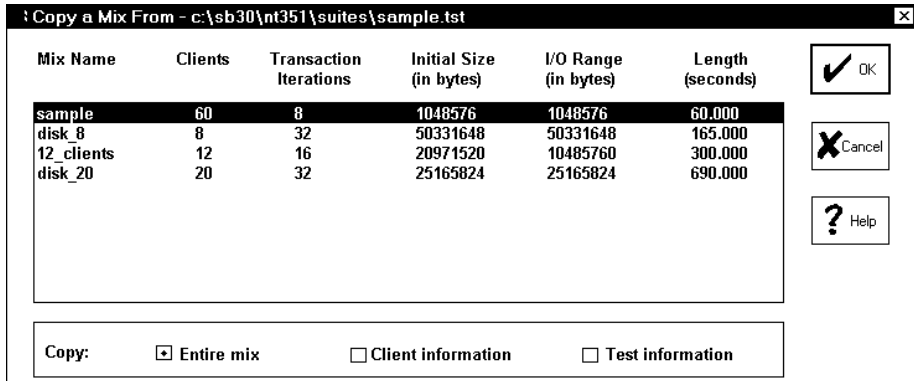
After you select the Copy fields from another mix option, ServerBench displays the Copy Test Suite File dialog box. Highlight the test suite you want to use. You can only select one test suite.

Figure 5-26: Copy Test Suite File dialog box



Once you click on OK, the Copy a Mix from dialog box appears.

Figure 5-27: Copy a Mix from... dialog box



This dialog box displays information about the mixes in that test suite. It gives you the mix names, the number of clients in each mix, the number of transactions each mix performs, the value for the Disk test file initial size parameter, the value for the I/O range parameter, and the value of the Length parameter.

You now have three options

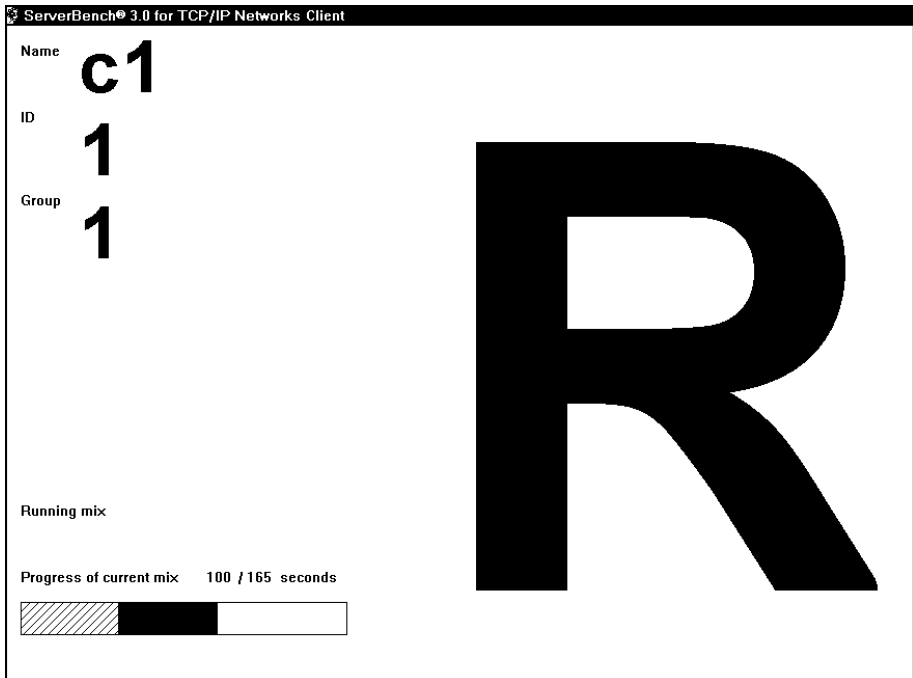
- **Entire mix.** ServerBench copies all the values in the selected mix to the Mix Definition window.
- **Client information.** ServerBench copies only the client information to the Mix Definition window.
- **Test information.** ServerBench copies only the transaction definitions to the Mix Definition window.

NOTE: If you want to copy multiple mixes at one time, use the Copy mixes button on the Mixes in Suite dialog box. (See the section "Adding existing mixes to a test suite" earlier in this chapter.)

The Client window

ServerBench also displays a Client window on each client. You can use the information in this screen to monitor the state of each client that is running a test.

Figure 5-28: The Client window



The key feature of the Client window is a very large letter indicating the client's state. The purpose of the letter is to let you see at a glance the stage the client is in. This ability can be very handy when you're running a large number of clients. For example, when the client is running a test mix, ServerBench displays the letter "R" in the Client window. The client states are:

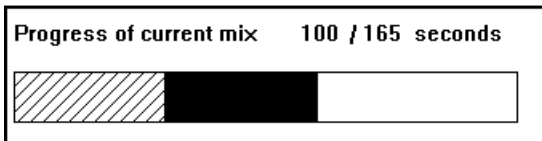
- **B** Blocked. ServerBench blocks a client between states to prevent one client from advancing to the next stage before the other clients are ready. For example, if Client 1 completes the test first, ServerBench will block Client 1 from sending its results until all clients have completed the test.
- **I** Initializing. The client is initializing the current test mix.
- **R** Running. The client is running the current mix.

- **S** Sending. The client is sending its log to server.
- **D** Done. The client has finished sending its log to server.
- **X** Excluded. The client is excluded from the mix.
- **E** Error. An error has occurred.

This screen also displays the client's identification number and group number. At the bottom of the screen, the client displays its current status or any error messages.

In addition, the Client window includes a bar graph that shows the progress of the current mix. When the bar is full, the mix is complete. ServerBench uses hash marks in the bar to let you know when the client is executing transactions that it is not counting (i.e., transactions that start during the Ramp up or Ramp down period). When the client is recording transactions, ServerBench fills the bar with a solid black color. For example, in Figure 5-29, the current mix has been running for 100 seconds and is scheduled to run at least 65 seconds more. Almost one-third shows hash marks, indicating that those transactions that have occurred so far began during Ramp up, so the client did not record them. The solid black color tells you that the client is now recording the transactions that complete.

Figure 5-29: Sample progress bar that appears in the Client window



ServerBench displays the Client window on the monitor of each client that is running the benchmark.

End of chapter

Chapter 6

Starting, Stopping, and Interrupting ServerBench

The goal of this chapter is simply to serve as a reference point for the mechanics of starting and stopping ServerBench and running a test suite.

In this chapter you'll find information on:

- Starting ServerBench on the controller, server, and the clients.
- Disconnecting ServerBench from the clients.
- Exiting from ServerBench.
- Interrupting a test run.
- Using Help.

Starting ServerBench

Starting ServerBench involves four basic steps:

1. Make sure all the machines you are using are turned on. Both the server and controller must be running before you start any ServerBench programs. You also need to have the clients that you want to use in the tests running (but you can wait and turn them on when you are ready to choose the client command lines). For a quick run of ServerBench, you may only want to have three or four clients.

In addition, your network software must be up and running before you start ServerBench on the controller.

2. Start ServerBench on the controller.

3. Start ServerBench on the server.
4. Start ServerBench on the clients.

**Tip:**

You must always start ServerBench on the controller first, on the server second, and on the clients third. If you don't start the ServerBench programs in this order, ServerBench won't start.

To start ServerBench, you need to actually go to each of these machines and click on an icon or enter a command line.

NOTE: If you're not planning to run test suites, you don't have to start ServerBench on the server and clients. ServerBench does not require these machines if all you want to do is create test suites and view existing results from the controller.

The following sections step you through starting ServerBench on the controller, the server, and the clients.

Starting ServerBench on the controller

To start ServerBench on the controller:

1. Start the ServerBench controller program.

On a Windows 95 controller, choose:

```
Start -> Programs -> Ziff-Davis Benchmarks ->  
ServerBench 3.0 Controller
```

On a Windows for Workgroups controller, choose the controller icon from the Ziff-Davis Benchmarks program group.

2. When the main ServerBench window appears, choose the Start Test button.
3. When the controller window appears, choose the Start button.

**Tip:**

If you get either of the following error messages when you press Start Test at the main ServerBench window or Start in the controller window:

- File Error: Cannot Find winsock.dll
- Error 4: Cannot initiate network connection

you probably have an uninstalled TCP/IP stack. You'll need to install your TCP/IP stack before you can run ServerBench.

Starting ServerBench on the server

To start ServerBench on the server:

1. Go to your ServerBench directory on the server.
2. Enter the correct command line for ServerBench on your server operating system. (In most cases you can start the server executable by entering **SVR**. However, there may be other command parameters you need to enter or you may have a different starting command. To determine the correct server command line, see the on-line ServerBench installation manual for your server operating system.)
3. Once you've started the server program, return to the controller. When the server has connected with the controller, the controller will display a pop-up box telling you to start the clients. (If you don't get this box, then something is wrong. Halt the controller and server programs, reboot your machines, and then start again.)

Starting ServerBench on the clients

To start ServerBench on the clients, you must go to each client individually. At the client:


1. Start the ServerBench client program

On a Windows 95 client, choose:

Start -> Programs -> Ziff-Davis Benchmarks -> ServerBench 3.0 <TCPIP or NW41>

On a Windows for Workgroups client, choose the controller icon from the Ziff-Davis Benchmarks program group.

The Client window will appear.



Tip:

If you get the one of the following error messages (either in a pop-up box or on the client screen) when you start ServerBench on a client:

- File Error: Cannot Find winsock.dll
- Cannot initialize TCP/IP stack

you probably have an uninstalled TCP/IP stack. You'll need to install your TCP/IP stack before you can run ServerBench.

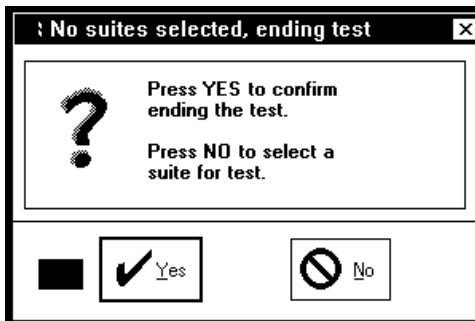
2. Repeat step 1 for every client you want to run in the test.
3. After ServerBench is running on each client, return to the controller. You'll notice that ServerBench has highlighted a square on the client grid for each client you connected. Choose OK in the Connect Clients dialog box.

Disconnecting the server and clients

ServerBench automatically disconnects the server from the controller when you:

- Answer No at the Select Test Suites pop-up box. When you answer NO, ServerBench displays the following pop-up box asking you to confirm that you want to end this test session. If you answer Yes here, the clients disconnect from the server and the server disconnects from the controller.

Figure 6-1: End of test pop-up box



- Close the controller window, (for example, by choosing the Close option from the System menu).

When the server disconnects from the controller, the clients automatically disconnect from the server if no tests are running.

Once the server, controller, and clients disconnect, you won't be able to run another test suite until you restart ServerBench on the server and the clients.

You can avoid disconnecting the controller, server, and clients and still perform other ServerBench tasks by:

- Entering the key sequence ALT-TAB. This key sequence switches you between the controller window and the main ServerBench window.
- Minimizing the controller window. You minimize the controller window by double-clicking on the down caret in the upper right corner of the window. You can minimize the controller window while a test suite is running. ServerBench will continue to run any test suites you specified.

Exiting from ServerBench

When you finish running your ServerBench test suites, you can either minimize ServerBench so that it is an icon or you can exit from ServerBench.

If you want to quit ServerBench and you're at the controller window:

1. Choose No at the Select a test suite pop-up box.
2. Choose Yes at the pop-up box that asks if you want to end the test.
3. Now choose the Quit button.

When you do this, ServerBench disconnects the server and the clients. If the main ServerBench window is open, you'll need to close it as well. You can do this by:

- Choosing the Quit function button from the main ServerBench window.
- Choosing Exit from the File menu at the main ServerBench window.
- Using the key sequence ALT-F4.
- Choosing Close from the system menu.

NOTE: If you are reviewing your results, you return to the main window by choosing Exit from the drop-down File menu in Excel.

Interrupting a running test suite

You can interrupt or abort a running test suite without disconnecting your server and clients. To interrupt a test, make sure you're at the controller window and:

1. Choose Quit. You can either click on the Quit button or select Quit from the drop-down Session menu.
2. When the pop-up box appears, choose the option Skip all Suites.

When you choose Abort Test, ServerBench halts whichever test is running and returns control to the controller. You can then set up another test suite or choose to quit ServerBench.

ServerBench does not save any results for the test mixes that were running during a suite when you abort a test suite.

In addition to aborting the running suite, the Quit dialog box lets you:

- Skip a mix.
- Skip a test suite.
- Restart a mix.

If you skip a mix or a test suite, ServerBench does not save results for that mix or test suite, but it does save results for the other mixes and test suites it ran.

If you choose to restart a mix, ServerBench saves only the test results from the most recent run of that mix.

Using ServerBench's on-line Help

If you encounter a problem while running ServerBench or want more information about a ServerBench feature, you can use ServerBench's on-line Help. To start Help, you can choose either the Help button in ServerBench's main window or Help from the drop-down Help menu. You can access Help from most of the windows on the controller.

NOTE: You must be at the controller to access Help. ServerBench does not provide on-line Help on the server or the clients.

ServerBench's main Help is based on this manual and the on-line installation manual for your server operating system. When you start Help, it displays a list of chapters from this manual. You can go down the list and select the topic that you want information on. ServerBench displays that chapter. Once you are in that chapter, you can get more information by selecting the appropriate chapter subtopics.

ServerBench gives you the following Help options in the main ServerBench window:

- **General.** This starts the primary on-line Help, which contains most of this manual.
- **Port Specific.** This starts another version of Help, which contains most of the information in the on-line ServerBench manual for your server platform. You can access this Help by using the drop-down Help menu on the main ServerBench window.
- **About.** This displays ServerBench's About screen. From this screen you can choose to re-read the ServerBench License Agreement or to view the Credits, which list the names of the members of the ServerBench development and documentation teams.

End of chapter

Chapter 7

Quickly Running a Sample Test Suite

Internally, ServerBench is a complicated benchmark. As a result, ServerBench can appear intimidating. However, you can actually run ServerBench easily and produce a general score for each mix you run on your server. To help you see this, ServerBench comes with a short, sample test suite that takes about two minutes to execute. By running this special ServerBench test, you will get a feel for how ServerBench works. This chapter steps you through the basics of:

- Executing the sample test suite.
- Looking at your results.

Remember, this is just a quick summary. For the details, you'll want to go to:

- Chapter 8 “Running ServerBench's Test Suites.”
- Chapter 12 “Let's Take a Look at Those Results.”
- Chapter 13 “Figuring Out What Your Results Mean.”

If you have not yet installed ServerBench, see the on-line ServerBench manual for your server operating system.

Running ServerBench's sample test suite

ServerBench provides a short test suite that you can run just to get a feel for how ServerBench executes. This test suite, **SAMPLE.TST**, contains one small mix of tests and takes about two minutes to run. You can then open the results spreadsheet and see what the total TPS was for the mix on your server. You'll find **SAMPLE.TST** with the rest of ServerBench's standard test suites in the controller's **SUITES** directory.



Tip:

If you want to get the details about the transactions in this test suite, you can view the test suite in the Mix Definition window. See Chapter 10 “Creating Your Own Test Suites” for information on going to the Mix Definition window.

You’ll need to have ServerBench running on the controller, the server, and the clients before you can execute **SAMPLE.TST**. (See Chapter 6 for the steps you perform to start ServerBench on the controller, server, and clients.) Once you start ServerBench on the clients, ServerBench displays the following message in a pop-up dialog box in the controller window:

Do you want to select a test suite?

This is your cue to choose **SAMPLE.TST**. Follow these steps:

1. When the Choose Test Suites dialog box appears, go to the **SUITES** subdirectory, which contain **SAMPLE.TST** and highlight that test suite. The dialog box opens to this directory by default.
2. Highlight **SAMPLE.TST** and choose OK.
3. At the Selected Test Suites dialog box, you can enter a name for the results file and enter any comments in the comments box. For example, you might name the results file **SAMPLE1** and enter the comment: *Trial run of ServerBench*. ServerBench will save your results as **SAMPLE1.TLG** in the controller’s **RESULTS** subdirectory.
4. Choose OK. ServerBench will start running the test suite. You can monitor the test run from the controller window.
5. When the test suite ends, go to the main ServerBench window and choose the View Results function button.
6. At the Select Results dialog box, choose **SAMPLE1.TLG** and click on OK.
7. The View Results dialog box appears. Select the following options from this dialog box:
 - Click on the More button. This tells ServerBench to display the rest of the dialog box options.
 - Under Workbook options, choose both Save workbook with results name and Load **svrbench.xls** Add-in module.
 - Under Worksheets, make sure each option is checked. Doing this tells ServerBench to display all the results tables.

NOTE: Because this is just a trial run of ServerBench, don't worry about selecting a database snapshot file. ServerBench uses this information to setup the server and client disclosure tables (tables 4 and 5). If it doesn't have a snapshot file, it uses the information it was able to capture about the server and clients.

8. Choose View. ServerBench launches Excel (if it's not already running) and displays your results as tables in an Excel workbook. You'll find the overall score for your server in Table 1: ServerBench summary.
9. To exit from ServerBench, quit Excel. Now choose the Quit button from the main ServerBench window (if the controller window is still open, you'll need to close that window).

Saving the results

When you choose "Save workbook with results name," ServerBench automatically saves a copy of your results tables in a file called **SAMPLE1.XLS**. If you don't select this option, you can use the Excel Save as feature to save your results to an Excel spreadsheet that you can open at any time.

End of chapter

Part 3

Executing ServerBench's Test Suites

The focus of this part of the manual is on testing. The first chapter tells you how to execute a test run. Next you'll find information on how to get the best scores possible. After that you can learn how to set up your own test suite. And finally, you can read about the individual tests themselves.

Chapter 8

Running ServerBench's Test Suites

This chapter tells you how to run the ServerBench tests. It contains information on:

- The basic steps involved in running a test suite.
- What you should be aware of before you start a ServerBench session.
- How long you should allow for a test suite to run.
- Selecting test suites.
- What options are available to you as you run the tests.
- Running another set of test suites.
- Disconnecting ServerBench from the clients.

You always execute test suites from the controller window. Naturally, you'll need to start ServerBench first on the controller, then on the server, and finally on the clients before you can run a test suite. This chapter explains the options you have for running a test suite. For a list of the steps involved, see Chapter 7 "Quickly running a Sample Test Suite."

Before you run your test suites

Here are some points to keep in mind as you use ServerBench. Some of them are just for your information; others can affect your test scores.

-
- For best results, run ServerBench on a isolated network. (For more information, see Chapter 9 “Getting the Best Scores Possible.”)
 - Don’t have any unnecessary background applications running when you execute ServerBench. (For more information, see Chapter 9 “Getting the Best Scores Possible.”)
 - To avoid overloading the file server and causing clients to drop off the network we recommend that you scale up the client count (i.e., run test mixes that start with a small number of clients and gradually add more the way the standard ServerBench test suites do). This approach allows the server to scale its internal resources accordingly as the load increases.
 - ServerBench maintains its connection with the server and clients only as long as you are selecting test suites to run or running test suites. When you answer No at the Select Test Suites pop-up dialog box that appears each time your test run finishes, you lose the connections between the controller, server, and clients. (ServerBench will give you a chance to confirm this step before it actually stops running on the server and clients.)
 - You can perform other tasks at the controller while the tests are running. By pressing ALT-TAB, you can switch to the main ServerBench window where you can check your results or create a new test mix. (Pressing ALT-TAB again returns you to the controller window.) You can minimize the controller window while a test suite is running and perform other ServerBench tasks, such as viewing the results from a different test suite or creating a test mix. You can change the color Legend on the controller window while a test is running. You can even leave a test running and go home.

NOTE: Don’t do anything, though, that affects the currently running test suite. For example, if you attempt to view the results for a mix before it finishes executing, ServerBench will abort the test and will not save any test results for it. To check a mix and avoid aborting your test run, choose the Pause between mixes option. When ServerBench pauses, press ALT-TAB to return to the main ServerBench window. Now you can look at the results or use that mix definition file to create a new file, and so on.

Allowing enough time to run a test suite

The amount of time ServerBench takes to run a test suite on your server depends on a variety of factors. They include:

- The test suite you choose to run. If you select one test suite and specify that it just run for 30 seconds, you'll finish pretty quickly regardless of how fast your server is. However, if you create a test suite that uses a lot of clients and performs a lot of disk accesses, the test will take longer.
- How many clients are in the test.
- How large the test data files are that the clients are using. This is the amount of space the test data files take up on the server. Also, is your server performing a lot of work in file cache or is it breaking out of cache?
- How fast your server is.
- How fast your network is.
- Whether you're running on a test network or production network. If you're running ServerBench on a network that users are working on (even if you have two servers), you'll notice some bottlenecks. (We don't recommend that you run ServerBench on a production network.)

At ZD Labs, it takes about four and a half to five hours to run ServerBench's standard system test suite (**SYS_60.TST**) with 60 clients that are predominately 486-based PCs. Because of the time involved in running the tests, we connect all the clients at the beginning of the test and don't use the Pause option. This way the tests can run unattended.

If your goal is to get an overall measure of how well your system performs, use the system test suite. It contains test mixes that run a variety of transactions. While the transactions are the same for each mix, some of the mix parameters vary. Each mix includes four more clients than the previous mix did. In addition, we vary the value for the Length of the mixes. In testing ServerBench we found that, with a lower client count, we don't need the mix to run as long as it does with a higher client count in order to get repeatable results.

NOTE: You'll find the system test suites and ServerBench's other standard test suites in the **SUITES** subdirectory on the controller.

Setting up error-handling features

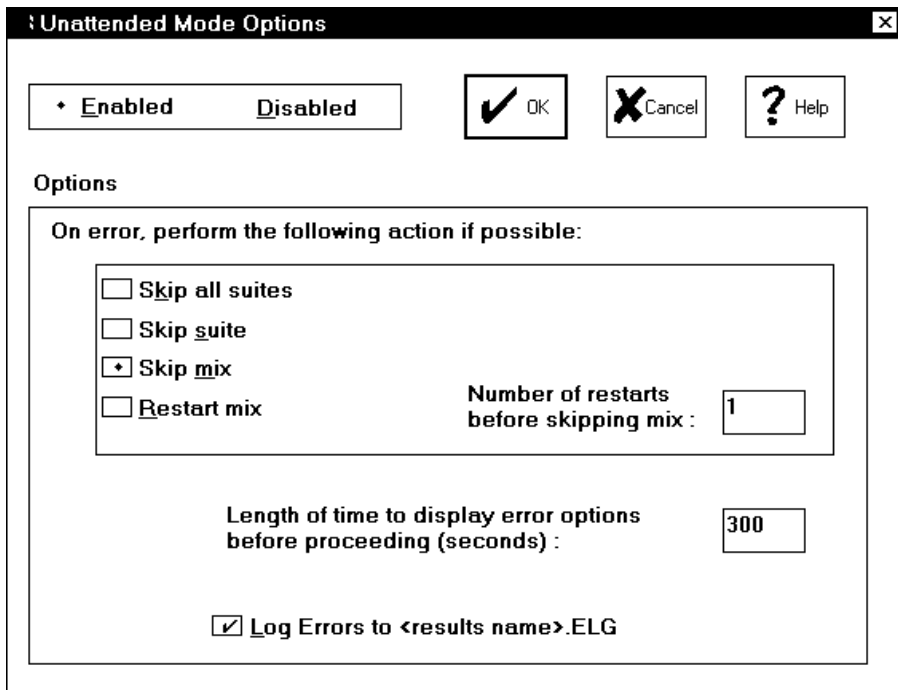
Because ServerBench runs in unattended mode by default, you can specify in advance what you want it to do if it encounters an error. Then, if an error occurs, ServerBench will displays a pop-up error dialog box on the controller and wait for

a specified period of time. Once that time's up, ServerBench will take the action you've specified, such as skipping the mix.

NOTE: If you've checked the option Log Errors, then after ServerBench displays the error message, it displays a pop-up Error Logs dialog box. This dialog box contains the name of the error file (or files, if there's more than one for this test run). You can view the error file (or files) using any text editor. To make sure you don't miss this error information, ServerBench will continue to display this dialog box until you close it.

To set up ServerBench's error handling features, choose the option Unattended Mode from the Advanced menu in the controller window. ServerBench displays the Unattended Mode dialog box.

Figure 8-1: The Unattended Mode dialog box



You can tell ServerBench to:

- Skip all test suites if an error occurs. This will end that test run.
- Skip the current test suite. ServerBench won't run any additional mixes in that suite. Instead, it will begin running the next test suite (if there are any test suites remaining to run).

- Skip the mix that is reporting the error. This is ServerBench's default.
- Restart the mix reporting the error. If you select this option, you need to specify how many times you want ServerBench to try to restart the mix. ServerBench resets the restart counter each time it begins a new mix.
- The amount of time you want it to wait before taking the action you specified.
- Write the error message to an error log (**.ELG**) file. By default, ServerBench writes the message to a file that uses the name *<results-name>.ELG* and saves this file in the controller's **RESULTS** subdirectory. You specify the name of the results file and the directory for it when you select the test suite

NOTE: When you have unattended mode disabled, ServerBench stops if it encounters an error. It only writes error messages to a log file when it's running in unattended mode. To disable unattended mode, choose the Unattended Mode option from the Advanced menu in the controller window. At the Unattended Mode dialog box, choose Disabled. You can re-enable unattended mode by choosing Enabled.



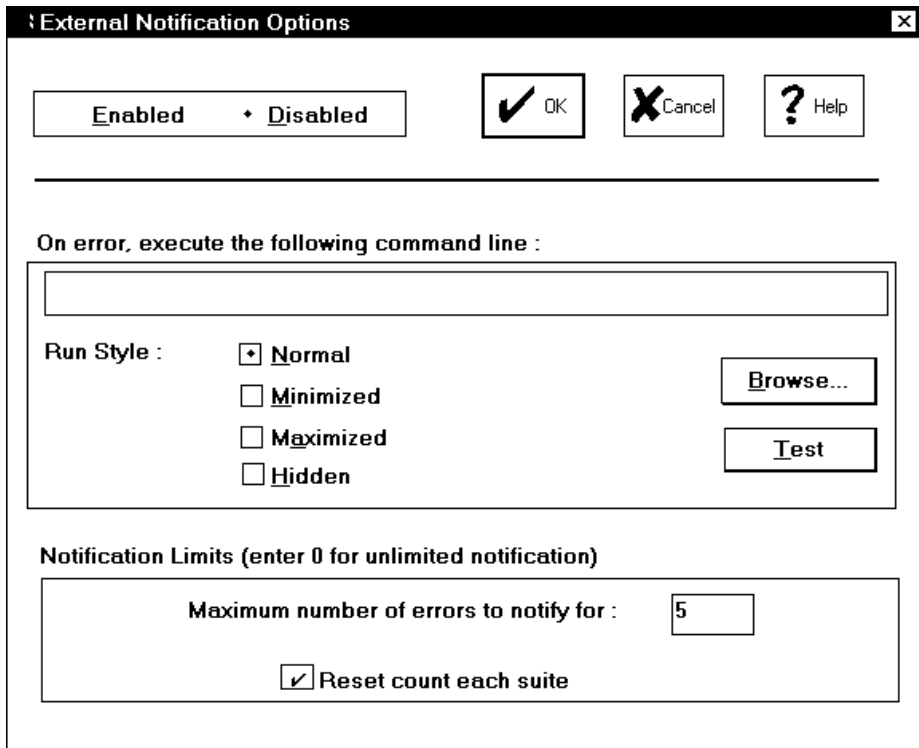
Reminder:

ServerBench saves its results each time it finishes a mix. It doesn't save the results for any mix that has an error. So, even if ServerBench has an error in one mix of a test suite, the results for all the other mixes in the test suite will be fine. For example, suppose ServerBench encounters an error at mix 8 in a test suite that has 16 mixes. It skips that mix but it finishes all the other mixes. You'll have valid results for mixes 1 through 7, and 9 through 16.

Setting ServerBench up to notify you of an error

ServerBench also provides an external notification feature that lets you configure the controller to run a program, such as a pager program, if an error occurs. You specify the options you want in the External Notification dialog box.

Figure 8-2: The External Notification dialog box



Basically, you enter a command line in the External Notification dialog box. Then, if an error occurs, the controller executes that command line. This command line can start any program, such as an executable, a **.bat** file, a **.pif** file, or an **.exe** file program. If you specify a pager program that dials a specified pager number and that program accepts alphanumeric pagers, you can actually send the error message to the pager. To set up ServerBench's external notification program:

1. Choose the Advanced menu in the controller window.
2. Choose the External Notification option.
3. At the External Notification dialog box, choose Enabled.
4. Enter the command line you want ServerBench to execute. If you include **%s** argument in this command line, the controller will replace that argument with the error message. For example, if you're executing a pager program that supports alphanumeric pagers, you can get it to display the error message when it pages you by entering a command line similar to


```
c:\pager.exe tester %s
```

This command line tells the controller to execute the pager program and display the error message.

5. Choose the run style you want the controller to use when it executes the program you specified. You can select any of the standard Windows run styles: Normal, Minimized, Maximized, or Hidden.
6. To see what will happen when the controller executes the command line, choose the Test option. This option lets you try the command line before starting the test suite.
7. Specify the maximum number times you want the controller to execute that program. If you enter 0 (i.e., unlimited), the controller will execute the command line you entered each time an error occurs. However, if you enter a number greater than 0, then, once the error count exceeds the notification count, the controller will no longer execute the command line.
8. Specify whether you want the controller to reset the notification count to the original value each time ServerBench starts a new test suite. For example, if you have the Maximum number of errors to notify for option set to 5 and ServerBench has encountered three errors, the notification count is now 2. This means the controller will stop executing the specified command line after it reports two more errors. However, if you choose the Reset count each suite option, when ServerBench begins a new suite, the controller will reset the notification count to 5.

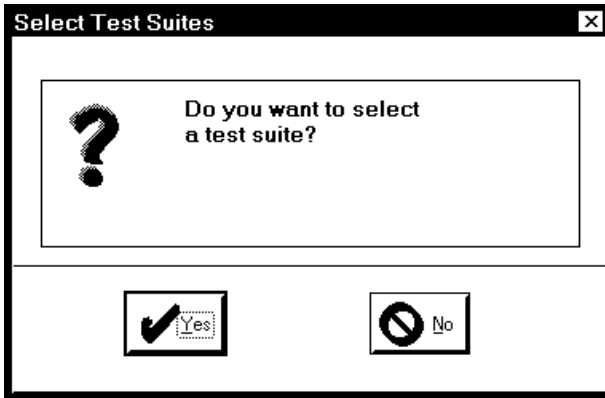
Selecting and running a test suite

To select a test suite, you need to answer Yes in the Select Test Suites pop-up dialog box.

ServerBench displays this box once you've told it you've connected all your clients, each time a test run completes, and when you select the "Add a test suite" option from the drop-down Session menu in the controller window.

NOTE: You can run as many test suites during a ServerBench session as you like. However, if you answer No in the Select Test Suites pop-up dialog box and confirm that you meant no, the controller, server, and clients disconnect. To run another test suite, you would have to restart ServerBench on the server and the clients.

Figure 8-3: The ServerBench message about test suites



ServerBench then displays the Choose Test Suites dialog box.

Adding test suites you want to run

When the Choose Test Suites dialog box appears, go to the directory containing the test suites you want to run and highlight the names of those suites. You can specify one test suite or multiple test suites for each test run. Use the Windows convention of holding down the shift key when you click on a second test suite name in order to choose a range of test suites. Or you can hold down the control (Ctrl) key when you click on test names if you want to select several different test suites instead of a range of test suites.

