

L-Soft international, Inc.



# Site Manager's Operations Manual

LISTSERV<sup>®</sup>, version 15.0



Last Updated: June 12, 2007

Information in this document is subject to change without notice. Companies, names, and data used in examples herein are fictitious unless otherwise noted. L-Soft does not endorse or approve the use of any of the product names or trademarks appearing in this document.

Permission is granted to copy this document, at no charge and in its entirety, if the copies are not used for commercial advantage, the source is cited, and the present copyright notice is included in all copies. Recipients of such copies are equally bound to abide by the present conditions. Prior written permission is required for any commercial use of this document, in whole or in part, and for any partial reproduction of the contents of this document exceeding 50 lines of up to 80 characters, or equivalent. The title page, table of contents, and index, if any, are not considered to be part of the document for the purposes of this copyright notice, and can be freely removed if present.

Copyright © 2007 L-Soft international, Inc.  
All Rights Reserved Worldwide.

LISTSERV is a registered trademark licensed to L-Soft international, Inc.

ListPlex, CataList, and EASE are service marks of L-Soft international, Inc.

LSMTP is a registered trademark of L-Soft international, Inc.

The Open Group, Motif, OSF/1 UNIX and the "X" device are registered trademarks of The Open Group in the United State and other countries.

Digital, Alpha AXP, AXP, Digital UNIX, OpenVMS, HP, and HP-UX are trademarks of Hewlett-Packard Company in the United States and other countries.

Microsoft, Windows, Windows 2000, Windows XP, and Windows NT are registered trademarks of Microsoft Corporation in the United States and other countries.

Sun, Solaris, SunOS, and PMDF are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries.

IRIX is a registered trademark of Silicon Graphics, Inc. in the United States and other countries.

Linux is a registered trademark of Linus Torvalds.

Intel and Pentium are registered trademarks of Intel Corporation.

All other trademarks, both marked and not marked, are the property of their respective owners.

This product includes software developed by the Apache Software Foundation (<http://www.apache.org/>).

Some portions licensed from IBM are available at <http://oss.software.ibm.com/icu4j/>

This product includes code licensed from RSA Security, Inc.

Manuals for LISTSERV are available in PDF format from **ftp.lsoft.com**. They are also available on the World Wide Web at the following URL:

**URL:** <http://www.lsoft.com/manuals/index.html>

L-Soft invites comment on its manual. Please feel free to send your comments by email to [manuals@lsoft.com](mailto:manuals@lsoft.com)

## Table of Contents

<b>Preface - About This Manual</b> .....	<b>xiii</b>
Editorial Note - New Version Numbering .....	xiii
LISTSERV Command Syntax Conventions .....	xiii
Contacting L-Soft .....	xiv
<b>Section 1 Who Should Read This Book?</b> .....	<b>1</b>
1.1 Where to Find Changes and Updates to the Manual? .....	1
<b>Section 2 Differences Between Architectures and Implementations</b> .....	<b>3</b>
2.1 Differences Between Architectures and Implementations .....	3
2.1.1 The File Server .....	3
2.1.2 The WWW List Archive and List Management Interface .....	3
2.2 Differences between LISTSERV and LISTSERV Lite .....	4
2.3 Operating Systems and Architectures Supported .....	5
<b>Section 3 Principles of Operation</b> .....	<b>7</b>
<b>Section 4 A LISTSERV How-To for Site Managers</b> .....	<b>9</b>
4.1 Installation and Startup Questions .....	9
4.2 Initial Configuration .....	10
<b>Section 5 Configuring Your LISTSERV Site</b> .....	<b>13</b>
5.1 Site Configuration Files .....	13
5.2 What can be configured? .....	14
5.3 Files Used by LISTSERV .....	23
5.3.1 Program Executables .....	23
5.3.2 BITNET Network Table Files .....	24
5.3.3 Internet and Peer Networking Table Files .....	25
5.3.4 LISTSERV's External Data Files .....	25
5.3.5 