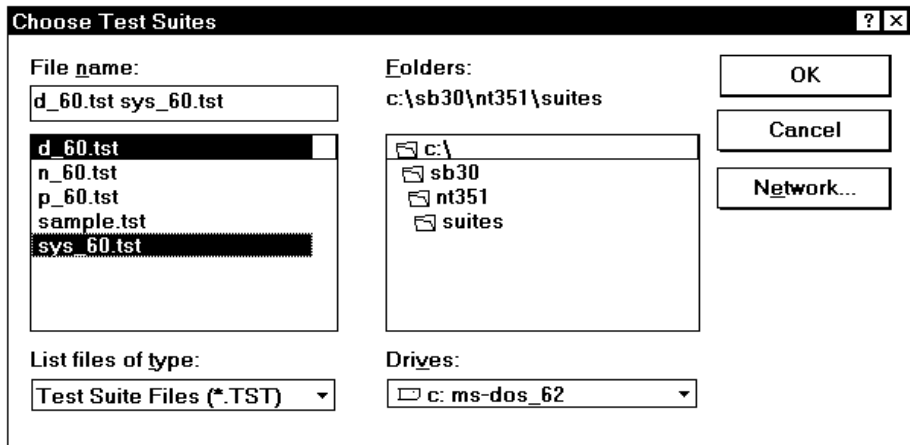
When the Choose Test Suites dialog box first appears, ServerBench shows the **SUITES** subdirectory on the controller. This directory contains ServerBench's standard test suites. ServerBench creates this directory in the controller installation directory when you install the controller files.



Tip:

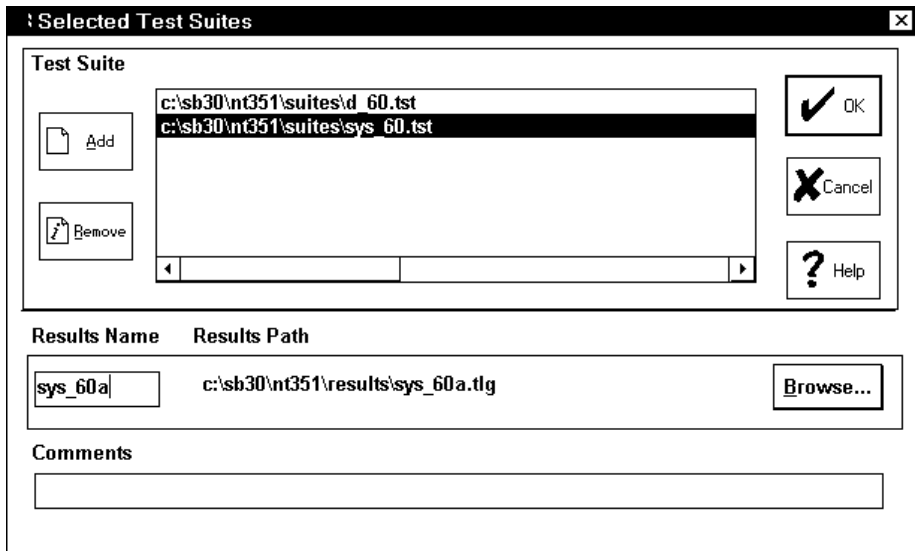
ServerBench uses sticky directories. This means that when you open a dialog box, the box opens to the directory you used most recently.

Figure 8-4: Choosing a test suite



Once you've specified all the test suites you want from that directory, click on OK. ServerBench displays the Selected Test Suites dialog box. From this dialog box, you can add more test suites or remove some of the test suites you currently have selected. Even more important, you can specify the name and path name you want your results file to use. If you don't enter this information, ServerBench will use the base name of the test suite as the name for your results file and it will place your results files in the subdirectory **RESULTS** on the controller. For example, the default file name and path name for the test suite **d_60.tst** shown in Figure 8-5 would be **d_60.tlg** in **c:\sb30\nt351\results**.

Figure 8-5: Selected Test Suites dialog box



When you've got all the test suites you want to run and entered the results information and any comments about the test suites, click on OK. ServerBench then begins executing the test suites in the order in which you listed them.

NOTE: Remember, it's the OK button, not the Start button, that tells ServerBench to execute the test suites listed. The only time you choose Start is when you're ready to begin ServerBench programs on the server and the clients.

Removing a test suite from the list of test suite files

If you decide you do not want to run a test suite that appears in the list of tests, you can delete it from the list in the Selected Test Suites window. To do this:

1. Click on the name of that test suite so that it is highlighted.
2. Choose the Remove button.

Using the pause option

ServerBench provides a Pause option that you can use if you want the benchmark to stop between mixes or test suites so that you can add more clients or perform other ServerBench tasks.

The advantage to connecting the clients as you need them for a mix is that you don't use as many system resources. If a client is connected, even if you don't use it in the mix, it still requires minimal system resources, such as memory and network services.

The disadvantage to adding clients as you need them is that someone must stay on site and monitor the ServerBench tests. If you specify a pause option, ServerBench will wait until someone chooses OK before it runs the next test. This why at ZD Labs we start ServerBench on all the clients at the beginning of the test run and execute ServerBench in unattended mode.

NOTE: If you try to set a pause option while unattended mode is enabled, ServerBench will display a warning dialog box.

You set the Pause option from the drop-down Pause menu. You can set it before you select any test suites to run or while a mix is running. When you set the Pause option, it stays in effect for the entire test session. Or you can use this menu at any point during the test run to set or change pause options. If you set a pause option while unattended mode is enabled, ServerBench will display a warning dialog box.

NOTE: You can't set any Pause options while the Selected Test Suites dialog box is open.

The options you specify apply to all the un-run test suites. You can tell ServerBench to:

- **Run without pausing.** This option tells ServerBench to execute all the test suites in this test run without stopping. If you select this option, you need to connect all the clients your test suites require before you tell ServerBench to begin the test run. For example, each one of the eight mixes in the standard test suite **SYS_60.TST** uses more clients than the previous mix. The first mix uses one client; the second, four; and so on until the last mix, which uses 60 clients. Thus, if you choose the Run without pausing option, you should connect 60 clients before you start the test run
- **Pause between mixes.** This option tells ServerBench to stop after it completes each mix and give you a chance to connect more clients. ServerBench displays the Connect Clients pop-up box when it reaches the pause point.
- **Pause between test suites.** This option tells ServerBench to stop after it completes each test suite and give you a chance to connect more clients. ServerBench displays the Connect Clients pop-up box when it reaches the pause point.

Figure 8-6: Connect Clients

With the last two options, ServerBench will not continue its test run until you choose OK in the Connect Clients box. While ServerBench is pausing, you can:

- Connect more clients. To continue the example using the standard test suite **SYS_60.TST**, you could choose Pause between mixes and start your test run with one client. Each time ServerBench paused, you could connect the clients you need for the next test mix.
- Reconnect any clients that terminated while a test was running.
- View results for the test mix or test suite that just executed. (Press ALT-TAB to get to main ServerBench window.)
- Modify or create a test suite or test mix. (Press ALT-TAB to get to main ServerBench window.)
- Add more test suites to your current test run. To do this, choose the Add option in the drop-down Session menu. When the Select Test Suites dialog box appears, you can add one or more test suites.

Running the test suites

To start the tests, choose the OK button in the Selected Test Suites dialog box. ServerBench will automatically begin executing the tests.

NOTE: The only time you need to choose START is when you first begin your ServerBench session and need to connect the server and clients.

As ServerBench executes a test run, you can do certain things that affect the test run. These are:

- Abort the test. When you abort a test, ServerBench halts the current test and any tests that follow it. ServerBench does not save any results for that test mix.

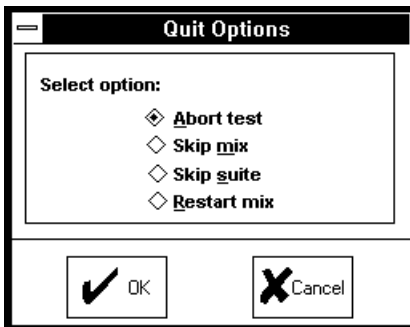
NOTE: If you need to interrupt a test suite, choose Quit and select the Abort Test option. ServerBench will halt the current test and any tests that

follow it without disconnecting any clients. When you interrupt a test, ServerBench does not save any results for the current test mix.

- Skip a mix in the test suite.
- Skip one of the test suites you specified.
- Restart a mix you already executed.

You can perform these tasks from the Quit Options dialog box. This box appears when you choose either the Quit button or the Quit option from the drop-down Session menu.

Figure 8-7: Quit Options



You can also do other things during a test run without affecting your server's scores. These include:

- Switching the colors in the Legend box.
- Changing your pause options.
- Creating a test suite or modifying an existing test suite (as long as it's not running at the time).
- Looking at results (again, as long as they are not for the test mix that is currently running).

After the test suites you specified complete, ServerBench asks you if you want to run other test suite files. If you don't want to run more test suites, answer NO here and then when the confirmation dialog box appears, answer Yes. The clients will disconnect and the screen will return to normal (i.e., it will look like it did when you chose Start Test from the main ServerBench window).

Monitoring a test

While ServerBench is running a test suite, it displays information both in the controller window and the client windows.

From the controller window, you can monitor the progress of the entire test. In the bottom right corner, ServerBench displays information on how much of the test suite is complete and how much time has elapsed during the current mix, the current test suite, and the current test session. In addition, each time all the clients enter a new stage of the test, ServerBench changes the color of their squares in the client grid.

The different client stages

From the Client Screen on each client, you can tell at a glance which stage the client is in by the large letter it displays. The client stages are:

- C** Connected. The client is connected to the server and is ready to execute a mix.
- B** Blocked. ServerBench blocks a client between states to prevent one client from advancing to the next stage before the other clients are ready. For example, if Client 1 finishes running the current mix, ServerBench will block Client 1 from sending its results logs until all clients have completed the test mix.
- I** Initializing. Each client must wait while initialization for the current test mix takes place on the server. Initialization includes creating disk test data files, processor test files, and allocating buffers.
- R** Running. The client is running the current mix.
- S** Sending. The client is sending its log files to server.
- D** Done. The client has finished sending its files to server.
- X** Excluded. The client is excluded from the mix; i.e., this client is not running any of the mix's tests.
- E** Error. An error has occurred.

You can also check the progress of the current mix on the client.

NOTE: You will probably notice a difference between the stage indicated in each client's window and the client grid in the controller window. This is because ServerBench waits and updates all the colors in the client grid at one time. So Client 1 may have finished the test and be waiting to send its results logs, but the color in the client grid will be the Executing stage color. By changing all the colors on the controller at once, we reduce overhead for the server. If you want to know a specific client's status in real-time, click on the client's ID number in the client grid. The pop-up box that appears displays the client's current status. In fact, if the client's

status changes while this box is open, you'll see it change in this box as well.

When the test suites you selected end, ServerBench asks you if you want to run another test suite. If you answer Yes, ServerBench again displays the Select Test Suites window.

Checking the status of all the selected test suites

If you want to check the status of all the tests you've selected during this test run, you choose the View Suite History option from the drop-down Sessions menu in the Controller Window. ServerBench displays the Test Suite History window.

NOTE: ServerBench also displays the Test Suite History window when you double-click on the name of a test suite displayed in the Test Suites box on the controller window.

You can use this window to review which test suites ServerBench has run and is currently running. ServerBench also tells you whether a test suite that has already run, skipped any mixes, or reported any errors.

In addition, you'll find the path name to each suite's results file.

If you request the status of a running test suite, ServerBench displays the status in real time; i.e., the current status of that test suite at that moment.

Running additional test suites

You can run as many test suites as you like during a ServerBench session. Once ServerBench finishes its current test suites, it displays the Select Test Suites pop-up dialog box and asks if you want to run another test suite. Answer Yes.

ServerBench then displays the Choose Test Suites dialog box. To select the test suites you want to run, repeat the steps in the section "Adding test suites you want to run," which appeared earlier in this chapter.

You can also add test suites using the Add a test suite option in the drop-down Session menu. Choosing this option causes ServerBench to display the Choose Test Suites dialog box.

When all the test suites you want to run are listed in Selected Test Suites dialog box, choose OK. ServerBench will start running the test suites.

NOTE: If you answer No in the Select Test Suites pop-up dialog box, ServerBench will ask you to confirm this decision. If you confirm it, the clients will disconnect from the server and the server will disconnect from the

controller. Thus, the next time you want to run a test suite, you will need to restart ServerBench on the server and then connect all the clients.

If you want to look at your current test results or modify a mix before running the next test suite, press the key sequence ALT-TAB. This key sequence switches you between the controller window and the main ServerBench window without causing the server, controller, and clients to disconnect.

Replacing a client that terminates

If you accidentally reboot a client during a “Connect Clients” stage and cause the ServerBench client program to terminate, you can continue without that client, replace that client with another client that has the same ID number, or reconnect that client. The controller will accurately reflect the number of connected clients when the test continues.

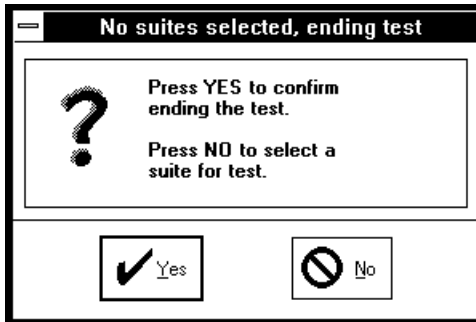
NOTE: The controller only updates the client information when a test is running. This means that, if you’ve just connected five clients, but you accidentally rebooted four of them, the controller will say you have five clients connected. However, once you press OK to indicate you’ve finished connecting clients, the controller will correct its display to show that you only have one client connected.

In addition, ServerBench gives you opportunities to reconnect clients that terminate. For example, if a client terminates when you’re running ServerBench with unattended mode disabled and the Pause option set to “Run without Pausing,” the controller will automatically give you one chance to re-connect the client when the mix ends. If you don’t reconnect the client, the test will proceed normally until it finishes.

Disconnecting the server and clients

ServerBench automatically disconnects the server from the controller when you:

- Answer No at the Select Test Suites pop-up box. When you answer NO, ServerBench displays the following pop-up box asking you to confirm that you want to end this test session. If you answer Yes here, the clients disconnect from the server and the server disconnects from the controller.

Figure 8-8: End of test pop-up box

- Close the controller window, for example by choosing the Close option from the System menu.

When the server disconnects from the controller, the clients automatically disconnect from the server.

Once the server, controller, and clients disconnect, you won't be able to run another test suite until you restart the ServerBench programs on the server and the clients.

End of chapter

Chapter 9

Getting the Best Scores Possible

This chapter contains some tips on how you can get the best scores your server is capable of. It contains information on:

- How we run ServerBench.
- How we recommend you run ServerBench.
- General tips for running ServerBench
- How test parameters can affect your results.
- Other factors that affect ServerBench.

Here's how we test ServerBench

At ZD Labs, we set up a dedicated network for testing ServerBench. We don't do anything else on that network. Only the absolutely necessary applications are running.

For the most part, we only run the ServerBench standard test suites. Because nobody likes playing nursemaid to ServerBench in the middle of the night, we connect the maximum number of clients (60) that the mixes will need before we start the test run.

The next section describes the testing procedures we recommend you use when you run ServerBench.

Here's what we recommend you do

In a best-case scenario, here's how we recommend you run ServerBench:

1. Your server operating system, any drivers it needs, and only the necessary software. If your server has tunable parameters, change them as recommended in the ServerBench installation manual for your server operating system.
2. Your controller PC's operating system (Windows 95 or MS-DOS® and Windows for Workgroups), any drivers it needs, and only the necessary software, such as Excel.
3. On each client, the client PC's operating system (Windows 95 or MS-DOS and Windows for Workgroups), any drivers it needs, and only the necessary software. In addition, streamline each client's **AUTOEXEC.BAT** and **CONFIG.SYS** files as much as possible.
4. Change your servers tunable parameters as suggested in the on-line ServerBench installation manual for your server platform. (If your server platform does not include tunable parameters, then ignore this step.)

NOTE: If you are using ServerBench to compare two servers running the same operating system, make sure that all the tunable parameters on the two servers are the same. Otherwise, you will not be able to get a meaningful comparison of results.

5. Now install ServerBench on the controller, the server, and each client.
6. Reboot your system.

While re-installing everything each time you want to run ServerBench is not likely to be a realistic testing environment for you, you should at least try to reboot your machines between ServerBench sessions. This is because you normally see the best results when you run ServerBench on a clean server; i.e., a freshly rebooted system that has a defragmented disk and a minimum of other software. In addition, when you reboot your server, controller, and clients before running ServerBench, you are always starting the test run from a known state.

If your server fails when running the ServerBench tests, you should exit ServerBench, reboot your server, and restart ServerBench.

Some general ServerBench testing tips

Here are some additional tips we have for running ServerBench. The next few sections describe different factors that can affect your server's scores. Some of the following sections repeat what we've said we do when we test ServerBench; however, they explain why that procedure is important. Other sections give you information on how to get the most representative scores for your server. These are not necessarily the best scores your server will produce, but they are scores that more accurately reflect how your server normally performs.

Keep the background activity down

Don't have any unnecessary applications running during your ServerBench tests. You want the server to be as clean as possible.

You also want the clients and the controller to be as clean as possible. Make sure you don't have any unnecessary TSRs or screen savers running on the client. In addition, if you're running Windows for Workgroups on your clients, streamline each client's **AUTOEXEC.BAT** and **CONFIG.SYS** files as much as possible.

And, naturally, make sure you don't have anything running on the server or clients that could interfere with network traffic.

Basically, you want to have as little running on the server and the clients as you can.

Use a test network, not a production network

We strongly recommend that you run ServerBench on a isolated network test network, not a production network. If people are working on the same network you are using to test ServerBench, your server's performance will suffer even if you have a second server to handle the work. In addition, you're likely to have some annoyed users.

Should you run ServerBench on your production network, it will not damage any files. It will, however, slow down the network because of the network traffic ServerBench generates. Besides, ServerBench can take up a lot of disk space.

Other ways to improve ServerBench results

You may see an improvement in your server's results if you:

- Reboot at least your server, and, if possible, your controller and clients, between each ServerBench test session. Doing this flushes the server disk

cache, which enables you to start ServerBench with an empty file cache and a minimum of other software. If you reboot your server, controller, and clients before running ServerBench, you are always starting the test run from a known state.

- Increase the disk cache size on the server. This primarily affects the disk tests. The size of the cache is very important. Larger sizes generally result in better scores. This is because accessing disk blocks that are outside of the cache requires disk I/O operations that are relatively slow. However, first you should figure out how much memory ServerBench and your operating system need. If you don't keep in mind the other aspects of your server when you increase the disk cache, you may give too much memory to the disk cache and cause your server to thrash. For example, the Processor test does not use any I/O. So if it can't keep information in memory, it has to fetch pages off the disk, which will lower its scores. A general formula for increasing the cache is to figure out how much space the operating system needs. Now add about half a megabyte for each client. If you have a self-tuning system, it should automatically adjust your disk cache size.
- Add intelligent disk and network controllers. These offload the CPU and increase performance.
- Add more disks to the server. This may help.
- Add more memory.
- Add processors where possible. Keep in mind, though, that because of poor processor scaling and other bottlenecks, too many processors may hurt your server's performance.
- Segment the network to reduce the number of clients on a single segment. You want to try to balance the client load on each segment.

If you have the time to stay on site and monitor the ServerBench tests, you might see some advantage in starting ServerBench on only as many clients as the current mix requires. Then, when the next mix requires more clients, start ServerBench on the additional clients. The reason for testing this way is that there is a small amount of overhead associated with each client that is running ServerBench. As a result, connecting 50 clients and then using only five to run tests is not the same as simply connecting five clients. Remember, you can always add more clients as you run tests.

NOTE: The disadvantage to this approach is that someone must always be on site to add the additional tests. At ZD Labs, we start ServerBench on all the clients the mixes will need at the beginning of the ServerBench session. We do not add clients as the tests run.

How hardware and software affect results

Both hardware and software factors play a role in the scores you get when you run ServerBench.

When you try to improve your server's scores on ServerBench, you quickly realize that everything affects your scores. For example, your server may have a very fast disk subsystem. If your network system is slow, though, your disk subsystem will also appear slower. This is because the transaction response time includes the amount of time the request travels along the network and the amount of time the disk subsystem spends responding to the request.

A look at what affects each server subsystem

The following lists breaks down the hardware and software factors into different groups. You'll notice some overlap in the components of the different groups. Where one factor plays a large role in more than one subsystem's results, we include that factor with each subsystem. For example, you'll see network resources listed more than once. Quite simply, network resources play a part in any ServerBench score you get.

Processor subsystem	CPU model and clock speed Number of processors in the server Internal and external CPU RAM caches Amount of memory Main memory speed Memory/CPU interface speed and width Wait states System configuration Network resources
Disk subsystem	Memory (the more RAM the better) Hard disk controllers and size of hardware cache (if any) Hard disk drive types Number of hard disks Hard disk driver software Software disk cache and size (if any) Fragmentation level of disk and remaining space on disk Compression Disk organization (for example, striped, RAID5, RAID0) Network resources

Software	The server's operating system
Network	Whether the server uses Ethernet, token ring, etc. Number of network segments Number of clients on each segment The network protocol Network Interface Card (NIC) type and driver

Factors related to your system's hardware and software

As you look at the factors that affect the specific server subsystems, keep in mind that the overhead of managing resources also affects your scores. You know, for example, that adding processors usually improves your server's scores. However, if you add too many, the overhead of managing them can cause your scores to go down due to poor CPU scaling.

You should also look for tips on what affects your server in the on-line installation manual that came with your server platform. For example, if you have a UNIX server, you should set your kernel parameters based on the recommendations in the on-line installation manual for that ServerBench platform. In addition, depending on the ServerBench UNIX platform, you have to be sure *not* to configure your server as a relational database server. Configuring your server as a relational database takes up almost all of your server's file cache and devotes it to shared memory. Since ServerBench uses file services instead of doing raw I/O, this makes your server look terrible. You'll also notice if you check the operating system kernel parameters that some of them change depending on the number of clients you are running.

Take a look at your test parameters

If you create your own test suites, you also want to consider the values you supply for some of your test parameters.

For example, if you're setting up a Server to Client test or a Client to Server test, you may want to set your Request Size parameter as close as possible to maximum packet size your network software supports. If you set this parameter to a small size, then the overhead involved in transferring the data can bring down your scores. If you set this parameter to a value that is greater than the maximum packet size your network implements, then your network software breaks the data into smaller packets.

When you are running a disk test, consider the Disk test file initial size parameter. By multiplying the value in this parameter by the number of clients running the

test, you'll be able to determine the total disk space used for the data files created by the test.

Adjusting the I/O range parameter allows you to control the range within the test data files that the disk test will access. Lowering this parameter results in a smaller portion of the file being used for disk I/O. This generally reduces the amount of file cache occupied by the file.

Increasing the I/O range parameter has the opposite effect. More of the file is accessed and more of the file cache is required. Multiplying the number of clients by the I/O range parameter gives a rough indication of the amount of file cache necessary to fully cache all of the disk accesses. The smaller the actual size of the file cache is compared to this value, the more disk accesses that will occur and the greater the stress on the server.

Suppose your server cache is 40 MB. If you set the I/O range to 2 MB and you had 16 clients, your total disk space used would be 32 MB, which would fit inside cache. However, if you ran the mix again with 24 clients, your test area would be 48 MB and would break out of the server cache. If everything else is the same, your results for the first mix would be better than your results for the second mix.

NOTE: Running ServerBench so that your I/O test area fits in cache, while it may produce better scores, is not necessarily producing the most realistic results. For example, always running the disk tests with a test size that fits in the server cache is not how users actually use a server. In the real world, requests frequently miss the cache.

The values you supply for Ramp up, Ramp down, and Think time also can affect your results.

Record your results while the server in a steady state

One of our goals in testing ServerBench is to get the server to a steady state before ServerBench actually starts recording test results.

By steady state, we mean that point in the test where the server's subsystems are working with a steady workload. The server has stabilized after starting the tests, and it has flushed out any residual caching effects that occurred as a result of creating test data files. All the clients are executing the tests, so the server has a steady-state load (unlike the load during the Ramp up or Ramp down periods when not all the clients have started the tests yet or some of the clients have already ended the tests). Instead, the server is executing a load that more closely reflects how it would perform under normal user activity.

In addition, steady state implies that the length of the queue of threads or processes waiting for processor time, the number of blocks of the file cache in use, and the length of the queue of messages to be transmitted on the network have all generally

stabilized and are responding appropriately based on the load the different tests are putting on them. Because different tests stress different areas of the server, a steady state for the server may look one way with one test and another way with another test. For example, during a disk test, the length of the queue of threads or processes waiting for processor time is likely to be much shorter than it is during an actual processor test.

Determining what to test on your system

As you look at the ServerBench tests, consider what would be the most useful information you can get about your system. For example:

- What subsystems do your primary software applications use the most? If you use applications that spend a lot of time accessing the disk, then you may want to spend more time running tests that use the ServerBench disk tests.
- Where are the bottlenecks in your system? To determine where your system encounters bottlenecks, you may want to run three different test suites, where each test suite focuses on a different server subsystem.
- Have you set the test parameters to provide a reasonable test of your system? For example, if you are using just one Ethernet for a test that uses 60 clients and no Think time, you won't get a valid score for your server. Instead, you'll be measuring the speed of a heavily congested Ethernet.
- Are there things you could do to optimize your server's performance? (See the on-line ServerBench manual for your server operating system.)
- What effect does changing a parameter have your results? You can experiment with changing system parameters and comparing the ServerBench results before and after the change. If you want to see the effects different parameters have, make sure you only change one parameter at a time between ServerBench test runs. This way you can gauge the effect that parameter has on performance. The server parameters you may want to change are:

Size of the disk cache.

Scheduling behavior.

Cache replacement policies.

Number of CPUs.

Number of disks.

- How many users do you have or are you expecting to have? If you are planning to add users, you may want to run that many clients on ServerBench to see how well your server performs under those conditions.

NOTE: Remember, though, that because ServerBench uses stress tests, one client is equivalent to more than one user. If you want to try to make each client equivalent to a single user, you should modify your test mixes so that they more accurately reflect how people actually use a server. For example, most test mixes have 0 for the Think time parameter. This affects how long a client waits before issuing a new request of the server. Most users actually work with the data they get from the server before they issue another request. By increasing the value of the Think time parameter, you can decrease the amount of stress a client places on the server.

One of the key ways you can use ServerBench is to see what happens to performance when you switch one piece of hardware while maintaining an otherwise constant operating system and test bed.

Another way you can use ServerBench is to compare the performance of different servers. By running the same test suites with the same settings on different servers, you can determine which server best meets your needs.

NOTE: Remember, though, that to get a meaningful comparison, you need to run ServerBench the same way on both systems. This means that your test parameters need to be the same, your testbed needs to be the same, and you need to run the same test suites in the same order. It also means that, if you are comparing servers that use the same operating system, you need use the same system setup and tunable parameters on the servers.

The best score isn't always the right score

As you can see from our recommended procedures and general tips, you can do a lot of things to get better scores for your server. However, the best scores aren't necessarily the most appropriate scores.

When you test your server, keep in mind that the most useful score for you is one that reflects how your server performs in a real world environment. For example, if your users typically work with large files, then running disk tests using small test data files that fit in the server cache won't tell you what kind of performance your users can expect from that server.

In addition, we see a substantial improvement in scores just by going from a one-processor system to a two-processor system. But if your server only has one

processor, then that's how you need to test your system. (Of course, you might want to use the two-processor scores to build a case for getting additional processing power.)

In general, try to test your system in a way that reflects how you use your system.

Creating test mixes to stress your server

Now that you've looked at all the ways to improve your results, here're some tips for making your tests more stressful. By increasing the stress of your tests to the point where TPS drops off, you'll get a prediction of how far you can push your server and still get reasonable performance. This information can be very useful if you're trying to decide whether to upgrade your system because your workforce has grown.

So, if you want to create test suites that really stress your server, you'll need to:

- Increase the number of times within a transaction that ServerBench executes the processor test.
- Increase the initial size of the data file that the disk tests use.
- Increase the total size of the disk tests.
- Reduce the Think time parameter.
- Use a large value for the I/O range parameter.

You can also increase the total size of the network tests; however, that will not affect your throughput as much as these other actions.

In addition, adding more clients always increases the stress on your server.

End of chapter

Chapter 10

Creating Your Own Test Suites

Each mix you run is actually part of a test suite file. A test suite file is simply a binary DOS file with a **.TST** extension that resides on the controller. It contains one or more mixes that tell ServerBench which transactions to run. You can create your own test suite files and tailor them to your system. You can also modify an existing test suite file, such as one of the ServerBench standard test suites.

This chapter takes you through the steps of creating and modifying test suite files. It explains the different parameters you can set when you create the test mixes and how they affect your server's performance. In this chapter you will find information on:

- Creating test suites.
- Using the Mix Definition window.
- Setting parameters for the mix.
- Setting parameters for each test in a transaction.
- Copying mix information from other mixes.

Developing test suite files

To run tests on ServerBench, you select the test suite file that contains the test mixes you want and then execute it. The test suite file is a DOS file that ends in a **.TST** extension. (If you don't give the file a **.TST** extension, ServerBench cannot locate the results files and display them using Excel.) Your test suite file should always reside on your controller. When you install ServerBench, we create a subdirectory called **SUITES** in the controller's installation directory. You'll find the

ServerBench standard test suites in that directory. It's also where we recommend you place any test suite files you create.

Here's how you can approach creating your own test suites:

1. Create a test suite. The test suite file is a container for one or more test mixes. It provides the base name for the results file and a location for the file. When ServerBench displays the results, it orders them according to the test mixes within the test suites.
2. Create the mixes. Each mix contains transactions that ServerBench executes. When you create a mix, you set certain global information that ServerBench uses when it runs the mix. This information includes how long to run the tests, how many clients to include in the mix, and, most importantly, which transactions to execute.
3. Create the transactions. Each mix contains one or more transactions. The transactions provide details on the work you want the server to do. The transactions tell the server which tests to execute. When you create a transaction, you supply two types of information: global information for the transaction and specific test parameters.

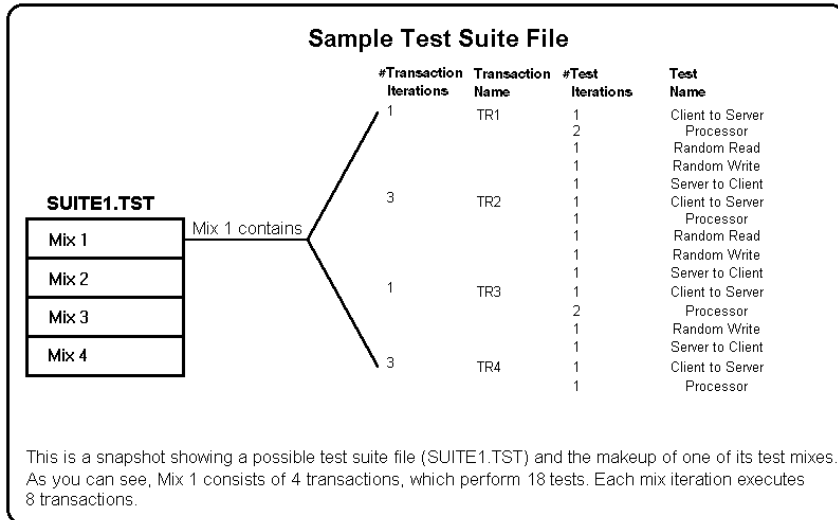
For the global transaction information, you'll enter the transaction name and the number of times a client requests that transaction during one iteration of a mix. If your transaction is using any disk tests, you'll also need to specify the initial size of the test data file and the I/O range parameter. Both these values will be constant for all the disk tests in the mix.

The specific test parameters can vary for each test. These include the number of times the server performs that test when a client requests that transaction and, if it's a disk test or network test, the Request Size and Total size (we explain these terms later in the chapter).

By layering information this way, the clients' request for service are more varied and better reflect a real world client/server situation.

Figure 10-1 shows you an example of how a test suite file might look.

Figure 10-1: Test suite file and the transactions in its first mix



This approach to creating test suites allows you to better target what you want to test on your system. For example, suppose you want to see how your server performs as you add clients. First, you would create a test mix that contains numerous different transactions. Next you would copy this test mix so that each mix in that test suite runs the same set of tests. Here's the catch, though: you'd change the client counts so that each subsequent test mix in the test suite file would run the tests with more clients than the previous test mix did (i.e., Mix 1 would have 1 client, Mix 2 would have 4 clients, Mix 3 would have 8 clients, and so on). Thus, once you run this test suite, you can tell by looking at your results where performance begins degrading as you add clients.

To summarize, here're some notes to keep in mind as you think about test suites:

- The test suite file is a container for the test mixes. There are no parameters you set that are associated with the test suite file.
- Your test suite files must reside on the controller. We recommend that you place them in the **SUITES** directory on the controller. This is where you'll find ServerBench's standard test suites.
- You can set certain parameters that apply only to a test mix. These include the number of clients in the mix, the Length of a mix, the value for Think time, and the maximum Delay a client will wait before starting a mix. ServerBench checks each of these parameters each time you run a mix.

- You can set parameters that apply only to each transaction within the mix. A transaction contains the tests that the clients ask ServerBench to execute. When you set up a transaction, you must give the transaction a name, specify how many times you want ServerBench to execute that transaction during one iteration of the mix, and tell it which tests to execute,
- When you specify a test within a transaction, you must set the parameters that apply to it. For example, if you have a disk test, you need to specify the Request Size, Total Size, and number of test iterations.

We explain all of these parameters later in this chapter.

Designing a test mix

Creating a test mix boils down to supplying a lot of values that tell ServerBench exactly what to do. You can save some time and get a test mix that fits your system better if you think about these issues in advance. As you set up a test mix, you'll need to tell ServerBench:

- How many clients you want to run in the mix and what the path names of their data files on the server will be. (If you specify a directory in the path name, make sure the directory exists before you run the test. If you don't specify a directory, ServerBench will create the disk test data files in the ServerBench directory on your server.)
- Whether you want to use different groups of clients.
- How much time you want ServerBench to allot for the Ramp up and Ramp down periods. ServerBench does not include any mix iterations that begin during these times in the TPS scores. This helps ServerBench avoid skewing the results because the load on the server was light while the first clients were starting up and the last clients were ending.
- How long to wait for Think time. This is the amount of time a client waits after it has received a response from the server before it issues another request to the server. In general, the smaller the Think time, the more each client stresses the server.
- How long the test mix will run at a minimum. The test mix will probably run slightly longer than the value you specify for Length because ServerBench does not stop any clients in the middle of a mix iteration. It always waits for the clients to finish their current iterations.
- Which tests you want to include in the transactions. How often do you want ServerBench to execute each transaction during one iteration of the mix. For the disk tests, how big do you want the initial test data files to be. How much data do you want ServerBench to pass to the operating system in a single

chunk (Request Size). In addition, you need to decide how much total data you want ServerBench to move (Total Size). ServerBench will continue to move chunks of data that are the Request Size you specify until it has moved the amount of data you specify in Total Size.

The following sections explain the steps involved in creating and editing test suite files.

Creating a test suite: the basic steps

Here are the basic steps involved in creating a test suite. While these steps don't explain the different mix and test parameters or what each step means, they do give you a feel for what the process of creating a test suite involves. Except for the first two steps, which are obvious, each of these steps contains a pointer to a section that explains in detail what that step means.

NOTE: The difference between modifying or creating a test suite is that when you modify a test suite you work with an existing test suite. When you create a test suite you enter the name of a file that does not yet exist.

To create (or modify) a test suite file:

1. Go to the main ServerBench window.
2. Choose the Create or Edit Test Suites button. The Create or Edit Test Suite File dialog box appears.
3. Enter the name of the test suite file you wish to create. If the name you enter does not exist in that directory, ServerBench asks if you want it to create the file. (If you want to modify a test suite file, highlight or enter the name of the existing file. ServerBench displays information on that test suite.) *See the section "Specifying a file name for your test suite."*
4. At the Mixes in Test Suite window, choose New. *See the section "Specifying a file name for your test suite."*
5. At the Mix Definition window:
 - a. Enter your mix name. (This step is optional, but we recommend you supply a mix name.) *See the section "Specifying the mix name."*
 - b. Enter the values for the Ramp up, Ramp down, and Length parameters in the section called Duration Information. (ServerBench's standard test suites vary these values.) *See the section "Entering the duration information for a mix."*

- c. Enter values for the Delay and Think time parameters in the section called Timing Information. The default is 0, which is what ServerBench's standard test suites use. *See the section "Specifying the timing information."*
- d. Define the types of transactions you want the server to perform in this mix. *See the section "Defining the Mix Test Definitions."*

When you create a transaction, you will also set the parameters for that transaction such as how many times you want the transaction to be requested during one mix iteration and what the name of the transaction is. For the different tests within the transaction, you'll need to specify:

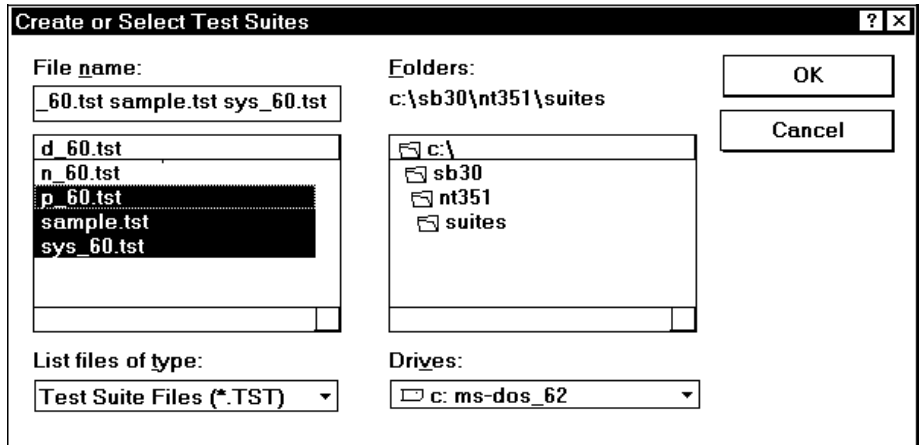
- 1a. The test type (Processor, Sequential Read, Sequential Write, Random Read, Random Write, Append, Server to Client, or Client to Server).
- 2a. If you're setting up a disk test or a network test, you'll need to specify a Request size and a Total size.
- 3a. If you're setting up a disk test, enter a value for Disk test file initial size and the I/O Range parameter. (ServerBench's standard test suites use a value of 10 MB for the I/O range parameter. If you are using a test suite you created for an earlier version of ServerBench, then ServerBench sets this value to the same size as your data file.)
- e. Enter the number of clients that you want to have run the mix and the path names of their data files. *See the section "Providing the basic client information."*
- f. Specify which groups of clients you want ServerBench to run in these tests. *See the section "Including client groups."*

Specifying a file name for your test suite

When you are working with test suites, ServerBench always asks you for the name of the test suite. If you are creating a new test suite, you need to enter the name you want that file to have. If you are modifying a test suite, you need to enter the name of the existing test suite file.

1. Go to the main ServerBench window.
2. Choose the Create or Edit Test Suites button. The Create or Edit Test Suite File dialog box appears.

Figure 10-2: Create or Edit Test Suite File dialog box

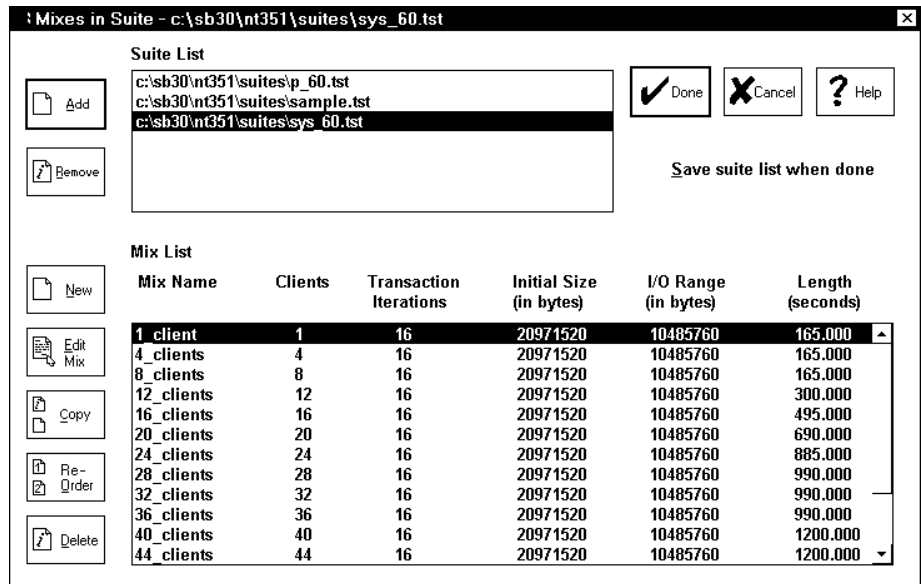


3. Enter the name in the File Name box or highlight the test suites you want. To create a test suite, type in the name you want the file to have. Remember, you must give your test suite file name a **.TST** extension.

If you want to modify one or more existing test suites, you can use the Directories and Drives sections of this dialog box to locate the directory and exact file name for that test suite and then click on that name.

4. Select OK. If you entered a file name that doesn't exist, ServerBench will ask you if you want to create a new file.
5. The Mixes in Suite dialog box appears on your screen. If you are creating a new mix, the Mix List will be empty. However, if you're modifying existing mixes, the Mix List shows all the test mixes included in the test suite file you opened. It also lists the number of clients the mix uses (Clients), the number of transactions executed during one iteration of the mix (Transaction iterations), the initial size of the data file the test will use (Initial size), the size of the I/O range parameter (I/O Range), and how long the mix is supposed to run (Length).

Figure 10-3: Mixes in Suite window



You can perform several tasks from this window. You can:

- Define a new mix for your test suite by choosing the New button. See the section "Creating or editing a mix" for more information.
- Edit an existing mix in your test suite by double-clicking on the mix you want to edit or by choosing the Edit Mix button. See the section "Creating or editing a mix" for more information.
- Copy a mix or mixes from another test suite by choosing the Copy button. See the section "Copy mixes into a test suite" for more information.
- Delete a mix from the test suite by highlighting the mix and then choosing the Delete selected mix button. See the section "Deleting mixes from a test suite" for more information.

Creating or editing a mix

When you are at the Mixes in Test Suite window and you tell ServerBench you want to create a new mix or edit an existing mix, ServerBench displays the Mix Definition window. This is where you set the parameters for the mix and set up the test transactions you want the mix to execute.

Figure 10-4: The Mix Definition window

Mix Definition - c:\sb30\nt351\suites\sys_60.tst

Mix Avanced Help

Mix Name **Duration Information (sec.)** **Timing Information (sec.)**

Mix 2 of 16 Ramp Up: 50.000 Delay: 0.000

4 clients Ramp Down: 15.000 Think: 0.000

Length: 165.000

OK Cancel Help

<< Previous Mix Next Mix >>

Transaction Definitions

Transaction Iterations	Transaction Name	Test Iterations	Test Type	Request Size (bytes)	Total Size
1	TR1	1	CS	500	500
		15	P	N/A	50
		2	RR	2048	2048
		15	P	N/A	50
3	TR2	2	RW	2048	2048
		1	SC	2048	2048
		1	CS	500	500
		7	P	N/A	50
		2	RR	2048	2048
		7	P	N/A	50
		2	RW	2048	2048
		1	SC	2048	2048

New Delete

Disk test file initial size (MB): 20.000

Disk test file I/O Range (MB): 10.000

Client Information

Total Number of Clients: 4

Client# Data file pathnames:

1 data1

2 data2

3 data3

4 data4

Enter in a name for the mix up to 15 characters in length.

In the Mix Definition window, you:

- Specify the Mix name.
- Enter values for the Ramp Up, Ramp Down, and Length parameters in the section called Duration Information.
- Enter values for the Delay and Think time parameters in the section called Timing Information.
- Define the transactions you want the server to perform in this mix.
- Enter client information, including the path names of the data files ServerBench creates on the server during the mix initialization.
- Specify which groups of clients you want ServerBench to run in these tests. (Optional)

If you want to enter information on several different mixes at once, you can enter the information in one mix and then choose the Copy mix fields across this suite option in the Advanced menu to tell ServerBench to include that information in all the mixes in this test suite.

You can also use the Previous and Next buttons to scroll through and edit all the mixes in a test suite.

While working at the Mix Definition window, you can insert a new mix into the test suite. This new mix will follow the mix that you currently have open. Use the Insert a mix option in the drop-down Mix menu. This menu also has an option that lets you delete the current mix from the test suite.

You can include as many mixes in a test suite as you like. Keep in mind, though, that the more mixes you include, the slower it'll be to work with and manipulate those mixes. We recommend that you include no more than 100 mixes.

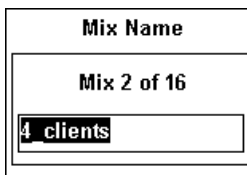
NOTE: If you try to leave the Mix Definition screen without entering the necessary mix information, ServerBench issues an error message telling you what information it needs. The only way you can exit the mix is to supply the correct information or press cancel and lose all the information you've entered. In addition, ServerBench always validates mix information before it executes a mix. (ServerBench does not include the time it spends validating this information in the test time.)

The next sections describe the different portions of the Mix Definition window and tell you what information you need to enter.

Specifying the Mix Name

You enter the Mix Name in the box under the title Mix Name. This name can be any alphanumeric entry up to 15 characters long. Just above the mix name box, ServerBench tells you which mix this is in the test suite, such as Mix 2 of 16.

Figure 10-5: The Mix Name



What the Mix Name does

This field is optional; however, we recommend that you always supply a descriptive Mix Name. ServerBench uses the Mix Name to identify the mix in the results tables, in the controller window as you run the test suites, and in the Mixes in Test Suite window. The field has no effect on your test results.

Entering the duration information for a mix

The parameters in this section of the Mix Definition window are:

- **Ramp up.** This is a fixed number of seconds at the beginning of the mix. ServerBench does not record any results from mix iterations that begin during the Ramp up period. This way your test results aren't skewed because the server load for the first few clients was very light, thus resulting in an artificially high score.

Try to select a value that gives the server enough time to have reached a steady state. You can verify this by looking at the valid iterations column in Table 3 of the results. If all the valid iterations count for the clients in the mix are consistent, then the Ramp up period is long enough. (The value of Ramp up varies from mix to mix in the ServerBench standard test suites.)

- **Ramp down.** This is a fixed number of seconds at the end of the mix. ServerBench does not record any results from mix iterations that begin during the Ramp down period. This way your test results aren't skewed because the server load for the last few clients was very light, thus resulting in an artificially high score.

Try to pick a time that is long enough to allow all of the clients to switch from the Running stage to the Sending Results stage of the test. (The value of Ramp down varies from mix to mix in the ServerBench standard test suites.)

- **Length.** This is the minimum number of seconds that the mix executes. (Because ServerBench always finishes the current iteration of a mix, a mix might run longer than the amount of time specified by the Length parameter.) The value for this parameter includes time for both the Ramp up and Ramp down periods.

(The value of Length varies from mix to mix in the ServerBench standard test suites.)

For more information on how you determine a value for these parameters, see the next three sections.

These three parameters apply to all the clients in a mix.

All timing on ServerBench is done in seconds.

Figure 10-6: Duration Information

Duration Information (sec.)	
Ramp Up:	50.000
Ramp Down:	15.000
Length:	165.000

What the Ramp up, Ramp down, and Length parameters do

Essentially, ServerBench executes a test mix in three parts:

Ramp up	Recording test information	Ramp down
---------	----------------------------	-----------

The clients actually start sending requests to the server during Ramp up period. They continue sending requests until after the end of the Ramp down period (i.e., until they finish their last iteration of the test). However, a client only records TPS information for mix iterations that begin during the Recording test information period. If a mix iteration begins during a Ramp up or Ramp down period, the client does not record any information about the transactions it requests. If the mix iteration begins during the Recording test information period, the client counts how many times it requested each transaction and how long it took to get a response from the server.

NOTE: An iteration is one complete pass through the mix. For example, each time a client has requested all of the transactions in a mix, we say that it has completed one iteration of the mix.

By executing a test mix this way, ServerBench only measures your server when it is in a steady state. We like to measure your server's performance when it is at a steady state because that's when the server's subsystems are executing an appropriate workload for the test ServerBench is running. The server is no longer thrashing as a result of starting the tests and it has flushed out any residual caching effects that occurred as a result of creating test data files. In addition, all the clients are executing the tests, so the server does not have a light load. (A light load occurs when only a few clients are running tests, such as during the Ramp up or Ramp down periods when not all the clients have started the tests yet or some of the clients have ended the tests.)

The actual Recording information part of the test is the time that remains once you subtract the Ramp up time and the Ramp down time from the total time (i.e., Length) for the mix. For example, if you specify a test Length of 60 seconds with a Ramp up time of 5 seconds and a Ramp down time of 10 seconds, ServerBench will actually record information for only 45 seconds.

NOTE: Remember that the test may last longer than the amount of time you set for the length because ServerBench will not stop the test until the last mix finishes.

Choosing values for Ramp up, Ramp down, and Length

The unit of measure you define for the Ramp up, Ramp down, and Length parameters is always in seconds.

There's no fixed formula for deciding what values to supply for these parameters. What you're trying to do is set the Ramp up and Ramp down parameters so that ServerBench only records test results when your server is in a steady state.

As a rule of thumb, you might want to make the Ramp up and Ramp down periods last as long as it takes all the clients to perform one iteration of the mix. If your Ramp up value equals one iteration, then your server should be at a steady state when ServerBench starts recording test results because all the clients will be participating in the test. If your Ramp down value equals one iteration, then, if a client begins an iteration just before the beginning of the Ramp down period, all the other clients will still be requesting transactions of the server when that client finishes. In both cases, all the clients should be executing transactions during the period when ServerBench records results.

In general, you'll want the Length parameter to be long enough so that ServerBench can record the results from several iterations of the mix. The more iterations you have, the more likely you are to avoid gating errors. We recommend that you allow for at least 10 iterations of a mix. So calculate how much time you've specified for the Ramp up and Ramp down periods and then add enough time to that for the number of iterations of the mix you want ServerBench to perform.

NOTE: When you calculate a value for the Length parameter, remember to make sure it is greater than the sum of the Ramp up and Ramp down periods. Otherwise, because the clients ignore the results for any iteration of a mix that starts during Ramp up or Ramp down, the clients won't record any results for that mix.

Determining how long an iteration takes

To figure out how long an iteration takes, run a short test and then look at your results. Table 5 shows you the number of Ramp up iterations, Valid iterations (i.e., those iterations for which ServerBench recorded results), and Ramp down iterations. You can use this information to adjust the values you supply for these parameters.

Specifying the timing information

Two other general mix parameters that affect the time a test takes are:

- **Delay.** Enter the maximum amount of time in seconds (or fractions of seconds) that you want a client to wait before starting a test once the controller tells the clients to start. When you set the Delay time parameter, each client takes that value and generates a pseudo-random number. This way the Delay time can vary for each client. As a result, when you set Delay time, you stagger the clients' initial requests to the server instead of overloading the server with a burst of requests at the beginning of each mix. (The ServerBench standard test suite **SYS_60.TST** uses a Delay of 0 seconds.)
- **Think time.** Enter the number of seconds or fractions of a second you want a client to wait once it gets a response from the server before it issues another request to the server. (The ServerBench standard test suite **SYS_60.TST** uses a Think time of 0 seconds.)

Figure 10-7: Timing Information

Timing Information (sec.)	
Delay :	<input type="text" value="0.000"/>
Think :	<input type="text" value="0.000"/>

These two parameters apply to all the clients in a mix.

What the Delay and Think time parameters do

The Delay and Think time parameters help you regulate how much you stress your server.

When you enter a value for the Delay parameter, ServerBench staggers the clients' initial requests to the server. This way the clients don't overload the server with a burst of requests at the beginning of a mix.

Here's how the Delay parameter works: you specify a value in seconds for the Delay time. Each client then generates a pseudo-random number between zero and the Delay value in .1 second increments. When the clients receive a signal to start the next test mix, each client waits its prescribed Delay time and then begins executing the mix. This way the Delay times for the clients vary but do not exceed the amount of time specified by the Delay parameter. For example, if you set Delay to one second, each client will wait between .1 second and one second before sending its initial request to the server. If you set the delay to two seconds, each client will wait between .1 second and two seconds before sending its initial request to the server.

When you set a value for Delay time, make sure that value is shorter than the value for Ramp up.

The Think time is the amount of time you want a client to wait once it has received a response from the server before it sends another request to the server. For example, if you set Think time to 0.5, each client will wait half a second after receiving a response from the server before sending another transaction request to the server.

The shorter this amount of time you enter for Think time, the more the client stresses the server. This is because, in a normal work environment, most users send a request to the server, get a reply, do something with the data they receive, and then issue another request. They don't just take the data and then pelt the server with another request. You can use the Think time parameter to create tests that more closely reflect how users work with a server; however, you won't be stressing the server as much.

Defining transactions

The next part of setting up your mix is defining the transactions you want it to perform. A transaction can consist of a single test or several tests. The reason for using transactions instead of just sending tests to the server is that a transaction lets you bundle the tests together. This reduces the number of network requests, which can cause bottlenecks that aren't related to the server's performance.

When a transaction consists of only one test, we call it a singleton. These are the easiest transactions to create because you just enter the test mnemonic in the Transaction Name box.

Each time you set up a transaction, you specify parameters for the test you select. This means that you can have multiple tests with the same name, such as sequential write, but each test is different because its parameters differ from the parameters other tests are using.

The tests you can include in a transaction are:

- **Processor.** There is only one processor test and its mnemonic is P.
- **Disk.** Sequential read (SR), Sequential write (SW), Random read (RR), Random write (RW), and Append (A).
- **Network.** Server to Client (SC) and Client to Server (CS).

The basic steps for creating a transaction

You define the transactions in the Transaction Definitions section of the Mix Definition window. If you've already created some transactions, you'll see them listed here.

Figure 10-8: Setting up transactions

Transaction Definitions					
Transaction Iterations	Transaction Name	Test Iterations	Test Type	Request Size (bytes)	Total Size
1	TR1	1	CS	500	500
		15	P	N/A	50
		2	RR	2048	2048
		15	P	N/A	50
		2	RW	2048	2048
3	TR2	1	SC	2048	2048
		1	CS	500	500
		7	P	N/A	50
		2	RR	2048	2048
		7	P	N/A	50
		2	RW	2048	2048
		1	SC	2048	2048

Disk test file initial size (MB):
 Disk test file I/O Range (MB):

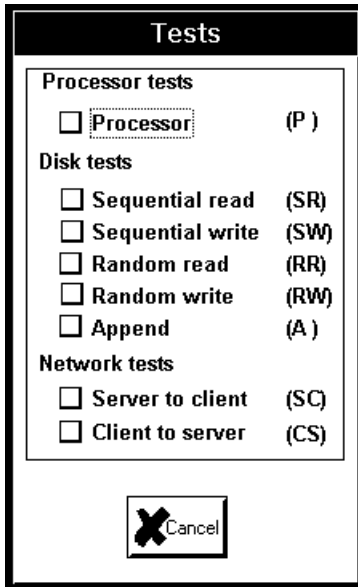
To create a transaction:

1. Choose the New button. ServerBench moves your cursor to the Transactions Iterations box.

NOTE: When you're creating several transactions, you'll notice that ServerBench always places the new transaction after the currently highlighted transaction.
2. Under Transaction iterations, enter the number of times you want each client to request the transaction during one iteration of a mix.
3. Enter the name of the transaction in the box labeled Transaction Name. You can enter any name you like or you can use an existing mnemonic for a ServerBench test. If you choose one of the mnemonics, such as SR, ServerBench will assume you want the transaction to be a singleton; i.e., a single test with the parameters you specify. To see a list of the test mnemonics, choose the down arrow next to this box.
4. Enter the number of times you want the server to execute the test within that transaction in the Test Iterations box. If you are creating a singleton, leave this box blank. ServerBench will only execute the test once.
5. If you did not enter a test mnemonic as your transaction type, then enter in the Test Type box the mnemonic for the test you want ServerBench to run in this

transaction. To see a list of the test mnemonics, choose the down arrow next to this box.

Figure 10-9: The tests



ServerBench automatically displays this box if you enter an incorrect mnemonic in the Test Type box.

6. Enter a value in bytes for the Request Size. The disk tests and the network tests require this parameter (if you're creating a processor test, ServerBench automatically puts N/A in this box). This is the amount of data ServerBench passes to your server's operating system at one time (i.e., this is the value the ServerBench application on the server uses for the size parameter when it requests file I/O or network I/O). For example, if you select a disk random read test with a Request Size of 512 bytes, ServerBench will read data from your server's disk in increments of 512 bytes.

NOTE: If the value you supply here is larger than what your network will handle in a single packet, then your network will break this packet into a smaller size.

7. Enter a value for the Total Size. If you're setting up a Processor test, this value will be 1 to 2800 iterations. For the other tests, this value will be 1 to 16777215 bytes. For the Processor test, this is the number of times you want ServerBench to perform the Processor test each time it executes that

transaction test. For the disk tests and the network tests, this is the total amount of data that you want ServerBench to move for that test. For example, if your Total Size for the random read disk test is 51200 bytes and your Request Size is 512 bytes, ServerBench would read data from your server's disk in increments of 512 bytes until it had read a total of 51200 bytes of data.

8. Enter a value from 0.001 MB up to 1024.000 MB for the Disk Test File Initial Size. This parameter tells ServerBench how big to make the data files it creates for the disk tests. During the Append test, these files can grow beyond this initial size.
9. Enter a value from 0.001 MB up to 1024.000 MB for the I/O Range Size. This parameter tells ServerBench to set aside a contiguous section of the data file and only use that section for file I/O.
10. To add another test to your current transaction:
 - a. Choose the New Test button.
 - b. Go to the Test Iterations box; i.e., TAB to the Test Iterations box. (*Do not* enter anything in the Transaction Iterations or Transaction Name boxes, or ServerBench will set up another transaction.)
 - c. Now follow steps 4 through 9.
 - d. Repeat these steps until you have included all the tests you want to include in this transaction.

You can delete a transaction from your list of transactions by selecting the transaction (i.e., highlighting it) and then choosing Delete.

How the parameters affect the transactions

The values you enter when you set up the parameters affect how much the transaction stresses your server. You can increase or decrease the effect the transaction will have on your server by the values you enter when you set up the transaction. Here's some information on how the parameters can affect your server's results.

Number of transaction iterations

This parameter tells each client how many times to ask the server to perform that transaction during one iteration of the mix. For example, if you specify three transaction iterations, each time ServerBench executes one iteration of the test mix, every client in the mix will request that transaction three times. If ServerBench performs four iterations of the test mix, by the end of the mix, each client will have requested that transaction 12 times.

The higher the value you enter here, the greater the weight that transaction will have in the test results. In addition, the more times a client requests a transaction during an iteration of a mix, the longer one iteration of the mix takes.

Number of test iterations

This tells ServerBench how many times to execute that test once it receives the transaction request. For example, if you enter a value of 4 here, then each time a client requests this transaction, ServerBench will perform this test four times.

Thus, at the end of five iterations of the test mix, each client will have requested this transaction five times; however, ServerBench will have performed the test in the transaction 20 times.

Within one transaction, you can have multiple tests. Each test can have a different value for test iteration. So you might have a transaction that includes the following tests: 1 Client to Server, 3 Processor, 1 Sequential Read, and 5 Server to Client. In this case, when ServerBench executes this transaction one time, it performs 10 tests.

The more iterations you specify for a test, the longer you make the transaction. Because each client measures the amount of time it takes the server to get a transaction and then send a reply, you may see lower TPS scores. This is natural once the transactions have become larger and thus will take longer to execute. It's even conceivable that some clients may be starved out.

However, the other side of the coin is that, by combining tests in a transaction, you avoid the overhead of the network that you get if a client requested each test separately. To continue the previous example, one network request and reply resulted in data from 10 tests.

How the Request Size parameter affects your results

The disk tests and the network tests all require you to enter a value for the Request Size. This parameter tells ServerBench the amount of data to request in one I/O operation for a disk or network test. For example, if you select a random read test with an Request Size of 512 bytes, ServerBench will read data from your server's disk in increments of 512 bytes.

NOTE: This is only the size that the ServerBench application passes to the operating system. The operating system may actually perform the request in larger or smaller amounts.

How the Total Size parameter affects your results

The disk tests and the network tests all require you to enter a value for the Total Size. The Total Size parameter tells ServerBench how much data in bytes to move for that test. For example, if your Total Size for the random read disk test is 51200 bytes and your Request Size is 512 bytes, ServerBench would read data from your server's disk in increments of 512 bytes until it had read a total of 51200 bytes of data. The larger the value, the more intense the test is and the longer it takes to execute the transaction.

How the number of iterations for the Processor test affects your results

The more iterations you specify as the Total Size value for the Processor test, the larger the CPU load that the test places on the system. By creating tests with different Processor test iterations, you can improve the way ServerBench stresses the processor subsystem on servers with different processor power.

How the Disk Test File Initial Size parameter affects your results

The disk tests use this parameter to determine how large the test data files are. ServerBench creates a test data file the size you specify here for each client running the mix. Thus, you can't consider the effect of this parameter without also taking into account the number of clients in the test. By multiplying the value in this parameter by the number of clients, you determine the size of your test area. For example, if you have 20 clients and you set this parameter to 1 MB, then your test area is 20 MB. If you set this parameter to 20 MB, then your test area is 400 MB.

The main impact that the combination of the Disk Test File Initial Size and number of clients has is to determine how much of the server's disk you test. The larger this parameter and the more clients running the mix, the greater the amount of disk the tests cover. If the test area is too small, then it will never break out of the server cache. In that case, your test results will be higher because ServerBench isn't actually using your disk subsystem; however, the test won't be as representative of how your system performs as it should be.

Thus, as your test area gets larger – either because you increase this parameter or you keep it static and increase the number of clients – the server can no longer service the client requests from cache. As a result, more disk accesses occur. Now your server's disk subsystem is having a large influence on your server's scores, which is one of the reasons for running a disk test.

To see how much of an influence the disk subsystem has on your results, you may want to run a disk test both ways – in server cache and out of server cache.

The larger disk files also mean that as your test area covers more of the disk, the server will have to perform more spread out disk operations. For example, the seeks that the tests perform will cover a greater distance. This results in more disk seeks during the test.

How the Disk Test I/O Range parameter affects your results

The I/O Range parameter, which is new with ServerBench 3.0, lets you specify the how much of the test data file you want ServerBench to use when it performs disk input/output operations. This affects the amount of cache ServerBench uses.

You can set this parameter to any number from 1 MB to the maximum size of the data file. The ServerBench standard test suites set the I/O range parameter to 10 MB for a 20 MB data file.

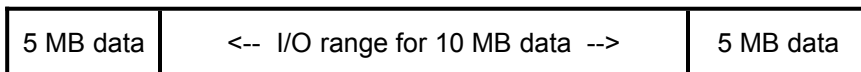
NOTE: If you're using a test suite that you created with an earlier version of ServerBench, then the default value for the I/O range parameter is the entire size of the test data file. This maintains consistency with previous versions of ServerBench. To change the value of that I/O range parameter, you'll need to modify the test suite.

Here's how the I/O range parameter works

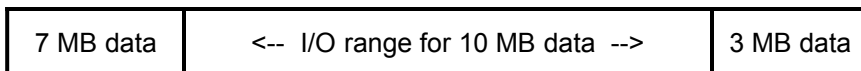
The I/O range parameter is another ServerBench feature designed to give you more control over the way you test your server. This parameter lets you increase the size of your test data files without having to increase the amount of file cache your server needs to perform the test.

The I/O range parameter sets aside a contiguous section of the test data file (i.e., a range) of the size you specify. ServerBench then performs all its I/O operations in this range; it does not perform file I/O operations anywhere else in the data file.

The starting point for the I/O ranges varies for each client. For example, if each client has a 20 MB data file with a 10 MB I/O range, then the I/O range for Client1's data file might look like:



while the I/O range for Client2's data file might look like:



With earlier versions of ServerBench, if you wanted to use more of your server's disk, you needed to increase the size of the test data files. As the size of the data files increased, they used up the available file cache quicker. This meant the

ServerBench tests broke out of cache quicker. For example, four clients that each had a 20 MB data file would try to use 80 MB of file cache. The only way to decrease the amount of file cache was to reduce the size of the data files.

With ServerBench 3.0, you still must increase the size of your test data files when you want to test more of your disk; however, you can adjust the amount of file cache that the tests use. The I/O range parameter lets you determine how much file cache each client uses. Now, if you have 20 MB data files for four clients, you can specify an I/O range 7 MB. This means you would use 28 MB of file cache instead.

Some different types of transactions

A transaction is the amount of work the client asks the server to perform before returning results. The following three examples illustrate how you can use transactions and the types of transactions (multiple tests, singletons, and singletons with multiple iterations, and multiple tests) you can create.

In ServerBench's standard mixes, most transactions consist of multiple tests; i.e., each transaction contains more than one individual test.

Here is an example of transactions containing multiple tests.

Figure 10-10: A transaction with multiple tests

Transaction Definitions					
Transaction Iterations	Transaction Name	Test Iterations	Test Type	Request Size (bytes)	Total Size
1	TR1	1	CS	500	500
		15	P	N/A	50
		2	RR	2048	2048
		15	P	N/A	50
		2	RW	2048	2048
3	TR2	1	SC	2048	2048
		1	CS	500	500
		7	P	N/A	50
		2	RR	2048	2048
		7	P	N/A	50
		2	RW	2048	2048
		1	SC	2048	2048

Disk test file initial size (MB):
 Disk test file I/O Range (MB):

This time, during one iteration of the mix, the client will request this TR1 one time and TR2 three times. For each iteration of TR1, the server will perform one Client

to Server test, two Random Read tests, fifteen Processor tests, two Random Write tests, and one Server to Client test (i.e., a total of thirty-six individual tests). For each iteration of TR2, the server will perform one Client to Server test, two Random Read tests, fourteen Processor tests, two Random Write tests, and one Server to Client test (i.e., a total of 20 individual tests). The client records the time each transaction takes from the moment it issues the request until it receives a response from the server. The client then updates its transaction logs.

If you want to create a transaction that is a singleton (i.e., one test), use the test mnemonic as the transaction name. The SR transaction (or Sequential Read) is an example of that type of transaction.

Figure 10-11: Creating singletons

Transaction Definitions					
Transaction Iterations	Transaction Name	Test Iterations	Test Type	Request Size (bytes)	Total Size
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
6	SR			1024	51200

Disk test file initial size (MB):
 Disk test file I/O Range (MB):

This transaction definition says that the client will request one SR transaction six times during one iteration of the mix. The transaction consists of one sequential read test that uses a Request Size of 1024 bytes and a Total Size of 51200 bytes. These parameters tell the server to read 51200 bytes sequentially from the data file in 1024-byte chunks each time it executes this transaction. Each time the client requests this transaction, the client records how much time the server takes to perform the transaction.

You can also define a third type of transaction, one where a single test performs multiple iterations.

Figure 10-12: A transaction with multiple iterations of a test

Transaction Definitions					
Transaction Iterations	Transaction Name	Test Iterations	Test Type	Request Size (bytes)	Total Size
3	1PROC	5	P	N/A	1024

Disk test file initial size (MB):
 Disk test file I/O Range (MB):

This transaction, named 1PROC, calls for three transaction iterations during a single mix iteration. Each time the client issues a request for this transaction, the server performs five Processor tests and then responds to the client. In other words, during one iteration of the mix, the client will issue three separate transaction requests where each request requires the server to execute five processor tests (i.e., a total of 15 Processor tests during a single iteration of the mix). And each of the five Processor tests will perform 1024 iterations. Request Size isn't included because it's not relevant to processor tests. Each time the client requests this transaction, the client records how long it takes to get a response from the server.

Providing the basic client information

There are two pieces of information that ServerBench looks for in the Client Information section of the Mix Definition window. They are:

- **Total Number of Clients.** Enter the maximum number of clients you want to have running this test mix. If you don't have this many clients connected, the mix will run with as many clients as you do have connected. (Each mix in the ServerBench standard test suites increments the number of clients running the tests.)
- **Data file path name.** This is the path name to the place on the server where you want ServerBench to create the test data files that the clients use for

disk I/O during the disk tests. By default, ServerBench creates these files in the ServerBench directory on the server. The standard test suites use the file names **DATA1**, **DATA2**, and so on up to **DATA60**. If you enter a full path name in this field, make sure the directories you specify exist before you execute any disk tests.

Figure 10-13: The Client Information

Client Information	
Total Number of Clients:	<input type="text" value="4"/>
Client#	Data file pathnames:
<input type="text"/>	<input type="text"/>
1	data1
2	data2
3	data3
4	data4
<input type="text"/>	

You can also use the Mix Definition window to restrict the mix to specified groups of clients. See the section “Including client groups” for information on doing that.

How to enter the client information

The data file path name gives the full path and file name on your server of the data file for each client. You must supply a unique data file for each client. To do this, make sure you have privileges for the directory you specify and that the directory path is valid. ServerBench verifies the data file path name when you run the test.

To tell ServerBench how many clients you want the mix to include and where to find the data file path names:

1. Enter a value for Total number of clients. This is the maximum number of clients that ServerBench will include in this mix.

NOTE: If you don't have enough clients connected to the server, ServerBench will run the mix with fewer clients than the maximum you specified.

- 2 Supply the data file path names for each client. There are four ways you can do this. You can:
 - **End the path name with an asterisk.** When you enter the path name this way, ServerBench treats the path name as template. This option lets you create paths for all clients at once. ServerBench will automatically append a number to that path name and enter it as the path name for each client up to the maximum number of clients. So, if you specified 60 clients, and you entered the path name **test***, ServerBench would create the path names **test1, test2...** up to **test60**.
 - **Copy client paths from ASCII file.** You choose this option from the Advanced menu. This option tells ServerBench to copy paths from a text file that contains a list of path names. To create this file, enter each path name on a separate line and end the line with a carriage return. If the file contains more client path names than you specified, ServerBench resets the total client count to match the number of path names in the file. Thus, if you had specified a data file with 4 clients and you supply an ASCII file with 8 clients, ServerBench resets the total number of clients to 8 and enters all 8 path names.
 - **Copy fields from another mix.** You choose this option from the Mix menu. This option lets you copy the path names from an existing mix. Once you choose this option, ServerBench will ask you to choose an existing test suite. Then it will ask you to select the mix you want to copy information from. Once you have the mix highlighted, click on the Copy client information button. If the mix has a different number of clients than you specified, ServerBench resets the total client count to match the number in the mix, regardless of whether that number is higher or lower than the one you specified. For example, if you specified a total of 8 clients and the mix only has 4 clients, the ServerBench resets the total number of clients to 4. However, if the mix has 12 clients, then ServerBench increases your total number of clients to 12 as well.
 - **Type in each path name.** To do this, click on each client in the list box. ServerBench then displays that number in the Client# box and lets you enter a path name in the Data file path names box.

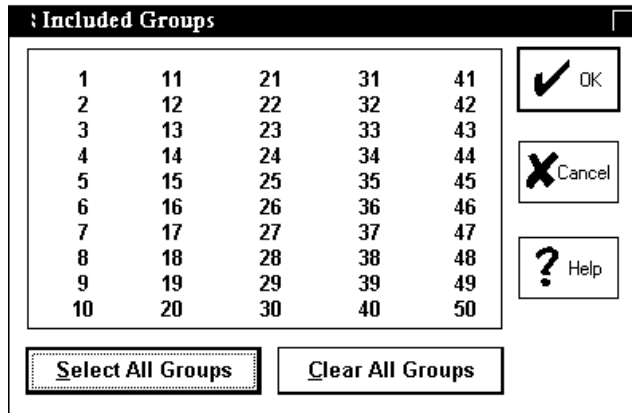
Including client groups

ServerBench lets you restrict the mix to certain client groups. You assigned group numbers to clients when you installed ServerBench on the clients. (For more information on setting up clients, see the on-line ServerBench manual for your server operating system.)

You don't have to specify any group numbers for a mix to run.

You get to the Included Groups dialog box by choosing the Groups option from the Advanced menu. To restrict the mix to certain client groups, click on the group number or numbers.

Figure 10-14: Defining groups of clients



You can also use the Groups function buttons to:

- **Select all** groups of clients to be in the mix (this has the same effect as selecting no groups). If you plan to use most groups in the mix, you can choose Set all and then just click on the group numbers you don't want included. When you click on a highlighted group number, ServerBench deselects it.
- **Clear all** the groups you had specified. Clear all lets you remove the current group assignments. You can then just click on the group numbers you want ServerBench to include in this mix.

How the client groups feature works

When you use client group numbers, you can run sets of clients together during a test. This way you can increase your server's work load incrementally. For example, you may want to run a test suite with 20 clients, then 40 clients, and then 80 clients. This way you can spot where the knee in the performance curve appears.

(Or you could just increase the number of clients in each mix and let ServerBench select which clients.)

You can also use the group feature to place clients in a single network segment in one group. Then, you can bring an entire segment on-line at once. To get information about work loads across segments, number the clients consecutively on each segment. Include equal numbers of clients from each group in the segment. Now create a mix that defines the number of groups you want and the number of clients from those groups.

If you specify any group numbers for the mix, ServerBench will only let clients with those group numbers run in the mix. For example, suppose you have groups 1 through 5, each of which has 10 clients. When you defined the mix, you specified that it include 50 clients but only from groups 1 and 2. Because you specified groups 1 and 2, ServerBench will only use clients in those groups. Since both groups combined only have 20 clients, 20 is the maximum number of clients that ServerBench will use for this mix, even though you specified a total of 50 clients when you set up the mix.

If you don't specify any group numbers or if you have more clients than the mix needs, the server decides which clients to use based on the client ID number. ServerBench always uses the lowest ID numbers first. This means that, if you have a client with the ID number 1, that client will always run the tests anytime you don't specify a group number.

Even when you specify a group number, ServerBench gives the clients with lower numbers priority over the clients with higher numbers. For example, suppose you're using clients from Group 1 and Group 1 includes clients with ID numbers 10 through 25. In this case, client 10 always runs a test, regardless of whether you specify one client in the mix, four clients, or 25 clients.

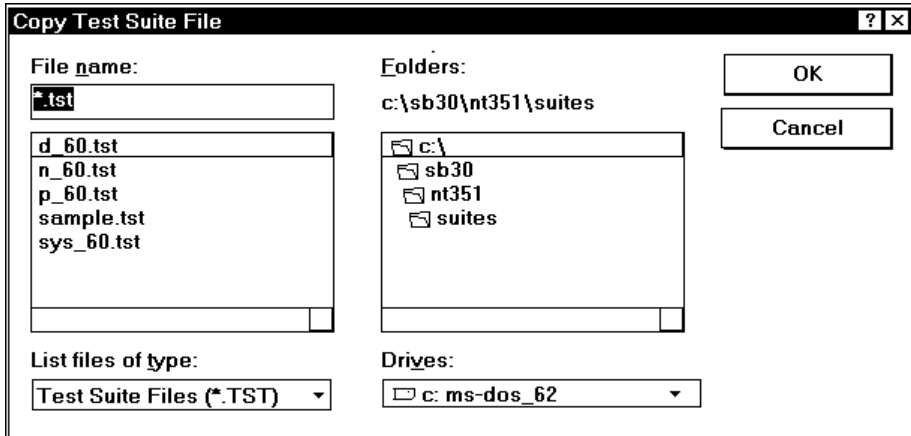
This means that, if you connect more clients than a mix uses, you will always get the same set of clients for a given mix.

Using another mix to set up the current mix

When you're working with a mix in the Mix Definition window, you can use the Copy fields from another mix option in the Advanced menu to replace the current values in the Mix Definition window with the values in an existing mix. When you choose this option, ServerBench lets you copy either one entire mix or the values you specify.

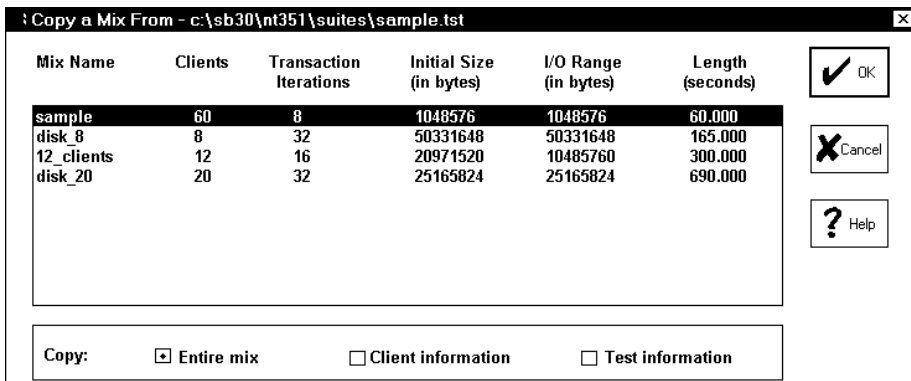
After you select the Copy fields from another mix option, ServerBench displays the Copy Test Suite File dialog box. Highlight the test suite you want to use. (You can only select one test suite.)

Figure 10-15: Copy Test Suite File dialog box



Once you click on OK, the Copy a Mix from dialog box appears.

Figure 10-16: Copy a Mix from... dialog box



This dialog box displays information about the mixes in that test suite. It gives you the mix names, the number of clients in each mix, the number of transactions each mix performs, the value for the Disk test file initial size parameter, the value for the I/O range parameter, and the value of the Length parameter.

You now have three options

- **Entire mix.** ServerBench copies all the values in the selected mix to the Mix Definition window.
- **Client information.** ServerBench copies only the client information to the Mix Definition window.
- **Test information.** ServerBench copies only the transaction definitions to the Mix Definition window.

NOTE: If you want to copy entire mixes at one time, use the Copy mixes button on the Mixes in Suite dialog box.

Copying mix fields across a test suite

ServerBench lets you duplicate mix parameters across all the mixes in the test suite. To do this choose the option Copy mix fields across this suite from the drop-down Advanced menu in the Mix Definition window.

When you choose this option, ServerBench displays the Duplicate Mix Fields Across Suite dialog box. This dialog box lets you copy the value for one parameter or multiple parameters from the current mix to all the mixes in the test suite.

The options you can choose at this dialog box you are:

- **Duration information.** You can specify the Ramp up, Ramp down, and/or Length parameters.

NOTE: When you change the value for one parameter, consider how that parameter interacts with other parameters. It is possible that, if you globally update only one parameter across all the mixes, you may cause an error condition for another mix. For example, suppose you change one field, such as Length, without changing the other fields that work with Length — Ramp up and Ramp down. Your current mix (Mix 1) uses Ramp up and Ramp down periods of 30 seconds and a Length of 90 seconds. Mix uses Ramp Up and Ramp Down values of 1 minute each. If you tell ServerBench to change the Length value for the mixes to match the value for Mix1, you'll create an error condition for Mix2 because the Length of the test will in Mix2 no longer exceed the combined values of Ramp Up and Ramp Down.

- **Timing information.** You can choose the Think and/or Delay parameters.
- **Transaction definitions.** If you choose the Transaction Definitions option, ServerBench replaces the transactions in the other mixes with the transactions in the current mix. However, it does not affect the client information in the

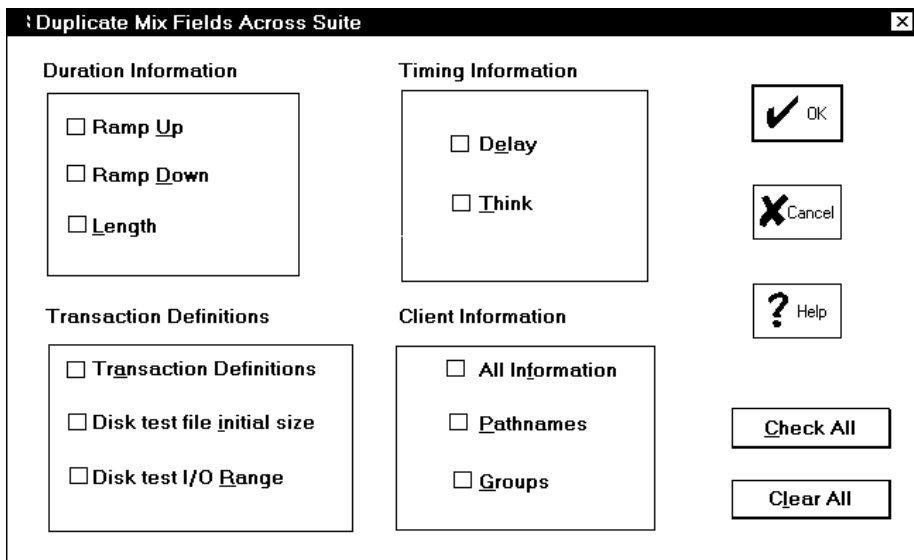
mixes. Make sure you want all the transactions to be the same in all the mixes before you choose this option.

If you choose the Disk test file initial size and/or the Disk test I/O range, ServerBench updates all the mixes so that they have the same value as the parameter (or parameters) you've chosen.

- **Client information.** If you choose All information, ServerBench makes the client information in all the mixes identical. You will now have the same number of clients in each mix with the same path names. Suppose you have three mixes in a test suite. Mix1 has 8 clients, Mix2 has 16 clients, and Mix3 has 32 clients. You're currently editing Mix2. If you choose All information, then each mix in this test suite will have 16 clients. All the path names and group designations in the three mixes will match the path names and group designations you specified in Mix2.

To change only one part of the client information without affecting the number of clients in the mix, choose either Pathnames or Groups. If you choose the Pathnames option, ServerBench will change only the client path names in the other mixes and only for as many clients as the current mix has. In other words, if the current mix is Mix2 with 16 clients, ServerBench would only change the first 16 path names in Mix3. The path names for clients 17 through 32 would remain untouched.

Figure 10-17: Duplicate Mix Fields Across Suite dialog box



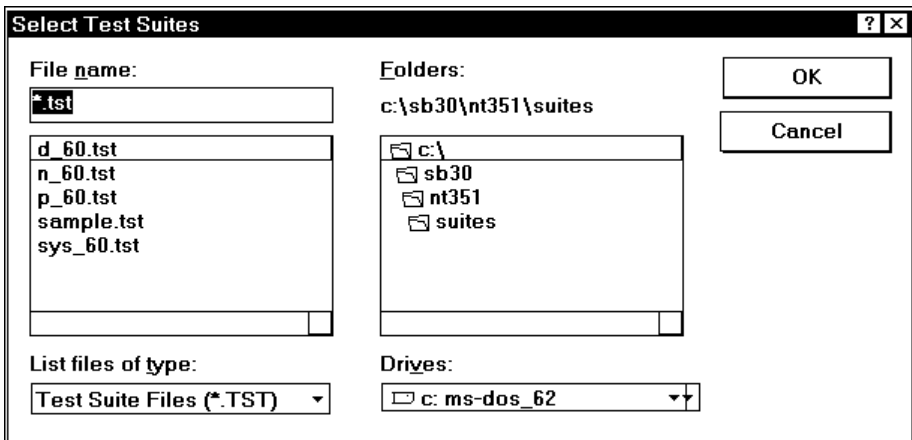
Copying mixes from an existing test suite

When you choose the Copy button from the Mixes in Suite dialog box, ServerBench lets you specify an existing test suite. It then displays all the mixes in that suite. You can highlight the mixes you want to add to your current test suite in the Mixes in Suite dialog box, and ServerBench will place those mixes after the current mix in the Mix List.

NOTE: If you just want to copy some of the information from one mix directly to the Mix Definition window for you to edit, use the Copy fields... option in the drop-down Advanced menu in the Mix Definition window.

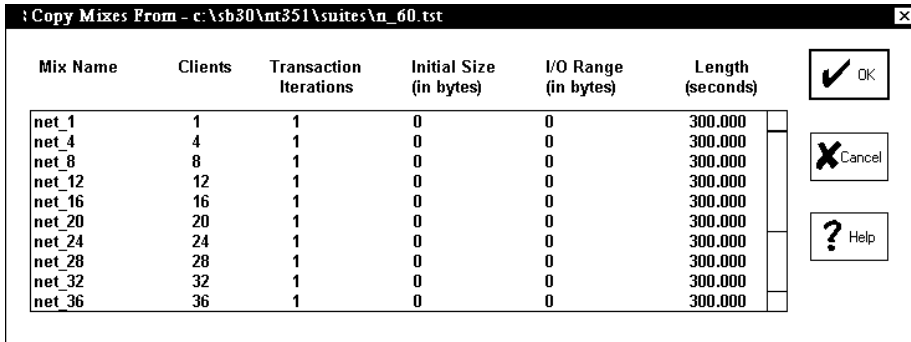
ServerBench displays the Select Test Suites dialog box when you choose the Copy button.

Figure 10-18: The Select Test Suites



Go to the directory containing the test suite you want and select that suite. When you press OK, ServerBench displays the Copy Mixes From dialog box.

Figure 10-19: The Copy Mixes From... dialog box



This dialog box displays information about the mixes in that test suite. It gives you the mix names, the number of clients in each mix, the number of transactions each mix performs, the value for the Disk test file initial size parameter, the value for the I/O range parameter, and the value of the Length parameter.

Highlight the mix or mixes you want to copy and then click on OK.

ServerBench returns you to the Mixes in Suite dialog box and places the mixes you've chosen in the Mix List after the highlighted mix.

Inserting new mixes into a test suite

When you're at the Mixes in Suite window, you can tell ServerBench where to place the mix you're creating. All you need to do is highlight in the Mix list the name of the mix that you want the new mix to follow. Now click on the New button. ServerBench will insert the mix you're creating right after the highlighted mix.

You can also insert a new mix while you're at the Mix Definition window. To do this choose the option Insert mix from the Mix menu. ServerBench will insert the new mix immediately after the current mix.

Reordering mixes

When you're at the Mixes in suite window, you can also change the order of the mixes in the test suite. All you need to do is highlight the name of the test suite you want to reorganize. Now click on the Reorder button. ServerBench displays the

Reorder dialog box. Move the mixes from the Previous Mixes list to the New Order list.

NOTE: Before you use this feature, make sure all your mixes have unique names.

Here're the steps for reordering the mixes in a test suite:

1. Go to the Test Suite window and highlight the name of the test suite in the Suite List. ServerBench displays all the mixes currently in this test suite in the Mixes in Test Suite list.
2. Click on Re-Order. ServerBench displays the Reorder Suite Mixes dialog box. This box lists the name of the suite in the title bar and displays the mixes in their current order in the Previous Mixes list.
3. In the Previous Mixes list, click on the mix that you want to be first and then choose Add. If you select multiple mixes, ServerBench will move those mixes to the New Order list based on their current order. For example, suppose the Previous Mixes list contains Mix1, Mix2, Mix3, Mix4, Mix5, and Mix6. You highlight Mix3, Mix5, and Mix6 and click on Add. ServerBench will place those mixes in that order in the New Order list.

NOTE: As with the New button, ServerBench inserts the mixes you're adding to the New Order list after the currently highlighted mix instead of simply appending the mixes to the list.

4. Continue moving the mixes to the New Order list until you've moved all the mixes you want and they're in the order you want. Then choose Done.

Deleting mixes from a test suite

You can delete mixes from the test suite when you are at the Mixes in Suite window.

To delete a mix from a test suite, click on the mix name and then choose the Delete button. ServerBench asks you to confirm that you want to delete the mix. If you answer yes, it deletes that mix permanently.

End of chapter

Chapter 11

The Details on ServerBench's Tests

This chapter gives you the facts about what the ServerBench individual tests do. It explains:

- What each test does.
- What ServerBench's standard test suites do.

Understanding ServerBench's tests

ServerBench provides eight tests that let you measure how well your application server handles requests from clients. You use these tests to create the transactions that make up the test mixes and, ultimately, the test suites (the test suite is what you actually tell ServerBench to execute).

The individual tests fall into three categories:

- Processor. One processor test.
- Disk. Five disk tests. Sequential Read, Sequential Write, Random Read, Random Write, and Append.
- Network. Server to Client and Client to Server.

You can specify one or more tests in a transaction. In addition, you can vary the test parameters for each test. For example, you can specify different Request Sizes for the disk read and write tests.

NOTE: Test parameters differ from mix parameters. You can have multiple individual tests in a mix where you vary the parameter values for each test. However, the values for the mix parameters will stay the same for all

the tests in that mix. These mix parameters include Length, Ramp up, Ramp down, Delay, Think time, the number of clients running the tests, and the path names for the disk test data files.

The next sections describe the individual ServerBench tests and the different parameters that affect them.

The Processor test

The processor test is a general measure of a server's ability to do work without I/O. The test does not analyze specific types of operations the way the Linpack and Whetstones benchmarks do. Instead, it performs all the common types of instructions. These include data searches, string manipulation, integer arithmetic, memory copying, and so forth. This test uses common programming constructs, such as if-else, looping, and procedure calls.

Each processor test that the server runs consumes a large amount of RAM (about 400 KB).

The bottom line is that the processor test tries to have the CPU do as many different types of operations as possible. We've created it so that it mimics the processor-intensive portions of typical database servers.

In addition, when you include a Processor test in a transaction, you enter a value of from 1 to 2800 iterations in the Total Size parameter. By specifying the number of iterations, you have more control over the CPU load the test places of the server.

The processor test works by manipulating a 125-byte record. This record contains the following fields:

- A Row ID, a two-byte integer
- A Customer ID, a two-byte integer
- A Salesperson ID, a two-byte integer
- Available credit, a four-byte integer
- Amount bought, a four-byte integer
- Outstanding balance, a four-byte integer
- Order pending, a one-byte flag field
- Business type, a one-byte enumeration 1 to 10
- The customer's name, a 20-byte character string
- The customer's address, a 75-byte character string

- The customer's phone number, a 10-byte character string

The data set is 2800 records of 125 bytes for each client for a total of 350,000 bytes of data for each client.

When you install ServerBench, it creates this data set on the server. Then, when you tell ServerBench to run the Processor test, the server allocates 350,000 bytes of memory for each client and reads the data set into the client's work area. The data set is memory resident. This means that, after the initialization stage when service providers read the data set into memory, the Processor test does not perform any explicit I/O on the data set. All of the data manipulations that the Processor test performs occur on the memory resident image of the data set.

NOTE: ServerBench only allocates the memory for the data set if a client requests the Processor test.

ServerBench manipulates the data file by:

1. Totaling the outstanding balances of each salesperson's customers.
2. Counting the number of customers who have exhausted their credit.
3. Dividing the total amount sold by the number of people to produce an average.
4. Searching the data for business type "1" and copying that data into the temporary buffer.
5. Searching the data for customers with outstanding orders, sorting the data, and copying that data into the second temporary buffer.
6. Looking at the amount of goods purchased for both of the subsets described above and, for the complete set of records, computing the minimum, maximum, and average amount of goods purchased in each group, and then totaling the amount of goods purchased in each group.
7. Giving a total amount due the company.

Because ServerBench doesn't display the final results of this test, it doesn't format the data.

When you include a Processor test in a transaction, you set the Total Size parameter. You can specify from 1 to 2800 iterations. By entering the number of iterations, you have more control over the CPU load the test places on the server.

The disk tests

The five disk tests help you determine how well your disk subsystem is performing. These tests are Sequential Read, Sequential Write, Random Read, Random Write, and Append.

NOTE: The Append test can affect other tests. This is because it can fragment disks as it appends data to existing files.

When you tell ServerBench to run a disk test, it creates the test data files it needs for the tests. ServerBench keeps those files for the entire ServerBench session and uses them with the next disk tests. When you halt ServerBench, ServerBench deletes these test files.

NOTE: If you want ServerBench to recreate the test data files for each disk test, you can include a **-F** option on your server command line. See the on-line ServerBench manual for your server operating system for more information.

When you're setting up a disk test, you can specify the following parameters:

- **Data File Pathname.** This is a path name to the location on the server where you want ServerBench to create each client's Disk test data file. Each client must have a separate path name. If you enter a simple path name, ServerBench creates these files in the ServerBench directory on the server. For example, the standard test suites use the path names **data1**, **data2**, and so on. If you specify a directory in these path names, make sure that directory exists before you run the disk tests.
- **Data Test File Initial Size (MB).** The default is 1 MB. This parameter tells ServerBench how big to make the data files it creates for the disk tests. During the tests, especially the Append test, these files can grow beyond this initial size.
- **Disk test I/O Range (MB).** The default is 1 MB. This parameter tells ServerBench how much of the disk files to use when it performs input/output operations during a disk test. The I/O range is a contiguous chunk of data in a file. While the size of the I/O range is the same for each client, the location of the range within the data file varies from client to client.
- **Request Size.** This is the amount of data in bytes you want ServerBench to move at one time. For example, if you select a Random Read disk test with a Request size of 512 bytes, ServerBench will read data from your server's disk in subsets of 512 bytes. If you're running the Append test, then ServerBench will append text to the test files in chunks of 512 bytes.

NOTE: The Request size is the amount of data ServerBench passes to your server's operating system as a single unit. The operating system may perform the actual I/O with a larger or smaller size.

- **Total Size.** This is the total amount of data in bytes you want ServerBench to move during the test. To continue the previous example, if your Total size for the Random Read disk test is 51200 bytes, ServerBench would read data from your server's disk in subsets of 512 bytes until it read a total of 51200 bytes of data.

Each of the disk tests starts by having ServerBench create a test data file for each client based on a seed file that is included in the ServerBench package. ServerBench makes this file the size you specify in the Initial Size parameter. ServerBench uses data from the seed file located on the server to fill the data files.

NOTE: ServerBench does not include the amount of time it takes to create the test data files as part of the test time. It does, however, include the amount of time the Append test spends adding text to a file as part of the test time.

A key feature of all of the Disk tests is that they pick a random offset in the test data files as the starting point for whichever operation (read or write) they'll be performing. Each client generates a random number for the starting offset. Because the number can vary from client to client, the tests don't all start at the same place in a disk file. Once a client generates the random offset, ServerBench checks to make sure there is enough space from the random offset to the end of the file to meet the Total Size requirements of the test without forcing a read operation to read past the end of the file.

When performing the disk tests, the server aligns its read and write operations on boundaries based on the value in the Request Size parameter. For example, if the Request Size parameter is 2048, then the server will align the read and write operations on 2048-byte boundaries.

The disk read tests all work by reading data from this test file. The amount of data they read at a single time is based on the size you specify in the Request size parameter. After reading a chunk of data, they discard it. They continue to read data until they have read the amount of data you specified in the Total size parameter. If a client requests a Random Read test, each read request occurs at a different offset. If a client requests a Sequential Read test, each read request occurs sequentially.

The disk write tests each read a chunk of data from the file. The size of the chunk is based on the value you specify in the Request size parameter. They then modify this data and write it back to the same place in the file. If a client requests a Random Write test, ServerBench picks random spots in the file to read a chunk of data and then write it back. If a client requests a Sequential Write test, ServerBench moves through the file sequentially.

The Append test asks the server operating system for a fixed amount of space to append to the existing file. It then fills that amount of space with text from test data pages. The Append test determines how much data to add to the file based on the value in the Total Size parameter. The Append test continues to add data to the file in chunks that are the size specified by the Request Size parameter until it has added the amount specified by the Total Size parameter.

Because the Append test does not request that all the data be contiguous, it can fragment disks as it increases the size of the existing data file.

For information on how the values place in these test parameters affect your results, see Chapter 10 “Creating Your Own Test Suites.”

Table 11-1 briefly describes the functions of each disk test.

As you use the disk tests, keep in mind that ServerBench:

- Creates test data files only if the mix you execute includes transactions with disk tests. ServerBench uses a seed file it provides to create these files at the locations you specify in the Data File Pathname portion of the Mix Definition window.
- Fills the test data files before the mix starts. ServerBench does not include this time in the tests' time.
- Requests additional file space and then fills this space with data when it runs the Append test. ServerBench includes the time it takes to append data to the file as part of the total time it takes the test suite to run.
- Does not worry about concurrent file access because there is an exclusive data file for each client in the test.

Table 11-1: Functions for each disk subsystem test type

Sequential read	<ol style="list-style-type: none"> 1. Selects a random starting point 2. Reads the specified Request size of data 3. Discards the data 4. Repeats steps 2 and 3 until it meets the Total size 5. Sends an acknowledgment to the client
Sequential write	<ol style="list-style-type: none"> 1. Selects a random starting point 2. Reads the specified Request size of data 3. Changes that data 4. Writes the data back out using the same address 5. Repeats steps 2 through 4 until it meets the Total size 6. Sends an acknowledgment to the client
Random read	<ol style="list-style-type: none"> 1. Selects a random starting point 2. Reads the specified Request size of data 3. Discards the data 4. Repeats steps 1 through 3 until it meets the Total size 5. Sends an acknowledgment to the client
Random write	<ol style="list-style-type: none"> 1. Selects a random starting point 2. Reads the specified Request size of data 3. Changes that data 4. Writes the data back out using the same address 5. Repeats steps 1 through 4 until it writes the specified Total size of data 6. Sends an acknowledgment to the client
Append	<ol style="list-style-type: none"> 1. Requests additional file space equal to the Total size 2. Writes data in chunks equal to the specified Request size to the file space 3. Once it has written enough data to satisfy the Total size parameter, it sends an acknowledgment to the client

The network tests

We designed ServerBench's network tests to execute at the OSI (Open Systems Interconnect) transport level. This means the network tests operate at a high level. The advantage of using tests at this level is that ServerBench can get test results closer to what you would see in a production system. For example, if you replaced Ethernet with Token Ring on your server, you'd see a difference in performance. ServerBench's network tests would give you an indication of what the difference would be. (ServerBench itself would not notice the difference and would run the tests the same way regardless of whether you used Ethernet or Token Ring.)

The two network tests that ServerBench provides are Client to Server and Server to Client.

With both these tests, the data transfer is only one way. In the Client to Server test, a client sends data to the server.

In the Server to Client test, the server sends data to the client. First, though, the server must get a request from the client asking for the data.

In any networking setup, there're two main variables that affect the amount of network traffic — the rate of requests and the size of the requests.

With ServerBench, the following factors all affect the rate of requests:

- The number of clients involved in a test suite.
- The percentage of the individual tests devoted to the network tests.
- The value you supply for Think time.

The two key factors in determining the size of the request are the value you specify for the Request size parameter and way your network protocol implements packet sizes. The Request size indicates how much data you want ServerBench to transfer across the network in a single chunk. However, if the size of this transmission is too large, the network protocol must fragment, sequence, transmit, and reassemble the data at the other end. If it is too small, the network protocol may hold onto it and bundle it with another small transmission.

You can tailor the network tests by specifying the size of the data ServerBench transfers in the tests. By varying the size of the transmissions, you can define the test to more closely resemble the network activity of a production server.

The standard ServerBench test suites

ServerBench comes with four standard test suites and one demonstration test suite (**SAMPLE.TST**). You'll find these test suites in the ServerBench subdirectory **SUITES** on the controller. This section describes the standard test suites (for information on **SAMPLE.TST**, see the next section).

Each of the four standard test suites consists of 16 mixes that go from 1 client to 60 clients in groups of four.

The standard system test suite roughly balances the amount of processor and disk subsystem load it places on a server, along with a modest amount of network work. It exercises your system as a whole. For this test suite, we modeled the transactions in the mixes after typical database server applications. The mixes, however, put a greater load on the server than the typical database server application.

Each of the other three standard test suites focuses on an individual server subsystem. These disk, network, and processor test suites isolate and gauge the peak performance of each of these subsystems.

These are the standard test suites that come with ServerBench.

- **System.** The **SYS_60.TST** test suite combines processor, disk, and network tests in a manner that is modeled after the way typical database applications blend together these operations. This test suite uses 20 MB data files for the disk tests.
- **Disk.** The **D_60.TST** test suite focuses on the disk subsystem. It uses 20 MB data files for the disk tests with a ratio of three read tests to one write test. The mixes divide these tests equally between sequential file I/O and random file I/O. Each disk test reads and writes 8 KB of data in increments of 512 bytes, 1 KB, 2 KB, or 4 KB.
- **Processor.** The **P_60.TST** test suites exercises the processor subsystem. Each iteration of each mix in the processor test suite executes one transaction containing 100 processor tests. The value of the Total Size parameter for each individual Processor test is 280 iterations.
- **Network.** The **N_60.TST** test suite targets the network subsystem. Each iteration of each mix executes one transaction containing five Server to Client tests (i.e., a single transaction performs five iterations of the Server to Client test). The request size is 10 KB and the total size is 10 KB. Therefore, each transaction sends a total of 50 KB from the server to the client.

For specific information on the mixes and transactions these test suites use, look at these test suites in the Mix Definition window. To do this:

1. From the main ServerBench window, choose Create or Edit Test Suites.
2. At the dialog box that appears, go to the **SUITES** subdirectory and choose the test suites you want to look at.
3. At the Mixes in Suite window, highlight the test suite you want to look at.
4. Choose the Edit Test Suite button. This takes you to the Mix Definition window where you can see the different parameters for the mix and the transactions in it.

Looking at SAMPLE.TST

The goal behind the **SAMPLE.TST** test suite is to give you a way to quickly determine whether your server and clients are working. This test suite executes one mix containing eight transactions. Each transaction is a singleton; i.e., it only performs one test. The Ramp up time for the mix is 5 seconds; Ramp down, 10 seconds; and Length, 60 seconds. Think time and Delay are both 0. All the disk tests use an initial test data file size of 1 MB and an I/O range of 1 MB.

Table 11-2 lists the information about the transactions. Because these are singletons, the transaction name is the mnemonic for the individual test.

Table 11-2: The transactions in SAMPLE.TST

Transaction	# of Test Iterations	Request Size	Total Size
Proc	2	N/A	140
SR	1	512	4096
SW	1	512	4096
RR	1	512	4096
RW	1	512	4096
A	1	512	4096
SC	1	512	4096
CS	1	512	4096

End of chapter

Part 4

Checking Out ServerBench's Results

If you're interested in looking at, understanding, and publishing your results, then this is the part of the manual you want to read. The first chapter walks you through the results spreadsheets ServerBench displays. It tells you what each column on each table means. Next it explains how to set up a disclosure database that contains information on your server and clients — that is very important information you'll need to include if you want publish your ServerBench results (we thought we'd give you hand in getting the details of your system setup together by adding this feature to ServerBench 3.0).

Chapter 13 takes you into the next level of working with your results: It explains them to you. This chapter gives you the facts you need to interpret what those numbers in the ServerBench tables mean.

The last chapter in this part of the manual is crucial if you want to publish your results. It tells you what you need to do and gives you some examples of the disclosure information.

Chapter 12

Let's Take a Look at Those Results

This chapter explains how to read and understand the Excel spreadsheets that ServerBench uses to display your server's results. In this chapter you will find information on:

- Displaying your results using Excel spreadsheets.
- The types of results information that ServerBench produces.
- The disclosure information that ServerBench produces about your system.

Viewing test results

Each time you run a test suite, ServerBench saves information about the test in log files. It also records your disclosure information (i.e., data about how your system was set up when it ran the test). ServerBench bases the log file names on the name you entered for the results file at the Selected Suites dialog box. The default name is the name of the test suite you ran. For example, when you run the test suite **SAMPLE.TST** and you don't enter a results file name, ServerBench saves the information in the following log files:

- **SAMPLE.CLG** (client information).
- **SAMPLE.RLG** (overall results)
- **SAMPLE.TLG** (test suite description log)
- **SAMPLE.PLG** (log showing the command line parameters you entered when you started the ServerBench executable on the server)
- **SAMPLE.DLG** (log showing disclosure information for the server and the clients)

- **SAMPLE.ELG** (log showing errors clients reported; ServerBench only creates this file when you run it in unattended mode and choose the option to log errors)

While you can view the error log file from any text editor, the other log files aren't designed for direct viewing. As a result, ServerBench includes special Excel macros to format these logs into an Excel workbook that contains a series of tables. The seven tables ServerBench creates each contain different information about the test suite you ran.

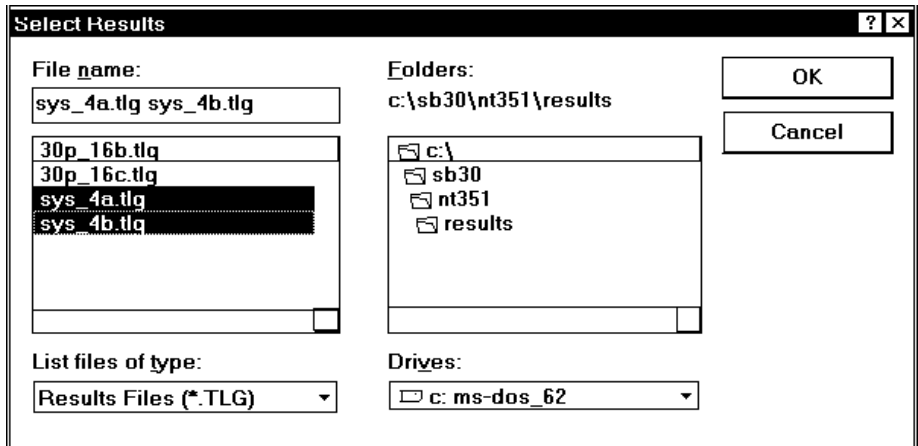
NOTE: Excel must be in your system path if you want to display your ServerBench results.

Here's how to display ServerBench's results workbooks

If you want to view your test results, here's what you do:

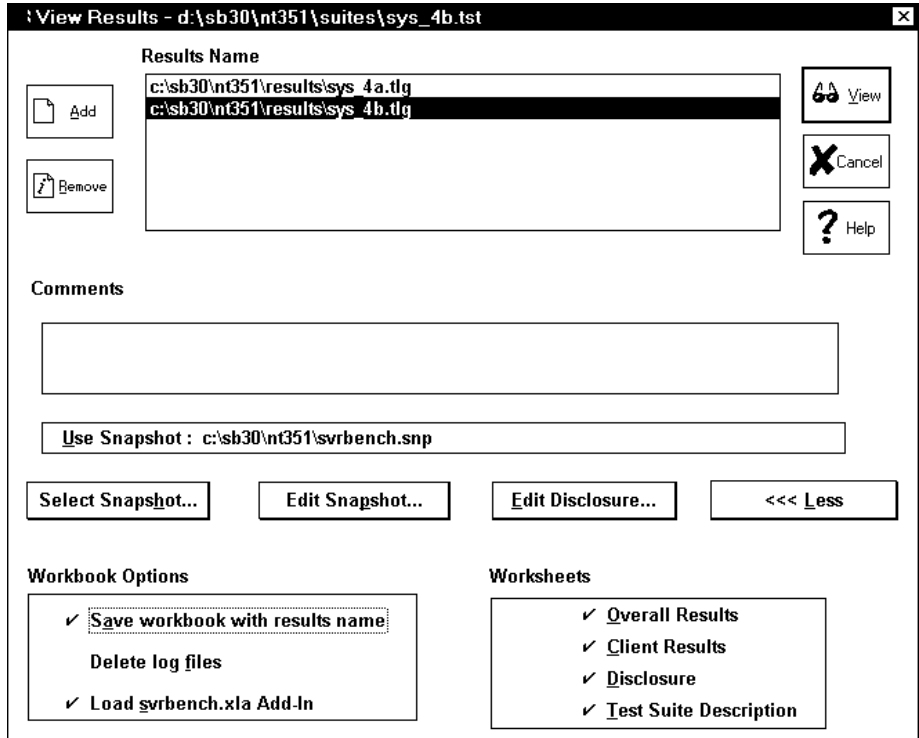
1. Go to the main ServerBench window.
2. Click on the View Results button. ServerBench displays the Select Results dialog box.

Figure 12-1: The Select Results dialog box



3. Highlight the name of the results files you want to view. You can choose one or more results file.
4. Choose OK. ServerBench displays the View Results dialog box.

Figure 12-2: View Results dialog box



5. At this dialog box you can:

NOTE: ServerBench uses the same options for all the results files listed in the Results Name box.

- **Add** more results files to your current list of files. When you choose the Add button, ServerBench displays the Select Results dialog box.
- **Remove** one of the currently selected results files. If you decide you don't want to view all the results files you've chosen, you can use the Remove button to take the files you don't want out of the list.
- **Use Snapshot**, if checked, tells ServerBench to use the snapshot file displayed in this box as the basis for creating the Server Disclosure and Client Disclosure results tables. If you don't check this option, ServerBench uses the information in the **.DLG** logs to create those tables.
- **Select Snapshot** lets you choose a different database snapshot file. ServerBench uses the information in the database snapshot file to create the Server Disclosure and Client Disclosure results tables. You need to supply the information in these tables if you want to publish your results.

You can create one snapshot file that contains all the information on your test setup and then use it for each set of results as long as your test setup stays the same.

- **Edit Snapshot** lets you modify the server and client disclosure information stored in the snapshot file specified by the Use Snapshot option.
- **Edit Disclosure** lets you modify the disclosure information that ServerBench captured about your server and clients. When you run a test suite, ServerBench places information it captures about the server and the client in the **.DLG** log files that it creates for that set of results. ServerBench displays this information in the Server and Client Disclosure forms when you choose Edit Disclosure. If you also have the Use snapshot option checked, ServerBench includes the information from that snapshot file in these forms as well as the **.DLG** information. You can add the **.DLG** information to your current snapshot file by checking option Update snapshot file... at the bottom of the Server and Client Disclosure forms. If you don't check this option, ServerBench won't save the information on the disclosure forms to the snapshot file.
- **Less/More** lets you choose how much of the View Results dialog box ServerBench displays. By default, ServerBench only displays the top portion of this dialog box; it doesn't display the Workbook and Worksheet options. If you choose More, ServerBench displays the entire dialog box and the name on the button switches to Less. Then, when you choose Less, ServerBench displays the abbreviated dialog box.
- **Save Workbook with results name.** When checked, this option tells ServerBench to automatically save each set of a results to an Excel workbook that has the same name as the results file. For example, if your results are called **SYS_4A.TLG**, Excel would save them to a workbook called **SYS_4A.XLS**.
- **Delete Log files.** When you check this option, ServerBench will delete the results log files it uses to generate the results tables after it creates the results spreadsheets. Then, if you don't save the spreadsheet using Excel (or didn't tell ServerBench to automatically save the spreadsheet), you won't be able to view those results again. In addition, if you didn't tell ServerBench to create all the results tables for that set of results, you won't be able to generate them later.
- **Load svrbench.xla Add-in.** When you check this option, ServerBench automatically loads its **SVRBENCH.XLA** Add-In module for Excel. This module lets you create a TPS summary graph and a variance summary graph. It also enables you to easily print the graphs or tables from within the ServerBench workbooks without printing other forms of data. When

you select this option, ServerBench adds a ServerBench option under Excel's Data menu.



Tip:

While you can permanently install the **SVRBENCH.XLA** module through Excel itself, we recommend that you install this module from the View Results window. This way, it won't affect your daily Excel work. If you install the module permanently in Excel, you'll notice that your Excel start-up time is slower even when you're not generating ServerBench results.

- Select the results tables you want to view. When you check the category, ServerBench displays those tables when it creates the results workbooks. Your options here are:

Summary	Table 1– ServerBench Summary. With this table, you'll also find the Test Information box and the TPS graph of your server's results.
Overall Data	Table 2–Overall ServerBench Data
Client Data	Table 3–Client Data
Disclosure	Table 4– Server Disclosure Table 5– Client Disclosure
Suite Definition	Table 6– Test Suite Information Table 7– Transaction Information.

Once you have all the results files you want to view and have set up database and workbook options, click on View and ServerBench will display your results.

NOTE: If you have a Excel session currently running, ServerBench will use that session of Excel to display the results workbooks. Otherwise, it will start a session of Excel.

Using the results workbook

When ServerBench creates the test suite log files, it writes your server's configuration information and results data files in different formats, such as ASCII comma-delimited format. ServerBench includes Excel spreadsheet macros that import and format these log files into tables that appear in a single Excel workbook. Once ServerBench has generated the test suite results, it opens Excel and displays this workbook for you to see.

You can print the tables from within Excel. You can also have ServerBench automatically save the results to an Excel file that you can open later using either Excel or an Excel-compatible spreadsheet program. (This is an option in the View Results dialog box.)

The results spreadsheet includes five reports with a total of seven tables. Each report contains different types of data.

The next sections explain what information is in the tables for each type of report.

Summary report

The Summary report contains the key ServerBench metric: the overall score for your server. It also includes information about the test as well as a graph showing how the server performed.

Table 1

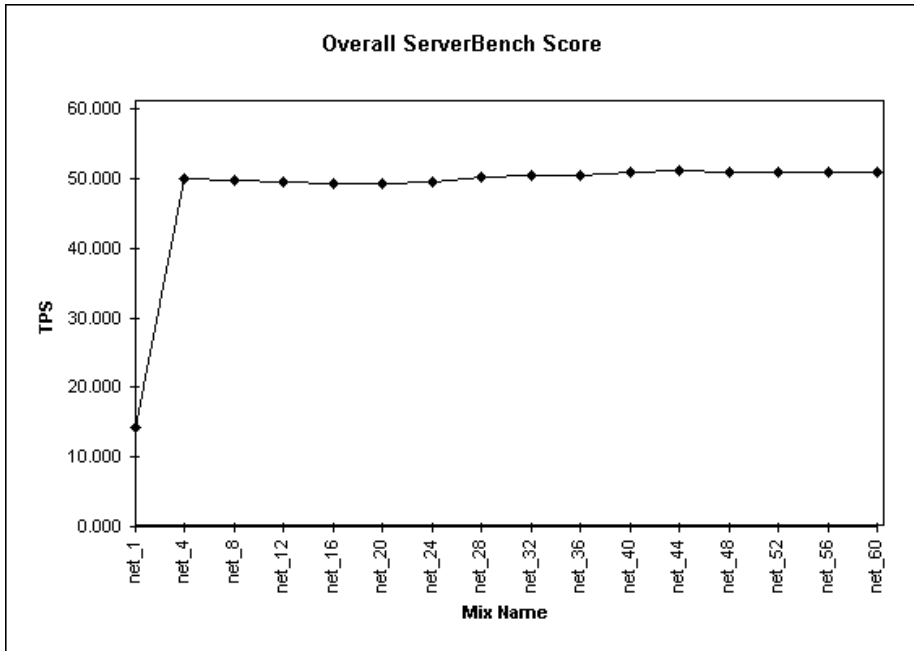
The main information in Table 1: ServerBench Summary is the overall score. This is the score for your server. ServerBench calculates this score by applying a weighted harmonic mean to the numbers in the Total Transaction Score column in Table 2. The number in parentheses in the Total Transaction Score column indicates the weight that transaction had. (This is the same number that you'll see in the Transaction Iteration column in Table 7.)

ServerBench also displays information some test information. This includes comments about the test run as well as the time when the test began and ended.

Figure 12-3: Table 1

<i>Table 1: ServerBench Summary</i>		
<i>c:\sb30\suites\n_60.tst</i>		
<i>Mix Name</i>	<i>Harmonic mean of Total TPS Scores</i>	<i>Test Information</i>
net_1	14.263	ServerBench 3.0
net_4	49.866	Start Suite: Mon Feb 12 13:14:00 1996
net_8	49.736	Finish Suite: Mon Feb 12 14:34:51 1996
net_12	49.507	Comments: 1 CPU network test
net_16	49.303	
net_20	49.263	
net_24	49.513	
net_28	50.151	
net_32	50.348	
net_36	50.510	
net_40	50.910	
net_44	51.026	
net_48	50.976	
net_52	50.828	
net_56	50.881	
net_60	50.904	

ServerBench also displays a graph that plots the TPS scores against the name of each mix.

Figure 12-4: The graph associated with Table 1

Overall Data

In the Overall ServerBench Data report, ServerBench presents Table 2. This table gives you an overview of each mix in the test suite. Because Table 2 gives you a general perspective on how ServerBench calculated your server's overall score, we include that score in this table as well.

Table 2

This table provides overall data about the mixes.

Figure 12-5: Table 2: Overall ServerBench Data

<i>Table 2: Overall ServerBench Data c:\sb30\suites'n_60.tst</i>								
<i>Mix Name</i>	<i>Clients Participating</i>	<i>Clients Starved Out</i>	<i>Elapsed Time (seconds)</i>	<i>Average Ramp Up Iterations</i>	<i>Average Valid Iterations</i>	<i>Average Ramp Down Iterations</i>	<i>Harmonic Mean of Total TPS Scores</i>	<i>Total Transaction Scores (TPS)</i>
net_1	1	0	300	1272.000	2966.000	0.000	14.263	NET (1) 14.263
net_4	4	0	300	1112.500	2223.250	371.000	49.866	NET (1) 49.866
net_8	8	0	300	556.875	1111.375	185.250	49.736	NET (1) 49.736
net_12	12	0	300	369.083	738.000	123.000	49.507	NET (1) 49.507
net_16	16	0	300	275.938	551.062	91.812	49.303	NET (1) 49.303
net_20	20	0	300	220.700	440.800	73.450	49.263	NET (1) 49.263
net_24	24	0	300	185.375	369.042	61.583	49.513	NET (1) 49.513
net_28	28	0	302	160.964	320.143	55.357	50.151	NET (1) 50.151
net_32	32	0	302	142.219	281.281	48.781	50.348	NET (1) 50.348
net_36	36	0	303	127.528	250.528	44.472	50.510	NET (1) 50.510
net_40	40	0	302	115.400	227.300	39.100	50.910	NET (1) 50.910
net_44	44	0	303	105.341	207.205	36.500	51.026	NET (1) 51.026
net_48	48	0	305	96.938	189.917	34.667	50.976	NET (1) 50.976
net_52	52	0	303	89.192	174.865	29.673	50.828	NET (1) 50.828
net_56	56	0	304	83.071	162.554	28.250	50.881	NET (1) 50.881
net_60	60	0	303	78.050	151.550	26.367	50.904	NET (1) 50.904

In this table, you'll find the following information:

- **Mix Name.** This is the name assigned to the mix when it was created.
- **Clients Participating.** This is the number of clients who completed at least one iteration of the mix during the time that ServerBench was recording test results. (ServerBench does not record results for any mix iterations that begin during the Ramp up or Ramp down periods.) Normally, this number should be the same as the number of clients included in the test mix.
- **Clients Starved Out.** This is the number of clients that were included in the mix who did not have one completed transaction.
- **Elapsed Time (seconds).** This is the amount of time in seconds that it took the mix to run. The controller generates this number. This number includes the time spent during the Run phase of the mix and does not include any time for initializing or reporting.
- **Average Ramp Up Iterations.** This the average number of iterations that the clients in this mix completed during the ramp up period. To arrive at this

number, ServerBench totals each client’s Ramp up iterations and then divides by the number of clients participating in the mix.

- Average Valid Iterations. This is the average number of valid iterations for all the clients in the mix. Each client counts its valid iterations in order to calculate its TPS scores. A valid iteration is a completed mix iteration that began after the Ramp up period ended and before the Ramp down period started. To arrive at this number, ServerBench totals each client’s valid iterations and then divides by the number of clients participating in the mix.
- Average Ramp Down Iterations. This the average number of iterations that the clients in this mix completed during the ramp up period. To arrive at this number, ServerBench totals each client’s ramp down iterations and then divides by the number of clients participating in the mix.
- Harmonic Mean of Total TPS Score. This is the same score you see in Table 1. It’s the overall score for your server, calculated by applying a weighted harmonic mean to the Total Transaction Score in the next column. The number in parentheses in the Total Transaction Score column indicates the weight that transaction had. (This is the same number that you’ll see in the Transaction Iteration column in Table 7.)
- Total Transaction Score (TPS). This is the total number of transactions completed for that mix. ServerBench doesn’t weight this score. The number in parentheses is the weight that transaction had. (This is the same as the number in the Transactions iteration column in Table 7.)

Client data

Table 3 gives information about how well each client did in the mix. This table contains data about the number of iterations each client performed as well as the scores the clients achieved for each transaction. This level of granularity lets you check to see if your clients are receiving good service.

Figure 12-6: Table 3: Client Data

Mix Name	Client Name	Client ID	Group	Data File Pathname	Ramp Up Iterations	Valid Iterations	Ramp Down Iterations	Client Score (TPS)	Deviation From the Mean	Client Transaction Scores (TPS)
net_1										NET (1)
	c1	1	1		1272	2966	0	14.263	0.000	14.263
	Mean:				1272.000	2966.000	0.000	14.263		
	Standard Deviation:				0.000	0.000	0.000	0.000		
net_4										NET (1)
	c1	1	1		1137	MAX 2136	MIN 362	MIN 11.971	-1.211	MIN 11.971
	c3	3	3		1074	MIN 2288	373	12.839	0.910	12.839
	c4	4	4		1125	2301	MAX 376	MAX 12.901	1.062	MAX 12.901
	c2	2	2		1114	2168	373	12.155	-0.761	12.155
	Mean:				1112.500	2223.250	371.000	12.466		
	Standard Deviation:				23.670	72.289	5.339	0.409		

In Table 3, you'll find:

- **Mix Name.** This is the name you gave the mix.
- **Client name.** This is the unique name for that client. You assign the client name when you install the ServerBench client program on the client. This is the name that appears in the **CLIENT.CFG** file.
- **Client ID.** This is the unique ID number for that client. This is the ID number that appears in the **CLIENT.CFG** file. ServerBench displays the clients in the order in which it included them in the mix.
- **Group.** This is the group number for that client. You assign group numbers in the **CLIENT.CFG** file that you create as part of setting up ServerBench.
- **Data file pathname.** This is the path name to the data file ServerBench created for that client. By default, ServerBench creates these files in the ServerBench directory on your server. In the standard test suites, we always specify a simple path name, such as **data1**, **data2**, and so forth. If you use different names, make sure the directory you specify exists before you run a disk test.
- **Ramp Up Iterations.** This is the number of completed mix iterations that began during the Ramp up period. ServerBench also places **MAX** next to the client reporting the maximum number of Ramp up iterations for that mix and **MIN** next to the client reporting the minimum number of iterations.
- **Valid Iterations.** This is the number of completed mix iterations that each client counts toward its TPS scores. A valid iteration is one that begins after the Ramp up period ends and before the Ramp down period begins. This column is the first place you should check if you're concerned that your clients are getting uneven service. If there is a wide variation in the number of iterations completed between the different clients, then they probably aren't getting even service. One way you can smooth out service is to increase the Ramp up time. Keep in mind, though, that other factors, including the speed of the client, can influence the number of iterations a client completes. ServerBench also places the client reporting **MAX** next to the maximum number of valid iterations for that mix and **MIN** next to the client reporting the minimum number of iterations.
- **Ramp down Iterations.** This is the number of completed mix iterations that began during the Ramp down period. ServerBench also places **MAX** next to the client reporting the maximum number of Ramp down iterations for that mix and **MIN** next to the client reporting the minimum number of iterations.
- **Client Score (TPS).** This is the individual ServerBench score for that client. ServerBench takes the raw TPS scores for the client (shown in the column Client Transaction Scores) and uses a weighted harmonic mean to calculate the

individual client ServerBench score. (ServerBench lists the weights for each transaction in Table 2. The weights are the numbers in parentheses in the Total Transaction Score column.)

- **Deviation from the Mean.** This is the distance that the client's overall score is from the mean score for that mix. You measure deviation from the mean by calculating the number of standard deviations that separate that client's score from the mean for the mix. A standard deviation of 0 to 1 is very good; 1 to 2 is OK, 2 to 3 means things are not as good as they should be, and 3 or more indicates a problem. (Mean is an arithmetic mean.) ServerBench places MAX next to the client with the largest standard deviation from the mean for that mix and MIN next to the client with the smallest deviation from the mean. (Because this is a measure of distance from the mean, it doesn't matter whether this number is positive or negative.)
- **Mean.** This is the average of the individual client ServerBench scores for the mix.
- **Standard Deviation.** This is a unit of measurement used to determine how closely the scores are clustered around the mean (i.e., to tell you how the clients scores are dispersed relative to the mean for the mix). Basically, standard deviation is a measure of what's normal. For example, if you look at a bell curve that shows a normal distribution, you'll see that most of the scores fall within one standard deviation from the mean. To calculate a standard deviation, divide the maximum distance from the mean by the mean.
- **Client Transaction Scores (TPS).** This column contains the TPS score for that client for each transaction in the mix. These are the raw TPS scores; i.e., they're simply the number of completed transactions divided by the total amount of time spent completing the transactions. In the column Client Transaction Scores (TPS), ServerBench combines these scores using a harmonic mean to produce a individual client ServerBench score for each client.

The Disclosure report

Tables 4 and 5 give you information about how your server and client test bed are set up. (This is information you'll need if you want to publish your ServerBench results.) ServerBench collects some of this information in the **.DLG** file it creates when it creates its results files. However, it can't collect all the necessary information about your server and clients. If you want the Server and Client Disclosure tables to be complete, you can either edit the **.DLG** file to include all the information or set up a disclosure database snapshot file that contains all the information. While you can only use the **.DLG** file with its associated results file, you can tell ServerBench to use the snapshot file with each set of your results. You

specify which file to use for generating these tables when you're at the View Results dialog box. (See the section "Setting up your disclosure database" later in this chapter for information on how to set up a database snapshot file.)

Table 4

Table 4 provides you with information about your server. We set up this table to help you gather the information the ServerBench License Agreement requires you to include if you want to publish your results. You'll also need to include the information in Table 5, which has the client disclosure information. (See the License Agreement for the details of what you must include any time you publish your ServerBench results.)

Figure 12-7: Table 4

<i>Table 4: Server Disclosure</i>	
Server Component	Description
Operating System:	
Server Name:	scosysv
Operating System:	SCO OpenServer
Version:	5
Additional Info:	Networking Supplement not installed
CPU/Memory:	
Processor Type:	Pentium
Processor Speed:	100 Mhz
Number of CPUs:	4
Memory:	128 MB
Additional Info:	
Network Configuration: 60 nodes over 4 switched Ethernet segments, 4 NE3200 NICs	
Disk Subsystem: Compaq SmartArray controller, 5 drives arranged as 1 OS + 4 work drives, no striping in effect	
Comments:	

NOTE: For illustration purposes, we've created an example of Table 4, which we show in Figure 12-7. This is not an actual disclosure sheet.

Table 4 contains the Server Disclosure information. We've broken it down into four sections to make it easier for you to follow. The first section deals with your server operating system and contains the following information:

- The server name. This is the name you supply for the server.
- Operating system. This is the operating system you use on your server; for example, NetWare, OS/2 Warp Server, SCO OpenServer, SCO UnixWare, or Windows NT

- Version. This is the version number of your server operating system.
- Additional Info. You can add any other information here that you find useful.

The second section focuses on your server's processor subsystem (i.e., CPU/Memory) and has information on:

- Processor Type. This is type processor your server is using, such as Pentium®.
- Processor Speed. This is how fast your processor is; for example, 90 MHz.
- Number of CPUs. This is how many processors you're using.
- Memory. This is how much memory your system has configured.
- Additional Info. You can add any other information here that you find useful, such as adding a note that you're using a multiprocessor configuration.

The third section is your Network configuration section. Use this section to identify the network cards and drivers your server is using.

The final section is your Disk Subsystem configuration section. You might enter something like you're using disk mirroring, what your server's disk drive type is, what the disk controller type is, how many drives your server's using, and what the disk drive name and version is.

At the end of the table there's a place where you can enter any general comments you want to.

The information in Table 5

Table 5 has the Client Disclosure information. This table contains information on each client in the test. You'll also need to include the information in Table 4, which has the server disclosure information if you want to publish your results. (See the License Agreement for the details of what you must include any time you publish your ServerBench results.)

Figure 12-8: Table 5

Client Name	Client ID	Processor	Operating System	Windows System	Memory (MB)	32-Bit File Access	File Cache Size	32-Bit Disk Access	Network Card	Protocol	Protocol Version
c1	1	Intel Pentium(R) Step 11 Features 1b/h, 133 MHz, mc-Oe-Chip	MS-DOS 7.00	Windows	16MB	off	zero	off	Unknown	TCP/IP	Microsoft Windows Sockets Version 1.1
c2	2	Intel Pentium(R) Step 11 Features 1b/h, 133 MHz, mc-Oe-Chip	MS-DOS 7.00	Windows	16MB	off	zero	off	Unknown	TCP/IP	Microsoft Windows Sockets Version 1.1
c3	3	Intel Pentium(R) Step 11 Features 1b/h, 133 MHz, mc-Oe-Chip	MS-DOS 7.00	Windows	16MB	off	zero	off	Unknown	TCP/IP	Microsoft Windows Sockets Version 1.1
c4	4	Intel Pentium(R) Step 11 Features 1b/h, 133 MHz, mc-Oe-Chip	MS-DOS 7.00	Windows	16MB	off	zero	off	Unknown	TCP/IP	Microsoft Windows Sockets Version 1.1
c5	5	Intel Pentium(R) Step 11 Features 1b/h, 133 MHz, mc-Oe-Chip	MS-DOS 7.00	Windows	16MB	off	zero	off	Unknown	TCP/IP	Microsoft Windows Sockets Version 1.1
c6	6	Intel Pentium(R) Step 11 Features 1b/h, 133 MHz, mc-Oe-Chip	MS-DOS 7.00	Windows	16MB	off	zero	off	Unknown	TCP/IP	Microsoft Windows Sockets Version 1.1
c7	7	Intel Pentium(R) Step 11 Features 1b/h, 133 MHz, mc-Oe-Chip	MS-DOS 7.00	Windows	16MB	off	zero	off	Unknown	TCP/IP	Microsoft Windows Sockets Version 1.1
c8	8	Intel Pentium(R) Step 11 Features 1b/h, 133 MHz, mc-Oe-Chip	MS-DOS 7.00	Windows	16MB	off	zero	off	Unknown	TCP/IP	Microsoft Windows Sockets Version 1.1
c9	9	Intel Pentium(R) Step 11 Features 1b/h, 133 MHz, mc-Oe-Chip	MS-DOS 7.00	Windows	16MB	off	zero	off	Unknown	TCP/IP	Microsoft Windows Sockets Version 1.1
c10	10	Intel Pentium(R) Step 11 Features 1b/h, 133 MHz, mc-Oe-Chip	MS-DOS 7.00	Windows	16MB	off	zero	off	Unknown	TCP/IP	Microsoft Windows Sockets Version 1.1

Table 5 tells you:

- Client Name. This is the unique name you gave that client in the client configuration file (**CLIENT.CFG**).
- Client ID. This is the unique identification number you gave that client in the client configuration file (**CLIENT.CFG**).
- Processor. Here you'll find information about each client's processor. Because your clients don't all have to be same, this information can vary from client to client.
- Operating system. This column contains the name of the client's operating system, such as MS-DOS, and its version number.
- Windows System. This is the name and version number of the client's Windows system; for example, Windows 95.
- Memory (MB). This is the amount of memory the client has.
- 32-Bit File Access. You can tell whether a client is using 32-bit file access. If the client is using it, the word "on" appears in this column. If the client isn't using it, you'll see "off" here.
- File Cache Size. This column tells you what size the client's file cache is, if anything.
- 32-Bit Disk Access. This column tells you whether 32-bit disk access is on or off.
- Network Card. This tells you what type of network card the client is using. You'll need to supply this information; ServerBench can't capture it.
- Protocol. This is the type of network protocol the client is using.
- Protocol version. This is the version number for the network protocol the client is using.

The Suite Definition reports

The two tables in this report provide you with details on the test suite you ran and the transactions it included. By looking at these two tables, you'll get a perspective on the work your server did in this test suite.

Table 6

Table 6 gives you information about each mix that you ran. This table gives you information about each mix. These are the mix parameters you set when you're at the Mix Definition window.

This table uses a separate line to present information on each mix.

Figure 12-9: Table 6

Mix Name	#Clients	Groups	Think (seconds)	Delay (seconds)	Transactions per Mix	Tests per Mix	Initial Size (bytes)	I/O Range (bytes)	Ramp Up	Ramp Down	Length
net_1	1		0.000	0.000	1	5	1048576	1048576	90.000	30.000	300.000
net_4	4		0.000	0.000	1	5	1048576	1048576	90.000	30.000	300.000
net_8	8		0.000	0.000	1	5	1048576	1048576	90.000	30.000	300.000
net_12	12		0.000	0.000	1	5	1048576	1048576	90.000	30.000	300.000
net_16	16		0.000	0.000	1	5	1048576	1048576	90.000	30.000	300.000
net_20	20		0.000	0.000	1	5	1048576	1048576	90.000	30.000	300.000
net_24	24		0.000	0.000	1	5	1048576	1048576	90.000	30.000	300.000
net_28	28		0.000	0.000	1	5	1048576	1048576	90.000	30.000	300.000
net_32	32		0.000	0.000	1	5	1048576	1048576	90.000	30.000	300.000
net_36	36		0.000	0.000	1	5	1048576	1048576	90.000	30.000	300.000
net_40	40		0.000	0.000	1	5	1048576	1048576	90.000	30.000	300.000
net_44	44		0.000	0.000	1	5	1048576	1048576	90.000	30.000	300.000
net_48	48		0.000	0.000	1	5	1048576	1048576	90.000	30.000	300.000
net_52	52		0.000	0.000	1	5	1048576	1048576	90.000	30.000	300.000
net_56	56		0.000	0.000	1	5	1048576	1048576	90.000	30.000	300.000
net_60	60		0.000	0.000	1	5	1048576	1048576	90.000	30.000	300.000

In this table, you'll find the following information

- **Mix Name.** This is the name assigned to the mix when it was created.
- **#Clients.** This is the maximum number of clients you want to run the mix. ServerBench will run the mix with fewer clients if necessary (for example, if you don't have that many clients connected to the server.)
- **Groups.** This is the list of group numbers selected for this mix. The ServerBench standard test suites do not select any groups. This way all the connected clients can run in the mix.
- **Think time.** This is the amount of time in seconds (or fractions of seconds) that a client waits once the server has replied to its request before it sends another request to the server. In general, the smaller the value for Think time, the more the test stresses the server. The ServerBench standard test

suites use a Think time of 0 seconds. For more information on Think time, see Chapter 10 “Creating Your Own Test Suites.”

- **Delay time.** This is the amount of time in seconds (or fractions of seconds) that the client waits to start the mix once the controller says all the clients can begin. By setting Delay time you can stagger the clients' initial requests to the server instead of overloading the server with a burst of requests at the beginning of a mix. The ServerBench standard test suites use a Delay time of 0 seconds. For more information on Delay time, see Chapter 10 “Creating Your Own Test Suites.”
- **Transactions per mix.** This is the total number of transactions a client requests during one iteration of the mix (i.e., it's the sum of the number of iterations for each transaction in the mix). Suppose you set the mix up to perform 6 iterations of TRANS1, 1 iteration of TRANS2, and 8 iterations of TRANS3. The number you'd see in this column would be 15 (6 + 1 + 8).
- **Tests per mix.** This is the total number of tests the server executes during one iteration of the mix.
- **Initial Size (bytes).** This is the size in bytes of the data file used by the disk tests when the test began. If you are creating a mix, this is the test value you supply for the Disk Test File Initial Size parameter. When you run the Append test, your final size will differ from your initial size.
- **I/O Range (bytes).** This is the number of contiguous bytes in the test data files that ServerBench used for input/output operations during a disk test.
- **Ramp up.** This is a fixed number of seconds at the beginning of the mix. ServerBench does not record any results from mix iterations that begin during the Ramp up period. This way your test results aren't skewed because the server load for the first few clients was very light, thus resulting in an artificially high score. (For more information on Ramp up, see Chapter 10 “Creating Your Own Test Suites.”)
- **Ramp down.** This is a fixed number of seconds at the end of the mix. ServerBench does not record any results from mix iterations that begin during the Ramp down period. This way your test results aren't skewed because the server load for the last few clients was very light, thus resulting in an artificially high score. (For more information on Ramp down, see Chapter 10 “Creating Your Own Test Suites.”)
- **Length.** This is the number of seconds ServerBench spends executing the mix. Because the mix will continue to run until each client finishes its current iteration, the value you see here is actually the minimum amount of time that the mix ran.(For more information on Length, see Chapter 10 “Creating Your Own Test Suites.”)

Table 7

Table 7 provides you with the details on the transactions in each mix. This table uses a separate line in the table for each test within each transaction.

Figure 12-10: Table 7

Mix Name	Transaction Iterations	Transaction Name	Test Iterations	Test Type	Request Size	Total Size
net_1	1	NET	5	Server to Client	10240	10240
net_4	1	NET	5	Server to Client	10240	10240
net_8	1	NET	5	Server to Client	10240	10240
net_12	1	NET	5	Server to Client	10240	10240
net_16	1	NET	5	Server to Client	10240	10240
net_20	1	NET	5	Server to Client	10240	10240
net_24	1	NET	5	Server to Client	10240	10240
net_28	1	NET	5	Server to Client	10240	10240
net_32	1	NET	5	Server to Client	10240	10240
net_36	1	NET	5	Server to Client	10240	10240
net_40	1	NET	5	Server to Client	10240	10240
net_44	1	NET	5	Server to Client	10240	10240
net_48	1	NET	5	Server to Client	10240	10240
net_52	1	NET	5	Server to Client	10240	10240
net_56	1	NET	5	Server to Client	10240	10240
net_60	1	NET	5	Server to Client	10240	10240

In this table you'll find:

- **Mix Name.** This is the name assigned to the mix when it was created.
- **Transaction Iterations.** During one iteration of the mix, a client may request a transaction more than once. This number tells you how many times a client requested this transaction during one iteration. The more times a client requests a transaction during a single iteration, the higher the weight of that transaction. Thus, this value determines the "weight" of the transaction in the weighted harmonic mean that ServerBench uses to compute the overall score.
- **Transaction Name.** This is the name you assigned to this transaction in the Mix Definition window when you created it. We recommend that you use descriptive names when you set up transactions because a single transaction can contain more than one kind of test.
- **Test Iterations.** This is the number of times a test is repeated each time a client requests that transaction. Suppose Transaction1 contains one Sequential Read test with a test iteration of 6 and 1 Sequential Write test with a test iteration of 2. During one iteration of Transaction1, the server would

perform a total of eight tests; it would execute the Sequential Read test six times and the Sequential Write test two times.

- **Test Type.** This column tells you which individual tests are in that transaction. You can create transactions using ServerBench's Processor, Sequential Read, Sequential Write, Random Read, Random Write, Append, Server to Client, and Client to Server tests.
- **Request size.** This is the amount of data ServerBench passes to your server's operating system at one time (i.e., this is the value the ServerBench application on the server uses for the size parameter when it requests file I/O or network I/O). For example, if you select a disk random read test with an Request Size of 512 bytes, ServerBench will read data from your server's disk in increments of 512 bytes. You'll see a Request Size for every test except the Processor Test. (See the section "How the Request Size parameter affects your results" in Chapter 10 "Creating Your Own Test Suites" for more information.)
- **Total size.** For a Processor test, this is the total number of iterations the test performed. For the other tests, this is the total amount of data that you want ServerBench to move for that test. For example, if your Total Size for the random read disk test is 51200 bytes and your Request Size is 512 bytes, ServerBench would read data from your server's disk in increments of 512 bytes until it had read a total of 51200 bytes of data. (Remember, Total Size is not the same as the Initial Size of the disk test data file. See the section "How the Total Size parameter affects your results" in Chapter 10 "Creating Your Own Test Suites" for more information.)

Creating graphs of multiple results

ServerBench lets you create a graph that contains the TPS results from multiple test runs. This is useful because it allows you to easily compare results from multiple test runs in a single graph. You can also create a variance summary graph that displays the ratio of the mean to the standard deviation for different test runs. If you want to create these graphs, you must always load ServerBench's Add-in module **SVRBENCH.XLA**. You can tell ServerBench to load this module when you're at the View Results window.

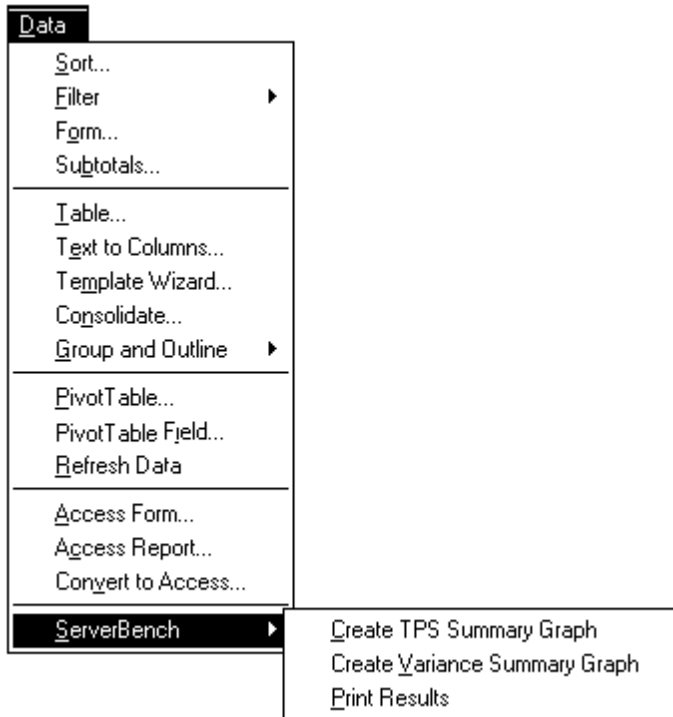


Tip:

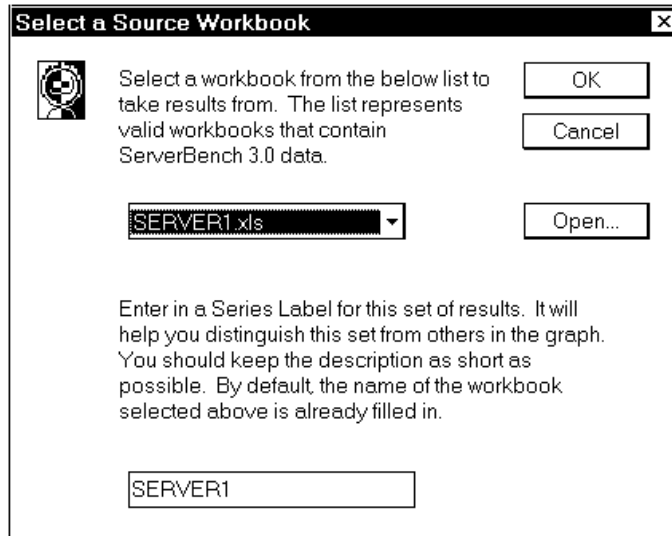
We recommend that, if you create a TPS graph or a variance graph, you use results from different runs of the same test suite.

Here's how you create these graphs:

1. Choose several results files and click on View at the View Results dialog box.. (Remember to also choose the option Load **svrbench.xls** Add-in module when you're at this dialog box.)
2. Once the workbook is open in Excel, choose the Data menu.
3. Choose the option ServerBench.

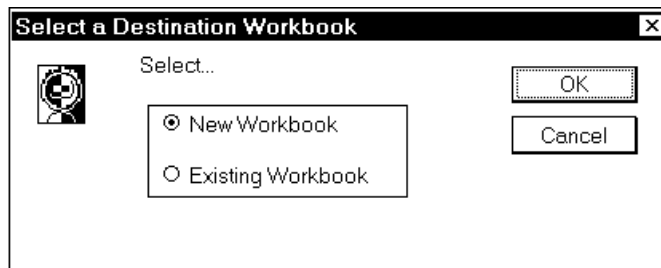


4. Choose either the option Create TPS Summary Graph or Create Variance Summary graph.
5. When the Select a Source Workbook dialog box appears, use the pull-down list to choose one of the current workbooks. (If the active workbook is valid, ServerBench by default marks it as selected; however, you can easily choose a different workbook.)

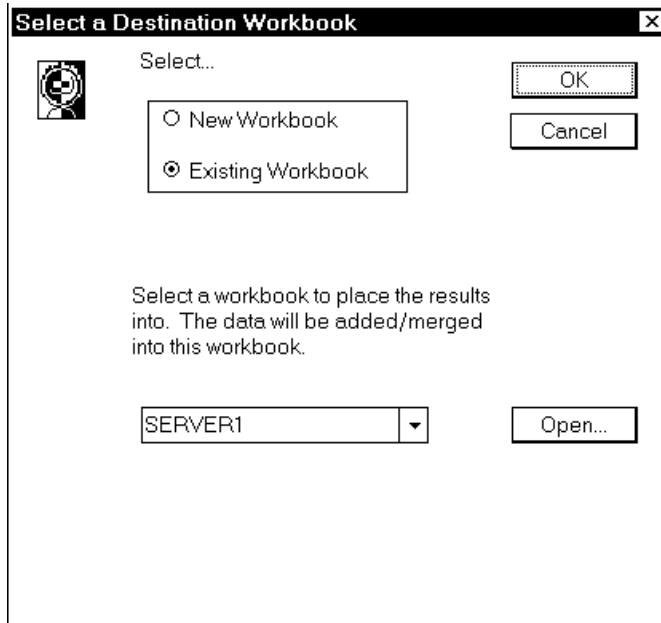


By default, ServerBench selects the active workbook (assuming it's a valid ServerBench workbook and contains data you can use).

6. Enter a label in the comment box to describe this set of results and click on OK.
7. When the Select a Destination Workbook dialog box appears, choose New if this is the first workbook you've specified for the graph and click on OK. (You could place the graph in your current workbook, but we recommend creating a new workbook.)



8. To add other results to the graph, repeat steps 2 through 6. Then choose Existing Workbook when the Select a Destination Workbook dialog box appears and highlight the name you've given this workbook.



ServerBench automatically chooses the existing workbook option if a Summary TPS or variance graph is present. If two or more of these graphs of the same type (i.e., either TPS or variance) are present, ServerBench chooses the first one.

9. Click on OK.

Using your disclosure database

Unless you specify differently, ServerBench creates Tables 4 and 5 based on information it captures and stores in the **.DLG** log file. While this is useful, it has the drawback that ServerBench can't capture all the necessary information about your server and clients. So, if you want all that information in the tables, you must supply it.

To make it easier to display information about your server and clients, we created a special ServerBench database file. This file, which is based on the **.DLG** file, contains all the information the License Agreement says you need to supply if you want to publish your results. And by creating this special database file, you only need to enter information ServerBench can't capture (such as the client and server network card and so on) and then you can re-use the information in this database as often as you like.

The first time you run a test suite after installing ServerBench, you'll see a grayed-out path name in the Select Reports dialog box. This is the path name ServerBench proposes for the disclosure file. If you agree to the path name, ServerBench will create this file (see the section "Creating a new database snapshot file" for more information). By default, ServerBench calls the file **SVRBENCH.SNP** and places it in the installation directory on the controller.

Because ServerBench takes a "snapshot" of the hardware and software on your server and clients when it creates this database file, we refer to the file as a *snapshot* file and use an extension of **.SNP** for it.

There's one important restriction about the database file, though. Once ServerBench creates the file, you *cannot* add more clients to it. So the first test suite you run should use the most clients you plan to use for any test suite.

NOTE: Actually, this is not a big problem, because you can easily create a new database file whenever you need to.

The big advantage to setting up an initial database with all your clients in it is that you'll only need to enter the hardware and software information ServerBench couldn't capture once.

Creating your first database snapshot file

Here's how we recommend you create your first database snapshot file:

1. Start ServerBench on the controller, the server, and the maximum number of clients that you plan to use as part of your ServerBench test bed.
2. In the controller window, select the drop-down menu Disclosure and make sure the option Generate Client info for each suite is checked (ServerBench checks this option by default).

<u>D</u> isclosure
Get Client <u>I</u> nformation
√ <u>G</u> enerate client info each suite

NOTE: We recommend that you always have this option checked when you run a ServerBench test suite. If it's unchecked, the **.DLG** file ServerBench creates won't contain any information on the clients. In addition, ServerBench won't create Table 5 in the results spreadsheets. If you never change this option, you don't need to verify this every time you start a test.

3. Select and run a short test suite that uses all the clients you have connected.



Tip:

We recommend you run **SAMPLE.TST** with the maximum number of clients you plan to use with ServerBench. For example, if you plan to use ServerBench's standard test suites, you'll need to have 60 clients connected to the server. **SAMPLE.TST** takes about two minutes to run and sets up a database for 60 clients.

4. Go to the main ServerBench window and choose View Results.
5. At the Select Results dialog box, choose the test suite you just ran and click on OK. ServerBench displays the View Results dialog box.

View Results - d:\sb30\nt351\suites\sys_4b.tst

Results Name

c:\sb30\nt351\results\sys_4a.tlg
 c:\sb30\nt351\results\sys_4b.tlg

Comments

Use Snapshot : c:\sb30\nt351\svrbench.snp

Workbook Options

Save workbook with results name
 Delete log files
 Load svrbench.xla Add-In

Worksheets

Overall Results
 Client Results
 Disclosure
 Test Suite Description

Because the snapshot file does not exist, ServerBench displays the default path name to the snapshot file in the Use Snapshot box, but it has grayed out this options.

NOTE: The Current Snapshot text may appear to be missing due to the color of your window text or the application background color. If you wish

to see the grayed path, change your system colors through the Windows Control Panel to the Windows default color scheme.

6. Choose Edit Snapshot. This tells ServerBench to display the Server Disclosure Form with the information in your current disclosure file. (This is the information contained within your current test suite results.)
7. ServerBench displays the Server Disclosure Form. Enter the correct data for your server. ServerBench automatically saves the information you enter to the current snapshot file. (See the section “Updating server disclosure information” for details on what information you should enter in.)

The ServerBench 3.0 license agreement requires you to disclose a system configuration summary when quoting any ServerBench 3.0 results. This Disclosure form enumerates many attributes of your network that can have a significant effect on test scores. ServerBench 3.0 cannot determine some attributes. It may have determined others incorrectly.

Server Name: ZDBOP TEST

Operating System: OS: Windows NT - Uniprocessor Free, Version: 3.51 build 1057

Network Configuration: [1] Novell NE3200 EISA Adapter, Microsoft, Novell 3200 Adapter Driver 3.51, test

Disk Subsystem: SCSI DiskPeripheral: COMPAQ M2622F-512 1506,

CPU / Memory: Processor: 80486-D0, Speed: [] # CPUs: 1, Memory: 32 MB

Additional Information: AT/AT COMPATIBLE, System BIOS date: 09/22/92, PC Compatible Eisa/Isa HAL

Additional Comments: []

Update snapshot: c:\sb30\nt351\svrbench.snp

Clients...

NOTE: If you choose Edit Disclosure, ServerBench lets you edit the **.DLG** file. If you then want to save this information to the snapshot file, you'll need to Click on the box next to Update snapshot... to the current snapshot and current test suite" so that a check mark appears in that box. This saves the information to your currently selected database snapshot file. If you don't check this box, then ServerBench will save the information to the current test suite results (i.e., the **.DLG** file), but it will not update your database snapshot file.

9. Choose the Clients... button.

10. ServerBench displays the Client Disclosure Form. Enter the correct data for your clients. (ServerBench will have already filled in some of this data for you.) ServerBench automatically saves the information you enter to the current snapshot file.

The ServerBench 3.0 license agreement requires you to disclose a system configuration summary when quoting any ServerBench 3.0 results. This Disclosure form enumerates many attributes of your network that can have a significant effect on test scores. ServerBench 3.0 cannot determine some attributes. It may have determined others incorrectly.

OK Cancel Help

<Template> c1 c2 c3 c4	Select All Clients <input type="text"/> <input type="text"/>	Memory (MB): <input type="text" value="16MB"/>
	ID: <input type="text"/> Name: <input type="text"/>	32-bit File Access: <input type="text" value="off for"/>
	Processor Type: <input type="text" value="486DX, 486DX2, 486DX4 or 487SX, 67 MHz, mc:80487"/>	File Cache Size: <input type="text" value="zero"/>
	Operating System: <input type="text" value="MS-DOS 7.00"/>	32-bit Disk Access: <input type="text" value="off"/>
	Windows System: <input type="text" value="Microsoft Windows 3.95"/>	Network Card: <input type="text" value="Unknown"/>
<input type="checkbox"/> Update snapshot: c:\sb30\nt351\svrbench.snp		Protocol: <input type="text" value="TCP/IP"/>
		Version: <input type="text" value="Microsoft Windows Sockets Version 1.1."/>

See the section "Updating client disclosure information" for details on how you can edit the client information.

NOTE: If you choose Edit Disclosure, ServerBench lets you edit the **.DLG** file. If you then want to save this information to the snapshot file, you'll need to Click on the box next to Update snapshot... so that a check mark appears in that box. This saves the information to your currently selected database snapshot file. If you don't check this box, then ServerBench will save the information to the current test suite results (i.e., the **.DLG** file), but it will not update your database snapshot file.

12. Click on OK in the Client Disclosure window. ServerBench returns you to the Server Disclosure window.
13. Click on OK in the Server Disclosure window. ServerBench returns you to the View Reports dialog box. You can either view your results by choosing OK in the Select Reports dialog box or return to the main ServerBench window by choosing Cancel. You can now use your newly created snapshot database file in subsequent testing (See the section "Using an existing database snapshot file").

Updating server disclosure information

The basic process for editing the server disclosure information is very simple. Simply tab to or click with the mouse on the fields and type in the appropriate information. ServerBench will have already filled in some of this data for you. The amount of information ServerBench provides depends on the port. In addition, you can get a detailed example of the information you need to enter by re-reading the License Agreement.

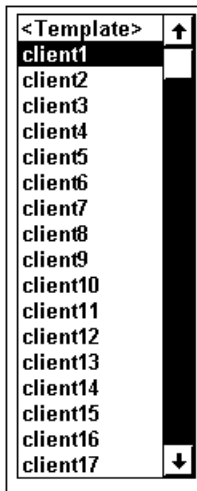
One key field that you can use for each test run is the additional comments field. You can type in any comments or additional information that was specific or different about the run that you feel is worth mentioning.

The check box Update snapshot... information... and the button Clients... are explained in the section “Creating your first database snapshot file”.

When you finish entering information, press OK to return to the Select Reports dialog box.

Updating client disclosure information

You have several options for updating the client disclosure information. The key feature in this window is the template list. It contains the name of each client that participated in the current test suite. You can select the client (or clients) you want to work with from the template list.



Selecting a client or clients

You can select one client by clicking on that client's name or you can select multiple clients.

To select a range of clients, click on the first client in the range and then press the shift key and click on the last client in the range. ServerBench will highlight all the clients in the range.

To select a random group of clients, click on the first client. Then press the control key and click on the next client. Continue using the control key and clicking until you've selected all the clients you want to select. ServerBench will highlight all the selected clients.

To select all the clients, choose the Select All Clients button. ServerBench highlights all the clients.

Editing the client information

You edit the client information by tabbing to or clicking on the appropriate field and then typing in the information.

When you're updating only one client, ServerBench displays all the information it has for that client. Go to the field or fields and enter your changes. ServerBench keeps each change you make for that client.

If you're updating the information for several clients at once, you'll notice that ServerBench places an asterisk in the Client ID box and enters <Template> in the client name box. ServerBench does not display any information about the clients in the other fields. To update the information for these clients:

1. Enter the information in the correct field. The only field that ServerBench can't capture information for is the Network Card. If you don't change a field, ServerBench keeps whatever information it had for that field.

NOTE: If you enter even a space in one of these fields, ServerBench will overwrite the current information in the field and replace it with whatever you typed. You can also erase your typed information by pressing the Clear Template button.

2. Once you've entered all the information, choose Duplicate. Doing this tells ServerBench to update the fields for all the currently selected clients. (You only have to use the Duplicate button when you're working with multiple clients.)

You can also copy the information from one client to other clients. Here's how you do this:

1. Click on the name of the client that you want to copy information from.
2. If necessary, modify that client's information.

3. Now press the control key and click on the word <Template> at the top of the template list. ServerBench stores the information on that client as the template information.
4. Select the range or group of clients that you want to have the same information.
5. Choose the Duplicate button. ServerBench will copy all the information in the template to each client you have selected.

You can also copy the information from one client to another client. Here's how you do this:

1. Click on the name of the client that you want to copy information from.
2. If necessary, modify that client's information.
3. Now press the control key and click on the word <Template> at the top of the template list. ServerBench stores the information on that client as the template information.
4. Select the client you just copied information from and the client you wish to copy to.
5. Choose the Duplicate button. ServerBench will copy all the information in the template to the two clients you have selected.

Updating or adding clients to a snapshot file

Here's how you can update a current database snapshot file and use it to create a new snapshot file.

For example, suppose the first test suite you run after you've installed ServerBench uses 30 clients, ServerBench captures information on those 30 clients and, when you tell ServerBench to save the information, stores it in the **SVRBENCH.SNP** file. You modify **SVRBENCH.SNP** to add the information ServerBench couldn't capture for your server and your clients. If you later run a test with 60 clients, you can use the information in **SVRBENCH.SNP** to update the database for your server and the original 30 clients with the information you've already supplied. Then you use that information as a template to fill in the data for the next 30 clients. You can then use your newly updated test suite results as a base for creating a new snapshot database file containing your "revised and expanded" network.

Here's what you can do:

1. Run a test suite with 60 participating clients.
2. Select the View Results button and choose that results file from the Select Results dialog box.
3. At the View Results dialog box, choose Use Snapshot. This tells ServerBench to use the information in the current database snapshot file (for this example, **SVRBENCH.SNP**) when it displays the server and client disclosure forms.
4. Choose Edit Snapshot. ServerBench displays the Server Disclosure Form with the information in it that was in the database snapshot file. Update the server information if necessary.
5. Choose the Clients button to go to the Clients Disclosure Form.
6. Update the client information. Remember, ServerBench will update the current test suite results information with the information in **SVRBENCH.SNP**. However, since the snapshot file only contains information for 30 clients, ServerBench will only update the information for clients 1 through 30.
7. If clients 31 through 60 use the same information as clients 1 through 30, you can copy the that information to them. (See the section "Editing the Client Information" for details on how to do that.)
8. Click on OK. When you choose OK, ServerBench saves all the information in the Client Disclosure form to the current test suite results and returns you to the Server Disclosure Form.
9. Click on OK in the Server Disclosure Form. ServerBench saves any changes you made to this form in current test suite results and returns you to the View Results dialog box.

10. You are now ready to create a new snapshot database file that will contain all of your updated changes to your network information. (See the next section “Creating a new snapshot database file”).

Creating a new snapshot database file

This assumes you have a results file.

Note: We recommend that the test suite contain filled out information for both the server and the clients. For details on how to enter in this information, see the sections “Updating server disclosure information” and “Updating client disclosure information”).

1. Choose the Select Snapshot button in the View Results dialog box.
2. In Select or Create a Snapshot Template dialog box, enter name you want your snapshot file to have. Remember, it must have a **.SNP** extension. Choose Yes when ServerBench asks if you want to create the snapshot file.
3. At the View Results dialog box, choose Edit Disclosure. ServerBench will display the Server Disclosure Form using information in the **.DLG** file. Edit the information in both the Server and Client Disclosure forms
4. Click on the Update Snapshot... box at the bottom of the Disclosure form.

NOTE: When you create a snapshot file, if you choose the Update Snapshot ... option in either the server window or the client window, ServerBench saves both the server and client information to the new file. This is the only time you don't have to click on both buttons to save all the information to the snapshot file.

ServerBench returns you to the View Reports dialog box. You'll notice that the path name to your new snapshot file and the Use button are both active (i.e., they're not grayed out.)

Your new snapshot file will also appear as the Current Snapshot the next time you're at the Select Reports dialog box. ServerBench always displays the snapshot file you selected last as the Current Snapshot after you press OK in the Select Reports dialog box.

You can also use these steps to create additional snapshot files based on your selected test suite.

End of chapter

Chapter 13

Figuring Out What Your Results Mean

This chapter tries to help you understand what the TPS scores you see actually mean about your server's performance. In this chapter you'll find information on:

- How ServerBench records results.
- Generating a graph to use in evaluating your server's performance.
- What the numbers mean.
- How to look for bottlenecks.
- How to compare your server with other servers.
- What to look for in the standard deviation in scores.
- What's the best way to compare results on different machines.

Looking for the main score

The score that interests most people is the overall ServerBench score for your server. You'll find this score in Table 4. ServerBench uses a harmonic mean to calculate an overall TPS score for your server for each mix.

ServerBench also produces scores for individual clients.

Keep in mind as you check your server's scores that you need to evaluate them in terms of the other data the results tables provide. See the section "Figuring out what your scores mean" later in this chapter for more information.

The measure of a ServerBench result

ServerBench measures its results in terms of transactions per second (TPS). When we talk about TPS, we're defining a completed transaction, which consists of three pieces:

1. A client sends a request to the server asking for some work.
2. The server performs the work.
3. The server sends a response to the client indicating the results of the work.

Because ServerBench only measures completed transactions, ServerBench uses this entire unit as the basis for measuring performance and calculating results.

When you create a transaction, you bundle one or more ServerBench tests together. These tests are Processor, Sequential Read, Sequential Write, Random Read, Random Write, Append, Client to Server, and Server to Client. Then you bundle the transactions together to create a mix.

Keeping track of ServerBench's results

ServerBench calculates the overall server scores for mixes based on the information each client supplies. The individual clients are the ones who actually collect timing information and keep track of how many transactions occur.

When you run a test suite, ServerBench tells the clients which transactions are part of which mix and which tests they contain. Each client takes the mix information, reorders the transactions (this way not all the clients are issuing the same requests at the same time), and then starts sending requests to the server. Each request specifies a transaction for the server to execute. The server executes all the tests in the transaction and then sends a reply back to the client.

NOTE: By reordering the transactions, the clients provide the server with a variety of requests that more closely resembles how users in a real world environment use the server.

Each client measures how long its transactions take to complete. When a client issues a transaction request, it records that time as the transaction start time. Then, when the client gets a response back from the server indicating the work in the transaction is done, the client records the transaction stop time. To calculate how long the transaction took, the client subtracts the start time from the stop time. The client uses this timing information when it computes its TPS scores.

In addition to timing the transactions, each client also keeps a tally of how many of its transaction requests the server completes. The clients use this information to compute raw client TPS score for each transaction in the mix. With each transaction, the client divides the total number of times the transaction was

executed by the time it took to execute them. This time includes sending the request, waiting in a queue for service, and responding to the request.

Once the mix finishes, the clients send their results data to the server. ServerBench then combines the results and calculates the server's overall score as well as other information, such as the client's standard deviation and variance.

Making ServerBench's results count

To give you a reasonable indication of the kind of performance you can expect, we designed ServerBench so you have the option of testing your server when it is in a steady state (i.e., it has an even load). To do this, though, you need to tell ServerBench to ignore any transactions that occur while your server has a light load.

You can do this by setting three mix parameters: Ramp up, Ramp down, and Length. If a mix iteration starts during either Ramp up or Ramp down, the client requesting the transactions ignores any results it gets. The client only records results from transactions where the mix iteration began *after* the Ramp up period and *before* the Ramp down period. This is why, when you use the Ramp up and Ramp down parameters, you need to set a value for your Length parameter that exceeds the sum of the Ramp up and Ramp down parameters.

|<----- Length ----->|

<p>Ramp up; <i>No transactions from mixes starting in this period are recorded</i></p>	<p>Recording test information <i>All transactions from mix iterations beginning in this period are recorded (even if the transaction ends during the Ramp down period)</i></p>	<p>Ramp down; <i>No transactions from mixes starting in this period are recorded</i></p>
--	--	--

Ideally, the result of using Ramp up and Ramp down is that no client records results that occur when the server has a light load. The Ramp up period covers the beginning of the mix and, ideally, ensures that all the clients have started running the mix before they start recording data. The Ramp down period means that at the end of the mix, when some clients have finished sending requests, other clients won't still be recording results. This is because the clients at the beginning and end of the mix would get an unfair advantage; after all, with fewer clients submitting requests, the server can work faster.

However, the clients do record results for mix iterations that begin before the Ramp down period starts, even if those iterations finish during the Ramp down period. We allow these results because your Ramp down period is usually long enough so that other clients will still be sending requests when that client finishes that iteration.

Looking at a graph of server's performance

ServerBench automatically displays a graph that plots your server's overall TPS scores against each mix in the test suite. You'll find this graph in the results spreadsheet under Table 1.

You can use this graph as you evaluate your server's performance. However, If you want this graph to be meaningful, each mix should only change one parameter from the previous mix. For example, the standard test suites consist of mixes that are the same except that the first mix specifies one client; the second mix, four clients; the third mix, eight clients; and so on.

Determining just what your scores mean

A key point to always remember when you look at your ServerBench scores is that you cannot make a judgment based on one piece of data. You need to look at the everything in the tables and think of the relationships between the different pieces of data, not the individual data values.

In general don't worry about the Max client and Min fields for valid iterations in Table 3. Unless you have a serious problem with a client, you can expect clients to trade peak "roles" on each run.

Another good example of related data is how the information you get in Table 3 works with itself. Here's the deal. One column tells you how many iterations each client completes. This is good information. Just don't be lulled into thinking that knowing the number of iterations per client gives you the full picture on how even the service was. For example, suppose Table 3 shows you that all the clients performed between 14 and 16 iterations. This information tells you service for all the clients was pretty good. However, it leaves a lot of room for potential variations, such as whether the worst client just barely completed 14 iterations or was actually one request away from completing 15 iterations. The best client may have barely completed 16 iterations, or it may have been one request away from completing 17. One fraction of a second could have made the difference between a 15 to 16 spread of iterations and a 14 to 17 spread.

So now you can check the individual scores of each client and learn how many standard deviations each client was from the mean. By combining the information in these columns, you can determine how even the service was. The section "Checking for uneven service" explains this process in more detail.

Checking for uneven service

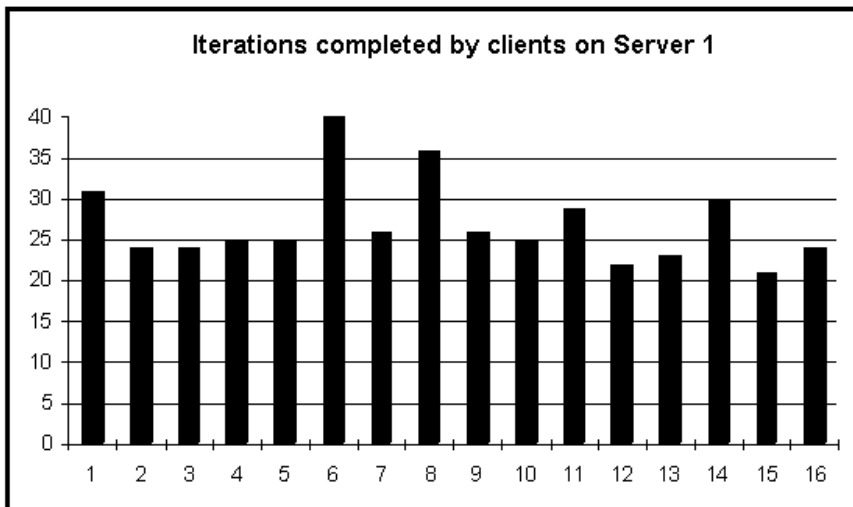
You can't detect uneven service just by looking at the overall score for your server. Uneven service can make your score go up or down or even have absolutely no effect on your score. (However, if one of your users is receiving bad service, you *will* notice the effect of uneven service.)

Consider this example. Suppose almost all of your data fits in cache. In this case, uneven service will actually improve your server's overall score. The reason for the improvement is that uneven service lets some of the clients keep all of their data in cache. Naturally, those clients get an incredible benefit that more than compensates for the clients who aren't getting good service. The other side of the coin is that if one client dominates the processor, it will push the overall performance down. You might see uneven service if you saturate one segment of a LAN or use a UNIX system that isn't properly tuned (see the on-line installation manuals for ServerBench's UNIX platforms for information on tuning your kernel parameters).

The easiest place to start checking for uneven service is in Table 3. Look at the Valid Iterations completed column. With even service, the number of iterations completed for clients within a mix should be very close. If these values vary widely, then you probably have uneven service.

NOTE: Remember, if you're comparing mixes, you can't compare a mix using 48 clients with a mix using 20 clients.

Another sign of uneven service is the presence of "out runners." These are clients who perform much better than average. For example, if you were to plot the actual number of iterations for 16 clients on Server 1, you might see a graph similar to the following:



The average number of iterations with 16 clients was 27. This graph shows that only five clients out of the 16 completed more than 27 iterations. Of those five, only three clients completed more than 30 iterations. This type of curve is typical when you have uneven service. You'll rarely get a smooth curve when the server scheduling is uneven. Instead, you'll tend to see a very unstable results curve.

Another way to determine if you have out runners is to go look at values for the Mean, the Standard Deviation, and the Max (these are also in Table 3).

NOTE: Variations in these numbers will also occur if your clients are different. For example, if your test bed includes both 486 clients and 386 clients, you'll see a difference in the client performance. In this case, a 486 is just faster than an 386. This is not a sign that your server is providing uneven service.

Look at what the mean is relative to the standard deviation. Normally, the standard deviation ought to be a small percentage of the mean. A good standard deviation is probably 3 to 4 percent of the mean. A standard deviation of 10 percent or more is really bad.

The Max value is the largest number of standard deviations that any client's overall score is from the mean. In general, the smaller this value is, the better your server performed. If this value is less than 2 standard deviations from the mean, you're probably in good shape. More than 2 standard deviations and you may have a problem. And start to worry if you have many clients with scores that are more than 3 standard deviations from the mean. You definitely have a problem somewhere. That's because statistically all your clients ought to be within 3 standard deviations of the mean.

Ideally, you want to see even service on a server that is under a heavy load. However, as the load on your server increases, you will notice that the standard deviation gets larger. This is an artifact of server scheduling and resource limits, not ServerBench. When your server is dealing with a heavy load, your clients are less likely to get even service. In some cases, clients are starved out and unable to get service. In other cases, the client did not complete even one iteration in the allotted time. You can tell what kind of service the individual clients are getting by looking at each client's variance. The variance tells you how far from the test mean that client was.

NOTE: If service is really uneven, you should make sure that each network segment has the same relative load.

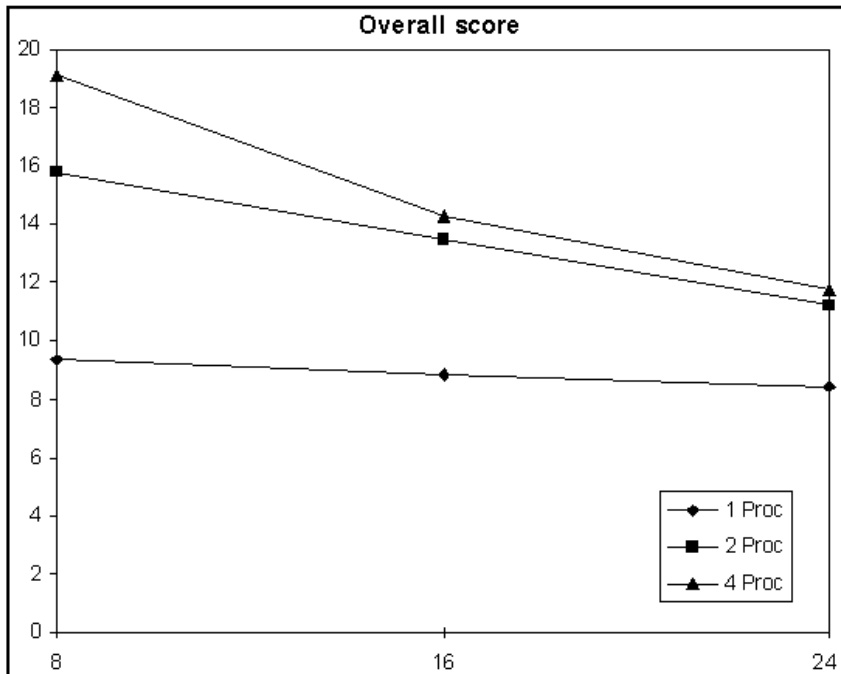
Remember, you need to look at all the data in the spreadsheet, rather than reading too much into a single piece of data. Here are some tips to keep in mind if you're looking for potential service problems:

- Compare the overall ServerBench scores with the average number of iterations. If the overall scores are about the same, but the average number of iterations are very different, you probably have uneven service.
- Look at your results curve. If it's not stable, you may have uneven service.

Checking for bottlenecks

If you want to check for hardware bottlenecks (i.e., areas that are heavily congested), you must run more than just the standard system test suites. The standard suites will give you an indication of bottlenecks; however, the proof of a bottleneck comes when you run the subsystem test suites, because you can isolate parts of your system for comparison. You may find bottlenecks hidden by other bottlenecks.

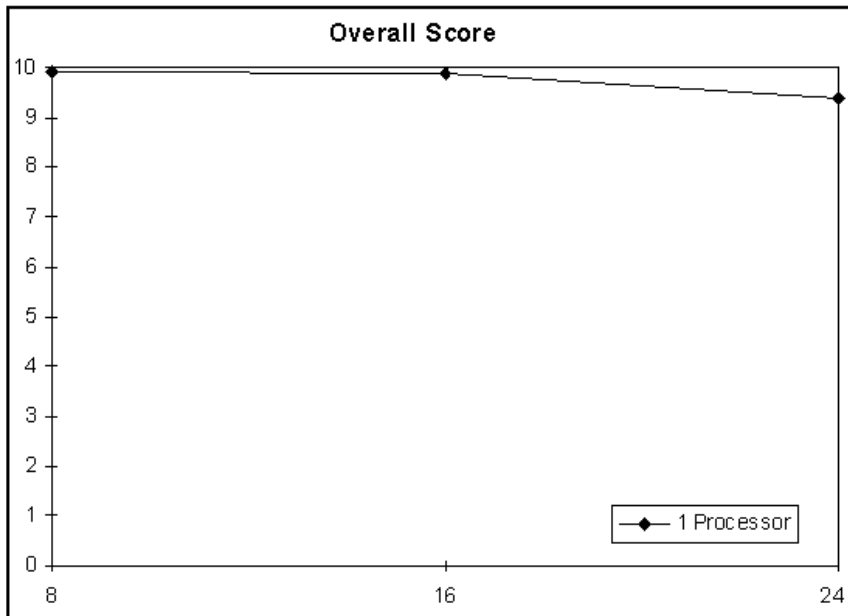
When you find a bottleneck, make sure you don't try to fix it by increasing the wrong resource. Suppose we get a result like this, but we don't do anything to the disk resources. Instead, we add more processors. Look at what happens when we increase the processing power.



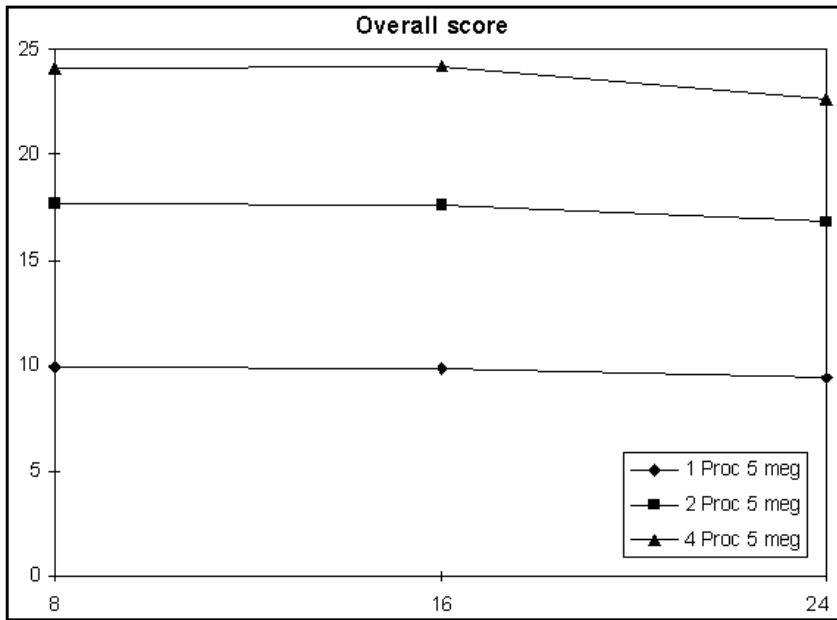
This graph shows how the overall score changes as we go from one processor to four processors for eight, 16, and 24 clients. Obviously, as we increase processing power, the disk becomes even more of a bottleneck. We're boosting the wrong resource so the disk subsystem continues to drag down our scores.

So we assume the disk is the bottleneck. We decrease the size of the test data files the disk tests use from 20 MB to 5 MB and, sure enough, performance goes up.

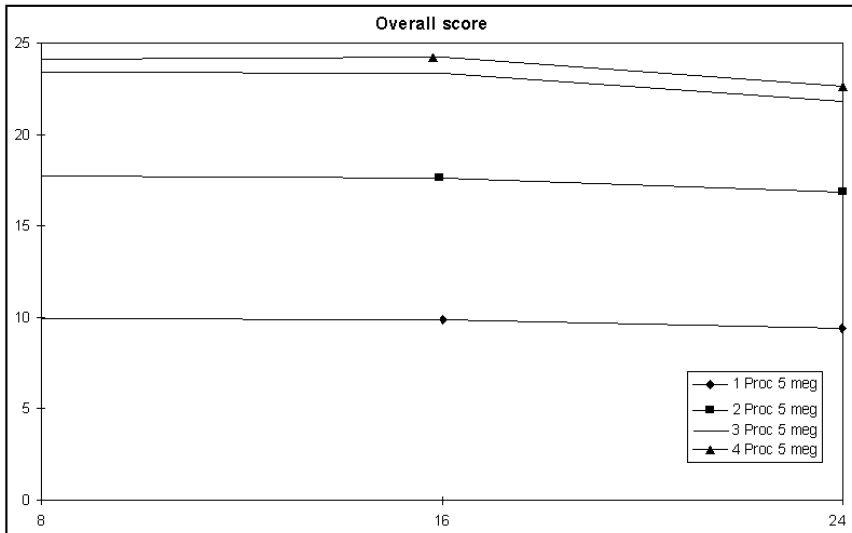
With just one processor, we get a graph that looks like the following one.



Now that we've improved the disk subsystem, let's take another look at what happens when we increase the processing resources.



As you can see, even with the drag on the system caused by the disk subsystem, each additional processor resulted in a substantial improvement in the overall score. While testing systems with 1, 2, and 4 processors is a common practice, we got curious and ran a test suite with 3 processors. The result? We discovered that we had a bottleneck we didn't know about.



When we went from 1 to 2 processors, we saw a 78 percent improvement in our overall score. A 58 percent increase occurred when we went from 2 to 3 processors. However, going from 3 to 4 processors brought about only a 6 percent improvement. This is a little extreme. The most likely culprit here is a bottleneck we hadn't considered: the memory bus is saturated. Probably, if the memory bus had not been a problem, the data point for eight clients would have been much higher.

So, when you start evaluating your system, remember to run multiple tests and keep an open mind. Sometimes one bottleneck hides another bottleneck.

The process for testing your system

The most common bottleneck you'll see on your system will probably be the disk. However, as we showed you in the previous section, there are other bottlenecks you may find if you take a close look. Here's the procedure we recommend you follow:

1. Look at the results you got.
2. Pick a single variable to change (you only want to change one variable at a time).
3. Alter the mix to reflect the change.
4. Run the mix.

Keep performing these steps until you're satisfied you know as much about how your system is performing as you want to know.

How to tell if your results are abnormal

In some cases, you will see that one client has a very high TPS score while the others have very low scores. This is often a sign that your server has a very large disk cache, so one client may be staying in the disk cache while the others are swapping out. If you notice a result like this, try increasing the size of the disk test data file and the I/O range parameter. You can also add more clients.

NOTE: You're more likely to see these scores when you're running only a small number of clients. This is because cache is large when you have only a few clients. Once you have everybody using cache equally, you'll see that service starts to even out.

Here's some background on the results

The next sections contain some background on what makes ServerBench's results useful as a measure of server performance.

As we've said before, ServerBench measures applications servers running in a client/server environment. The ServerBench tests interact with the server at a high level. For example, the disk tests measure how the server manages disk I/O, not how fast the disk is per se. ServerBench is not directly concerned with how fast bits move through the wire. ServerBench is only concerned with how long it takes a transaction to complete. ServerBench tests a server as you would use the server.

NOTE: ServerBench does not run any commercial server applications; it is a synthetic benchmark, not an application test.

ServerBench divides the server into three different subsystems: processor and memory, disk, and network. Here're some things you can do with ServerBench's results:

- Compare the performance of different servers.
- Check the speed of the basic server subsystems in a client/server environment.
- Analyze the effect that changing a single variable in a test has on one server's performance.

The individual tests ServerBench uses

ServerBench builds all its test suites using its processor and memory, disk, and network tests.

The Processor test mimics the processor-intensive portions of typical database servers. The test includes data searches, sorts, and integer arithmetic. As it runs, each processor test program on the server consumes a large amount of RAM (about 400 KB).

There are five disk tests. These tests perform random or sequential read or write operations and file appends. When you create a test mix using these tests, you can specify such characteristics as the size of the test file each client uses, the size of the chunks in which the test moves data, and where on your server ServerBench creates these files.

The network tests basically read and write data using the client-to-server network connection. As with the disk tests, you can enter key test parameters, such as the total amount of data to move over the network and the size of the chunks in which the data should move.

The transactions containing these tests continue to run until the time you specified as the Length of the test expires. (Actually, because ServerBench always waits until the end of a mix iteration, the mix usually runs slightly longer than the value specified in Length.)

The standard ServerBench test suites

If you're just running the standard test suites, then you should get a good indication of how well your server will perform in a client/server environment. This is because we modeled the test suites after typical database server applications. The standard system test suite roughly balances the amount of processor and disk subsystem load it places on a server, along with a modest amount of network work.

You can also check the performance of a particular subsystem. The disk test suite targets your server's disk subsystem, the processor test suite focuses on the server's processor subsystem, and network test suite measures the server's network subsystem.

For more information on these test suites, see Chapter 11.

Deciding which result is really the best for you

When you look at the ServerBench results, keep in mind that the server with the best results may not be the best server for you.

For example, if you are evaluating your system, you need to consider not just which server has the best throughput for the largest number of clients or the highest

peak TPS, but also how that server handles the decline in throughput. Plot a results graph and look at the knee in the results curve (this is the point at which throughput levels off or begins decreasing — if you don't have a knee in your curve, then you need to do some more testing).

Depending on how many clients your server supports, your needs may be better met by a server that shows a slow decline in throughput instead of a server that shows a higher throughput with a lower number of clients before dropping off sharply.

How ServerBench calculates scores

ServerBench uses several factors in calculating its results:

- The amount of time each transaction takes to complete.
- The number of completed transactions. ServerBench does not count incomplete transactions or transactions that began during Ramp up or Ramp down.
- The weight given to each transaction in the mix.

The overall ServerBench score for your server is calculated using a weighted harmonic mean.

You'll notice that the amount of time for a completed transaction includes the time the transaction spends:

- Traveling along the network, both to and from the server.
- Waiting in a queue on the server to receive a service.
- Receiving the service; for example, if the transaction requires disk service, this is the amount of time the disk took to provide the service.

NOTE: Once the client receives a response from the server and stops its transaction timer, the client validates the response. This validation does not cause any overhead on the server and is not included the test time.

Each client keeps track of the number of completed transactions it has and the time each of those transactions took. It then divides the total number of transactions by the time they took to complete. This is the individual client transactions per second score.

Comparing your server's results

You can use ServerBench to compare different servers by running the same test suites on each server. For example, you can run the standard ServerBench test suites.

If you want to compare the performance of different servers, follow this procedure:

- Follow the same testing procedure on each server. For example, if you reboot one server before starting ServerBench, then reboot any other servers you're using in the comparison.
- Use the same test suites with the same parameters on all the servers you're testing.
- Use the same testbed for all the tests. Because the testbed affects your results, you cannot reasonably compare results from different testbeds.
- If the servers you are comparing use the same operating system, make sure they are both set up the same way. For example, if they use tunable parameters, set the tunable parameters to the same value on both systems.

NOTE: Comparing server scores when one server ran ServerBench with 128 MB of memory and the other had 64 MB of memory will not give you a true measure of the relative speeds of the two servers. However, you can determine what effect using memory has on your server by running it one time with 128 MB and another time with 64 MB and then comparing the two results.

Any differences in the test suites or the test setup can affect the results ServerBench produces and invalidate your comparison.

You may also want to compare your server's results with published results for other servers.

End of chapter

Chapter 14

Publishing Your ServerBench Results

This chapter gives you some tips on what you need to do to publish your ServerBench scores and meet the terms of the ServerBench License Agreement. In addition, it directs you to publications where you may see information about how other servers scored on ServerBench.

If you want to publish results, read this

The ServerBench License Agreement requires that you include certain information any time you publish or distribute ServerBench results.

NOTE: The License Agreement at the front of this manual and in the **README.TXT** file states exactly what information you must include when you publish your results.

If you want to publish your ServerBench results, you will need to state what your results were and information on exactly how you had your test server and PCs set up. This means you will need to provide the information about your server, testbed, and ServerBench components.

You can get this information from the disclosure database that you set up when you view the results. (For more information on the disclosure database, see Chapter 12.)

Basically, you'll need to provide the following information if you want to publish your ServerBench results:

- Server Disclosure
 - Machine name
 - Number and type of CPUs, including clock speed
 - Size of hardware CPU cache
 - Amount of memory
 - Type of I/O bus
 - Number and type of hard disk controllers
 - Number and type of hard disks
 - Disk organization (striped, mirrored, RAID, etc.)
 - Disk controller driver name and version
 - Number and type of network controllers
 - Network controller driver name and version
 - Network operating system name and version
 - Any relevant modifications to default network operating system parameters
- Client Testbed Disclosure
 - Network type (10Base-T, Token Ring, etc.)
 - Number of clients
 - Number and type of hubs/concentrators (full duplex, switching, etc.)
 - Number of clients/segment
 - Client OS (Windows 95, Windows for Workgroups)
 - Client CPU type and speed in percentages
 - Client network controller broken down by percentages
 - Client network software name and version (drivers, protocols)
 - Size of any client network cache
- Controller Disclosure
 - Controller network software name and version(drivers, protocol)
- ServerBench Disclosure
 - ServerBench version
 - Description of the test parameters for each mix in the test suite

Sample ServerBench disclosure sheet

The following example shows the type of disclosure information you'll need to provide if you publish or distribute your ServerBench results. This information is taken from one of our test runs as we developed ServerBench 3.0.

Sample Server Disclosure

Name	Compaq ProSignia
CPU	1 Intel 486DX2/66
CPU cache	256 KB
Memory	64 MB
I/O bus	EISA
Disk controller	Compaq Integrated Drive Array
Disk drives	2 Conner CP3541 512 MB drives
Disk driver	cpqda386.dsk 12,621 5/29/92
Disk organization	hardware striping
Network controller	Compaq 32-bit NetFlex controller
Net driver	cpqethnw.lan 40,9059 /11/92
NOS	NetWare 4.1
NOS parameters	set maximum physical receive packet size = 4202

Sample Client Testbed Disclosure

Network	10Base-T
Clients	32
Hubs	8 Accton EtherHub-8mini
Clients/seg	16
CPU	75% 486/25, 25% 386/20
Network cards	75% Eagle NE2000, 25% Cabletron E20
Software	Windows 95, Microsoft TCP/IP using Windows Sockets Version 1.1
Cache	32-bit-file-access enabled, 1MB cache

Sample Controller Disclosure

Controller software	Microsoft Windows 95
---------------------	----------------------

Sample ServerBench Disclosure

Version	3.0
Mixes	standard test suite SYS_60.TST

ServerBench terms and measurements

If you publish your test results, you will want to use the names associated with ServerBench correctly. The following list contains some of the key ServerBench terms and how you should use them:

- **ServerBench® 3.0.** This is the name of this benchmark. The term "ServerBench" is a registered trademark of Ziff-Davis Publishing Company.

ServerBench is a Ziff-Davis benchmark that lets you measure the performance of an applications server in a client/server environment. It provides you with an overall score for your server. ServerBench is a portable benchmark and currently runs on several server platforms (see the ServerBench 3.0

README.TXT file for a complete list of which platforms ServerBench runs on).

- **Test suites.** A group of test mixes that ServerBench executes in sequence.
- **Test mixes.** A group of transactions and the parameters for those transactions.
- **TPS.** ServerBench reports all of its results in terms of TPS (transactions per second). A completed transaction is one where the client submitted a request to the server and then received a response from the server saying the transaction was complete. The client starts a timer when it issues the request and stops it when it gets a response from the server.
- **Transaction.** ServerBench defines a transaction as one or more individual tests bundled together. During a mix, each client submits each transaction as a single request across the network to the server.

End of chapter

Part 5

If You Have a Question

In this part of the manual, we've include chapters aimed at answering your questions.

Chapter 15 tells you how to get in touch with us...to ask a question or to request other benchmarks.

The last chapter in this manual is a quick overview of some of the questions we get about ServerBench. By now you probably know the answers to these questions. However, if you're in a hurry and just want to learn a little about ServerBench, then you'll want to read this chapter.

And if you have any questions about the terms, check the glossary.

Chapter 15

Contacting ZDBOp

This chapter explains the different ways you can contact the Ziff-Davis Benchmark Operation (ZDBOp). It also tells you how to get copies of ServerBench and other Ziff-Davis benchmarks.

Look here for possible solutions to problems

If a problem occurs while you're running ServerBench, here're some places to look for solutions:

- Check the on-line ServerBench manual for your server operating system. It contains some troubleshooting tips as well as any error messages you might see.
- Look in the **README.TXT** file for your ServerBench platform. This file contains any late-breaking information that did not make it into the manual. You'll find a copy of this file in the controller installation directory and on the CD-ROM in the directory that contains ServerBench for your operating system.
- Check the FAQ file for ServerBench, which we post on ZD Net™/CompuServe® Edition. (ZD Net is a Ziff-Davis on-line service. Access to CompuServe is available for a fee.)
- Read the messages in the ZDBENCH forum on ZD Net/CompuServe Edition to see if anyone else has had a similar problem.

Here's how you get in touch with ZDBOp

If you have a problem, fill out the Problem Report that is at the back of this manual and send it to us. If you have an Internet e-mail address, please include it when you get in touch with us. You can get in touch with us in several ways:

-
- If you have a modem and communications software, you can reach ZDBOp via its forum (GO ZDBENCH) on ZD Net/CompuServe Edition.

The address for our World Wide Web page is:

<http://www.zdnet.com/zdbop>

You can also send an on-line version of the Problem Report to us via e-mail to:

zdbopwebmaster@zd.com

- You can fax the Problem Report and any questions and comments you have directly to ServerBench Technical Support at the ZDBOp fax number 919-380-2879.
- You can mail the Problem Report and any questions and comments you have to ZDBOp at the following address:

Ziff-Davis Benchmark Operation
1001 Aviation Parkway, Suite 400
Morrisville, NC 27560

Attn: ServerBench Technical Support

Remember, if you're having a problem with ServerBench, fill out the Problem Report at the back of this manual and send it to us.

Reporting a problem

If you have an Internet e-mail address, please include it when you get in touch with us.

If you want to get a copy of a benchmark

We provide our benchmarks free of charge to anyone who wants them. You can get copies of ServerBench and other Ziff-Davis benchmarks by downloading the benchmarks from ZD Net/CompuServe Edition and ZD Net/World Wide Web Edition or by requesting them from ZDBOp.

Getting copies of benchmarks from ZD Net

You'll find copies of ServerBench and other Ziff-Davis benchmarks on ZD Net/CompuServe Edition and ZD Net/World Wide Web Edition.

NOTE: You'll need to pay your usual connection fees when you download the benchmarks.

When you download ServerBench, you'll need to download three files for your server platform — one each for the server program, the controller program, and the

client program. We provide a different set of files for each server platform ServerBench runs on.

To identify these files, we use the following naming scheme:

- `<port-name>s.<exe or tar>` for the server file.
- `<port-name>co.exe` for the controller file.
- `<port-name>cl.exe` for the client file.

where *port-name* is an abbreviated name we use to indicate that server operating system for that platform of ServerBench.

Requesting the benchmark CD-ROMs from ZDBOp

We distribute our server and PC benchmarks free of charge on CD-ROMs. The Server Benchmarks CD-ROM contains ServerBench and NetBench® (Ziff-Davis' industry-standard file server benchmark program). The Ziff-Davis PC Benchmarks CD-ROM contains WinBench® 96 , and Winstone® 96. You can also get a copy of the Ziff-Davis Macintosh Benchmark CD-ROM, which contains MacBench® 3.0 (a Mac™ OS system benchmark program).

To get a copy of either CD-ROM, send your request to ZDBOp. You can use the Benchmark Request Form located at the end of this manual. You can fax the form to the ZDBOp dedicated benchmark request fax number (919-380-2879), or you can mail the form to ZDBOp at the address listed at the top of the form. The benchmark will arrive via third-class US mail in four to six weeks.

If you want to receive a benchmark sooner and you have a Federal Express account, include your Federal Express account number and shipping instructions on the fax form. We'll send the benchmark out via Federal Express and charge the shipping cost to your Federal Express account.

End of chapter

Chapter 16

Frequently Asked ServerBench Questions

This chapter presents the answers to some of the questions we hear most often about ServerBench.

Q. What is ServerBench?

- A. ServerBench is a Ziff-Davis benchmark program that lets you measure the performance of applications servers in a client/server environment.

We created ServerBench so that you can use it to test different servers as they provide a variety of services to Windows-based clients. ServerBench provides you with an overall score for your server.

We modeled ServerBench after an application server environment. In this environment, most of your data and your applications exist on the server. The client PCs send requests for service to the server. The server executes the requests and sends a response back to the client. ServerBench computes an overall score that represents how fast the server handled the client requests.

Q. Is ServerBench "freeware" or "shareware"?

- A. Neither. All Ziff-Davis benchmarks are licensed software we distribute for free. You must license and register your copy of the benchmark program before you can use it. The License Agreement for each benchmark program appears on your screen the first time you run the benchmark.

Q. How does the way NetBench tests servers differ from the way ServerBench tests servers?

- A. NetBench and ServerBench measure different types of server environments. NetBench measures file servers while ServerBench measures application servers. In a file server environment, you have your application (such as a word-processing or spreadsheet program) running totally on the client PC and you're just using the server to access data. This means that the server's disk I/O speed and the network I/O speed are the major areas that affect your test score.

In an applications server environment, both the data and the application exist on the server. The client PC is primarily a front-end you use as an access point to the application, even though it does do some work as well. In this environment, the CPU power of the server plays a much greater role in your test scores.

NetBench accesses data through a well-defined, publicly available API. ServerBench only communicates with itself; it uses a proprietary client program to send a ServerBench request along the network to a proprietary server program that decodes the client request. Then the server program returns a response to the client program. NetBench returns scores showing I/O throughput for file servers while the ServerBench scores show how well servers handle client requests for a variety of operations.

ServerBench also lets you measure how individual server subsystems perform. These subsystems are transparent to the NetBench tests.

In addition, NetBench only runs a controller program and client programs while ServerBench runs a server program as well as a controller program and client programs. As a result, NetBench does not require any special network software other than what you currently have to allow your server and PC clients to communicate. ServerBench, however, only works with specific network protocols; if you don't have them, then you must buy them.

Q. How many clients can I have connected during a test run?

- A. ServerBench supports up to 1,000 clients; however, your operating system or network software may limit you to fewer than that. Don't worry, though. Because ServerBench uses stress tests, you can get an accurate measure of your server's performance with as few as three or four clients.

Q. Do different clients put different loads on the server?

- A. Yes. If one client has a more powerful processor and/or a faster network adapter or driver, it will issue transaction requests faster than a client with less powerful components. As a result, the first client will put a greater load on the server. This client will also produce a higher score for a mix. One consequence of this fact is that you can't meaningfully compare the results of a test using one testbed with the results of a test using another testbed.

Q. What do you mean by "stress tests"?

- A. When we say stress test, we're talking about a type of test that lets us impose arbitrarily large amounts of work on the server. These tests exercise the server in less time and using fewer clients than it would take for you to monitor the server's day-to-day performance.

Q. I'm confused by all these tests. Which ones should I run?

- A. To help you with this problem, we supply some standard test suites for you to run. These are the same test suites the Ziff-Davis publications use when they prepare articles dealing with ServerBench. For testing purposes, the key standard test suite is **SYS_60.TST**. This is an overall system test suite. If you just want to quickly run ServerBench to get a feel for how it works, you can run the test suite **SAMPLE.TST**. This is a demonstration test suite we provide to give you a sense of what's involved in running a ServerBench test suite without spending a lot of time. **SAMPLE.TST** only takes a couple of minutes to run, while the other test suites can take a few hours to run.

Q. Do ServerBench clients create data on the server?

- A. If a test mix includes a disk test (e.g., sequential read, sequential write, random read, random write, or append), the ServerBench server application creates a data file on the server for each client PC included in that test mix. The server then uses these data files to perform the disk tests for the clients. When the tests end, ServerBench deletes all the data files it created.

If you're running the ServerBench standard test suites, ServerBench creates these data files in its installation directory on the server. (You can specify a different location for these files when you create a test suite.)

The ServerBench server application does not create any data on the server for processor subsystem tests or network subsystem tests.

Q. When the clients execute a mix with more than one transaction, do they all request the transactions in the mix in the same order?

- A. No. Each client randomly reorders the transactions in the mix. The client determines the new order for the transactions by generating a random series of numbers based on the client's ID number. Because each client has a different client ID number, each client can create a different order (if there is the right balance of transactions and number of clients – for example, if there are only two transactions and 10 clients, then many of the clients will execute the transactions in the same order). When the client executes the mix, it requests transactions based on that order. The client repeats this same order of transactions for the duration of the mix. In addition, because the each client uses its ID number to determine its transaction order, the client will generate the same transaction order the next time it executes that mix. So this ordering method allows for repeatability.

Here's why the clients reorder transactions:

- To better reflect a real-world setting. In the real world, a server must handle a variety of operations concurrently.
- To avoid overloading one server subsystem. If all the clients send the same request to the server, one server subsystem will be overwhelmed with work while the others wait for something to do. The scores you'd get from a situation like this wouldn't accurately reflect how well your server can handle client requests.

Q. How does ServerBench report its results?

- A. ServerBench reports its overall score as transactions per second (TPS). It uses a weighted harmonic mean to calculate this score.

Q. I'm confused about when clients actually record test results. If they aren't constantly recording the mix results, how do they know which results to record?

- A. We designed ServerBench so that you can measure the server's performance when it is in a steady state (i.e., it has an even load). To do this, we let you set three mix parameters: Ramp up, Ramp down, and Length. The clients ignore any results from mix iterations that begin during the Ramp up or Ramp down periods. As a result, no client records results that occur when the server has a light load, such as at the beginning of the test when not all the clients have started sending requests or at the end of the test when some clients have already stopped sending requests. The clients do record results for any iteration of a mix that begins before the Ramp down period starts, even if that iteration finishes during the Ramp down period. This is because your Ramp down period is usually long enough so that other clients will still be sending requests when a client finishes the iteration it sent before the Ramp down period began.

The Length parameter is the number of seconds that the mix executes. This parameter includes the time for both the Ramp up and Ramp down periods. Thus, if you don't want the clients to ignore all the results for this mix, you need to set the value of Length to a period that exceeds the sum of the Ramp up and Ramp down periods by several iterations.

End of chapter

Glossary

Client

One of several PCs running Windows 95 or Windows for Workgroups that makes requests of the server.

Client/server model

In ServerBench's client/server model, each Windows-based client makes requests of the server for services that involve disk time, processor time, and/or network time. The server replies to each requesting client. Each client request that the server replies to is called a completed transaction. Each client divides the number of completed transactions by the total time it took to complete them to determine its raw TPS score. Since ServerBench is a client/server model, the final score reflects network time and time spent waiting in a queue for a service, not just the speed at which the server executes the request.

Controller

A PC running Windows 95 or Windows for Workgroups that you use to start, stop, and monitor the ServerBench tests. Unlike the clients, the controller does not run any of the ServerBench tests and is counted separately.

Deviation (also called variance)

This is the amount by which a single client deviates from the mean of a ServerBench test.

Disclosure database

This is information on how your server and clients were set up during a test run, such as which operating system they were using and so on. You need to be able to provide this information if you want to publish your ServerBench results. ServerBench can capture some of this information for you, but not all of it. So we created a database to allow you to set up the information once. You can then use with all your other test runs (if appropriate). You can also easily modify it if you make a change to your server or the clients. ServerBench displays this information in Tables 7 and 8 of the results spreadsheets.

Harmonic mean

ServerBench calculates its scores using a weighted harmonic mean. A harmonic mean is a way of combining scores to create a single, representative score for the test.

Iteration

An iteration is one complete pass through the mix. For example, each time a client has requested all of the transactions in a mix, we say that it has completed one iteration of the mix.

Knee

This is the point in a results curve where the throughput no longer increases; it either begins decreasing or flattens out.

Master provider

The process or thread on the server that provides the line of communication between the controller and the service providers.

Processor scaling

This is a procedure that deals with how well performance improves as you add processors.

Provider

Provider is a generic term representing the ServerBench process or thread on the server.

Provider model

This is the generic term for the model ServerBench uses to implement client/server requests on the server.

Server

The machine running the server software. For ServerBench's purposes, a server involves the machine running the server software, the network protocol used to communicate with the clients, and the different subsystems that the server uses in order to reply to client requests.

Service providers

The processes or threads on the server that do the work the clients request.

Singleton

A transaction that consists of a single subsystem test. You designate a singleton by entering the mnemonic for that test as the transaction name in the Transaction Name box in the Mix Definition window.

Snapshot file

This is the database file that contains information about your server and client testbed.

Standard deviation

Standard deviation. This is a unit of measurement used to determine how closely the scores are clustered around the mean. ServerBench displays information about the standard deviation in a mix so you can tell how the clients scores are dispersed relative to the mean for the mix. Basically, standard deviation is a measure of what's normal. To calculate a standard deviation, divide the maximum distance from the mean by the mean.

Starving clients

A starving client is one that does not receive service from the server. There are several reasons why a client might not receive service. For example, the client's request may get caught in a queue or the transaction the clients are performing may be very long while the total test time is short, causing the test to end before all clients receive service.

Steady state

This is the point in the test where the subsystems of the server are executing a steady workload. This is the point at which we like to measure your server's performance. The server is no longer thrashing as a result of starting the tests and it has flushed out any residual caching effects that occurred as a result of creating test data files. All the clients are executing the tests, so the server does not have a light load, which occurs when only a few clients are running tests (such as during the Ramp up or Ramp down periods when not all the clients have started the tests yet or some of the clients have already ended the tests). Steady state implies that the length of the queue of threads or processes waiting for processor time, the number of blocks of the file cache in use, and the length of the queue of messages to be transmitted on the network have all generally stabilized and are responding appropriately based on the load the different tests are putting on them (remember that different tests stress different areas of the server).

SYS_60.TST

The standard system test suite, which consists of test mixes containing all of ServerBench's tests. The ServerBench standard system test suite does not focus on a single subsystem; instead it is an overall test that roughly balances the amount of process and disk subsystem load it places on the server while including a modest amount of network work. We designed the to model the workload of a database truncations processing environment.

System throughput

The amount of work the server does and the number of clients receiving service in a given time.

Test

This is a test that exercises one of your server's subsystems. ServerBench has seven basic tests. You can vary the parameters for the tests. You can also bundle one or more tests together to create a transaction.

Test mix

A test mix is a group of transactions and the parameters for those transactions. You can vary these for each mix.

Test suite

A test suite is a group of test mixes that ServerBench executes sequentially. All the ServerBench standard test suites use a **.TST** extension.

TPS

Transactions per second. ServerBench returns scores for its tests in terms of TPS, or the number of completed transactions per second that occur. A completed transaction is one where the client submitted a request to the server and then received a response from the server saying the transaction was complete. Each client keeps a record of how long it takes to get a response back from the server once it makes a request. When the test finishes, ServerBench determines TPS by dividing the total number of transactions by the time in seconds that they took to complete. ServerBench measures transaction time on the clients, not on the server.

Transaction

ServerBench defines a transaction as one or more tests that are bundled together. During a mix, ServerBench submits each transaction as a single request across the network to the server. ServerBench measures the transaction time as the time it takes from the moment a client makes a request of the server until the client receives a response from the server.

Variance (also called deviation)

This is the amount by which a single client deviates from the mean of a ServerBench test.

End of glossary

Index

—.—
.PTH
 extension for path name file, 109
.SNP extension for snapshot file
 Snapshot file, 228
.TST extension
 test suites, 165

—A—
Aborting a test, 143
 time out occurs, 143
ALT-TAB key sequence
 switching windows, 132
Applications
 running in background, 151
Applications server, 23, 51
 ServerBench model, 35
AUTOEXEC.BAT, 151

—B—
Background applications
 affecting results, 151
Benchmark
 synthetic, 50
Benchmarks
 downloading, 260
 requesting, 260
Bottlenecks, 243
Browse button, 93

—C—
Cache, 56
Client grid, 79
Client groups, 185
Client information box, 82
Client window, 114
Client/server
 explanation, 50
Client/server model
 definition, 269
Clients, 24
 connecting terminated clients, 14, 146
 definition, 24, 269
 disclosure information, 232
 disconnecting, 146
 disconnecting from ServerBench, 120
 equal to more than one user, 53
 information on controller, 81
 maximum number, 25
 program, 37
 response time, 61
 running Windows 95, 13
 running Windows for Workgroups, 13
 stages in test, 144
 stress servers differently, 54
 test states, 114
Color legend
 controller window, 80
 figure, 80
CONFIG.SYS, 151
Controller

- client grid, 79
- color legend, 80
- defined, 24
- definition, 269
- new features, 18
- pause option, 140
- program, 37
- running minimized, 18
- running Windows 95, 13
- running Windows for Workgroups, 13
- setting error handling, 133
- setting external notification, 135
- switching to main window, 132
- tasks you perform there, 132
- Controller window, 76, 78
 - client information box, 82
 - external notification dialog box, 89
 - figure of function buttons, 83
 - Test Suite History window, 87
 - unattended mode dialog box, 87
- Copy a Mix From dialog box, 113
- Copy menu
 - Copy path from template option, 184
- Copy Mixes From dialog box, 102
- Copy path from template option
 - copy client path names, 184
 - Copy menu, 184
- Copy Test Suite File dialog box, 112
- Create or Select Test Suites dialog box, 99

—D—

- D_60.TST test suite, 201
- Delay
 - Parameters, 172
- Delay time
 - test parameter, 172
- Dependent fields, 111
- Deviation
 - definition, 269
- Directories
 - copying client path names from template, 184
 - sticky, 138
 - sticky directories feature, 19
- Disclosure information
 - publishing ServerBench's results, 253
- Disclosure database

- definition, 269
- Table 4, 217
- Table 5, 219
- using, 227
- Disconnecting
 - clients, 146
 - from ServerBench, 117
 - server, 146
- Disk cache, 56
- Disk subsystem, 35
- Disk test
 - parameter changes, 17
- Disk Test parameter, 178
- Disk test suite
 - changes, 16
- Disk test suites, 201
- Disk tests, 196
- Documentation, 1
 - on-line port manual, 4
 - on-line quick start manual, 4
- Duplicate Mix Fields Across Suite dialog box, 110

—E—

- Error handling, 133
 - running a notification program, 135
- Excel
 - used to display results, 31
- Extensions for files
 - .PTH, 109
- External notification dialog box, 89

—F—

- Faxing ZDBOp, 260
- Figures
 - Choose Test Suites, 91
 - client information, 82
 - color legend, 80
 - controller window, 78
 - Copy a Mix From, 113
 - Copy Mixes From, 102
 - Copy Test Suite File, 112
 - Create or Select Test Suites, 99
 - Duplicate Mix Fields Across Suite, 110
 - External Notification dialog box, 89
 - main ServerBench window, 76

Mix Definition, 104
Mixes in Suite, 100
Reorder Suite Mixes, 103
Select Results, 94
Select Results Path, 93
Select Test Suites, 102
Selected Test Suites, 92
Test Suite History window, 87
Unattended Mode dialog box, 87
View Results, 96

File extensions
results, 30

File size parameter
effect on test results, 178

Files
.PTH extension for path name file, 109
AUTOEXEC.BAT, 151
CONFIG.SYS, 151

—G—

General Help, 4

—H—

Hardware/software
effect on performance, 153

Harmonic mean, 64
definition, 270

Help, 4
General, 4
on-line, 123
Port Specific, 4

—I—

I/O range parameter, 179

Iteration
definition, 270

Iterations
measuring, 239

—K—

Key sequences
ALT-TAB, 132
Knees in results curves, 58

definition, 270

—L—

Length parameter
Length, 169
Length paramter, 239
License Agreement, 253

—M—

MacBench
Ziff-Davis benchmark, 261

Main ServerBench window, 75

Master provider
definition, 270

Mix Definition window, 104, 167
changes, 17

Mixes
adding to test suite, 101, 191
copying fields across suites, 110
copying fields across test suite, 188
deleting, 192
deleting mixes, 192
dependent fields, 111
null path names, 111
reordering in test suite, 103, 191
Mixes in Suite dialog box, 100
Multitasking, 57

—N—

N_60.TST test suites, 201

Names
.path name file, 109

NetBench
comparison table with ServerBench, 70
comparison with ServerBench, 67, 264
used with ServerBench, 32

Network protocol, 28

Network protocols
using TCP/IP, 14

Network subsystem, 36

Network test suites, 201

Network tests, 200

Notification program
 running when an error occurs, 135

—O—

OS/2 Warp Server platform, 13
Overall ServerBench score, 237

—P—

P_60.TST test suite, 201
Pager programs
 executing from the controller, 135

Parameter
 Disk Test, 178
 Test iterations, 177
 Transaction iterations, 177

Parameters
 Delay, 172
 Disk Test File Initial Size, 176
 effect on tests, 154
 Length, 169
 Ramp down, 169
 Ramp up, 169
 Request Size, 175
 SAMPLE.TST, 202
 test iterations, 175
 Test type, 175
 Think time, 172
 Total size, 176, 178
 transaction iterations, 174

Path names
 copying client path names from template, 184
 null, 111

Pause option, 140

Performance
 affected by hardware/software, 153
 keep AUTOEXEC.BAT clean, 151
 keep CONFIG.SYS clean, 151

Port-Specific Help
 General, 4

Processor scaling, 62
 definition, 270

Processor subsystem, 35

Processor test, 194

Processor test suite
 changes, 16

Production network, 11, 151
Proprietary programs, 26
Provider model, 45
 definition, 270
Publishing results, 253
 disclosing server information, 254
 disclosing ServerBench information, 254
 disclosing testbed information, 254
 sample disclosure sheet, 254
 sample server disclosure, 255
 sample ServerBench disclosure, 255
 sample testbed disclosure, 255

—R—

Ramp down parameter, 169, 239
Ramp up parameter, 169, 239
README file, 1, 2
Reorder Suite Mixes dialog box, 103
Repeatability in testing, 65
Request Size parameter, 177
Results
 abnormal, 248
 affected by hardware/software, 153
 duplicate names, 92
 file extensions, 30
 How ServerBench calculates them, 238
 publishing, 253
 standard deviation, 63
 summary of new features, 14
 svrbench.xla, 15
 Table 1, 211
 Table 2, 213
 Table 3, 214
 Table 4, 217
 Table 5, 219
 Table 6, 221
 Table 7, 223
 use Excel, 31
 use Excel workbooks, 14
 variance graph, 224
 variance in scores, 63
 viewing, 126, 206
Results curves
 knees, 58
Results workbook, 210
RISC platforms

Digital Alpha, 13
MIPS, 13
PowerPC, 13
Running ServerBench
background applications, 151

—S—

SAMPLE.TST
parameters, 202
running, 125
Scores
use Excel, 31
Select Reports window, 230
Select Test Suites dialog box, 102
Server
affect of subsystems, 52
applications, 51
definition, 270
disconnecting from ServerBench, 117
disconnecting, 146
disconnecting from ServerBench, 120
program, 37
publishing results, 254
stressing, 53
throughput, 61
Server Benchmarks CD-ROM
README file, 1
Server disclosure
publishing results, 254
sample, 255
Server subsystems, 35, 52
ServerBench
allowing time for a test, 132
before you run test suites, 131
clients, 24
comparison table with NetBench, 70
comparison with NetBench, 67, 264
decription, 9
definition, 23
documentation, 1
help, 123
main score, 237
new features, 13
on-line help, 4
on-line port manual, 4
OS/2 Warp Server platform, 13

overview, 23
programs, 37
proprietary programs, 26
publishing results, 253, 254
quick start manual, 4
README file, 2
requesting, 260
RISC platforms, 13
running SAMPLE.TST, 125
starting, 117
subsystems, 25
summary, 9
synthetic benchmark, 50
test scenario, 46
test setup (summary), 9
testing procedure, 150
tests, 26
viewing results, 126
ServerBench disclosure
publishing results, 254
sample, 255
ServerBench Quick Start Handbook, 4
Service
uneven, 241
Service providers
definition, 270
Singleton
definition, 270
Snapshot file, 228
creating, 228
definition, 270
updating, 232
Stages
client, 144
Standard deviation
definition, 271
in results, 63
Standard test suites
SAMPLE.TST, 202
Starting ServerBench, 117
Steady state, 155
definition, 271
Sticky directories
sticky, 138
Stress tests, 25, 53
Subsystems, 25
disk subsystem, 35

- network subsystem, 36
- processor subsystem, 35
- server, 35
- svrbench.xla module, 15
- Synthetic benchmark, 50
- SYS_60.TST test suite
 - system, 201
- System test suite
 - changes, 16
- System throughput
 - definition, 271

—T—

- Table 1 Client Disclosure, 220
- Table 1 ServerBench Summary, 211
- Table 2 Overall ServerBench Data, 213
- Table 3 Client Data, 214
- Table 4 Server Disclosure, 218
- Table 6 Test Suite Information, 221
- Table 7 Transaction Definitions, 223
- TCP/IP protocol
 - Winsock 1.1-compliant, 14
- Test
 - definition, 272
- Test environment
 - network protocol, 28
 - test scenario, 46
- Test iterations parameter, 177
- Test mix
 - definition, 272
- Test network, 151
- Test parameters
 - Delay, 172
- Test run
 - aborting a test, 143
- Test set up
 - summary, 9
- Test states
 - clients, 114
- Test Suite History window, 18, 87
- Test suites
 - adding mix, 191
 - allowing time to run, 132
 - before you run, 131
 - changes to Disk test suite, 16
 - changes to Processor test suite, 16

- changes to standard suites, 16
- changes to System test suite, 16
- copying mix fields, 110, 188
- creating, 163
- developing, 160
- definition, 272
- disk, 201
- file name, 164
- interrupting, 122
- monitoring, 144
- more clients in one suite, 16
- network, 201
- new features, 16
- parameters for SAMPLE.TST, 202
- Processor, 201
- removing, 140
- reordering mixes, 191
- SAMPLE.TST, 125
- standard, 201
- stress tests, 25
- viewing in Test Suite History window, 145
- viewing results, 126

- Testbed
 - publishing results, 254
- Testbed disclosure
 - publishing results, 254
 - sample, 255
- Testing
 - tips, 151
- Testing procedure, 149, 150
- Testing ServerBench
 - allowing time for a test, 132
 - repeatability, 65
- Tests, 26
 - causing a problem with a running test, 132
 - doing other tasks as tests run, 132
 - effect of parameters, 154
 - stress tests, 53
- Throughput, 61
 - definition, 271
- Time required to run ServerBench, 132
- Total Size parameter, 178
 - changes, 17
- TPS
 - calculating, 56
 - definition, 272
- Transaction

definition, 272
Transaction iterations parameter, 177
Transactions
 calculating, 56
 creating, 174
 defining, 173
 modeled after typical work, 54
 reordering, 238
TSRs
 running on clients, 151

—U—

Unattended mode, 133
 error handling features, 14
Unattended mode dialog box, 87
Users
 represented by clients, 53

—V—

Variance
 definition, 272
Variance graph
 creating, 224
View Results dialog box, 96

—W—

Weighted harmonic mean, 64
WinBench
 Ziff-Davis benchmark, 261
Windows
 Choose Test Suites dialog box, 91
 client, 114
 client grid, 79
 client information box, 82
 controller window, 78
 Copy a Mix From dialog box, 113
 Copy Mixes From dialog box, 102
 Copy Test Suite File dialog box, 112
 Create or Select Test Suites dialog box, 99
 Duplicate Mix Fields Across Suite dialog box,
 110
 External Notification dialog box, 89
 main ServerBench window, 75
 Mix Definition, 104
 Mix Definition window, 167

Mixes in Suite dialog box, 100
Reorder Suite Mixes dialog box, 103
Select Reports, 230
Select Results dialog box, 94
Select Results Path dialog box, 93
Select Test Suites dialog box, 102
Selected Test Suites dialog box, 92
Test Suite History, 87
Unattended Mode dialog box, 87
View Results dialog box, 96
Windows 95
 on clients, 13
 on controller, 13
Windows for workgroups
 on clients, 13
 on controller, 13
Winsock 1.1
 with TCP/IP, 14
Winstone
 Ziff-Davis benchmark, 261
Workbook
 results, 210
World Wide Web page, 260
 zdbopwebmaster@zd.com, 260

—Z—

ZD Labs, 133
 test procedure, 149
ZD Net
 contacting ZDBOp, 260
ZD Net/CompuServe Edition, 259, 260
ZD Net/World Wide Web Edition, 260
ZDBENCH
 ZDBOp forum, 260
ZDBOp
 CompuServe forum, 260
 contacting via ZD Net, 260
 contacting via zdbopwebmaster@zd.com, 260
 fax number, 260
 forum GO ZDBENCH, 260
 Macintosh Benchmark CD-ROM, 261
 mailing address, 260
 PC Benchmarks CD-ROM, 261
 World Wide Web page, 260
Ziff-Davis Server Benchmarks CD-ROM, 261

Problem Report Form for ServerBench

Information about you:

Name: _____

Address: _____

Company: _____

Phone or Fax: _____ E-Mail _____

Describe your ServerBench setup:

Please send us information on the version of ServerBench you're using. If you've set up the disclosure information for the server and client results tables (tables 4 and 5), you can just send us those tables.

ServerBench version number: _____ for server operating system: _____

Server: _____

Controller: _____

Client: _____

Network operating system: _____

Network protocol: _____

Details of the problem:

Which test suite were you running? (If you were running a test suite you created or a modified standard test suite, send us a copy of the test suite.) _____

Please write any server error messages you received here: _____

Please write any controller error messages you received and the stage the test suite was at here:

Please write any client error messages and the stage the client was in here: _____

Can you reproduce the problem? _____ Other comments: _____

Please send this form to:

Fax number: **(919) 380-2879**

or Mail: Ziff-Davis Benchmark Operation
1001 Aviation Parkway, Suite 040
Morrisville, NC 27560
Attention: ServerBench Technical Support

Benchmark Request Form

Please check the boxes of the products you want:

Ziff-Davis PC Benchmarks CD-ROM

Contains the **Winstone®** and **WinBench®** benchmarks for desktop PCs.

Ziff-Davis Server Benchmarks CD-ROM

Contains **NetBench®** for file servers with DOS, Windows for Workgroups, and Mac™ OS system clients and **ServerBench®** for client/servers for the server platforms Windows NT™ Server 3.51 using Digital™ Alpha™, MIPS®, PowerPC™, and x86-compatible processors, SCO® UnixWare®, SCO OpenServer Release 5, OS/2® Warp Server, NetWare® 4.1, and NetWare 4.1 SMP.

Ziff-Davis Macintosh Benchmark CD-ROM

Contains **MacBench®** for Mac™ OS systems and Power Macintosh™.

Please send these products to:

Name: _____

Company: _____

Address: _____

City: _____ State: _____ Zip: _____

Country: _____

Telephone: _____ FAX: _____

We answer requests in the order we receive them. We ship all benchmarks via 3rd-class U.S. Please allow 4-6 weeks for delivery. For faster shipment, provide your Federal Express account information below:

Your Federal Express account number: _____

Check one: priority overnight standard overnight

Please return this form:

Fax to: **(919) 380-2879**

or Mail to: Ziff-Davis Benchmark Operation
1001 Aviation Parkway, Suite 400
Morrisville, NC 27560

Acknowledgments

Numerous people worked together to create ServerBench version 3.0. Members of the primary ServerBench development team are:

Susan Carol Robinson	Technical Documentation
Lee Dorrier	Technical Director
Jeff Downey	Developer
Ronald L. Gittelson	Developer
Kasey Lee	Developer
Keith Turner	Developer

Other people who contributed to the ServerBench 3.0 product are:

Richard Butner
Israel Ehrisman
David Kiem
David King
Bruce Kurson
Irene Lee
Jeff Shafer

The ZDBOp support staff who helped make ServerBench 3.0 possible are:

Elizabeth Barnes
Jennie Faries
Laura Higgins
Libby Kiem
Gina Massel-Castater
Mitchell Moore
Stephanie Walthall

Many people in different parts of Ziff-Davis contributed to the design, testing, and production of ServerBench, including:

David Berlind

Steve Buehler

Warner Cheng

David Chernicoff

John Clyman

Jim Galley

Edward Henning

Bert Jensen

Shawn Kafaipour

Bob Kane

Kason Leung

Chuck Lin

Robert Lipschutz

Tom Mace

Anatoliy Nosovitskiy

Larry Seltzer

Nick Stam

Mark Stanczak

Michael Surkan

Terry Tam

Jeff Witt

Chris Yates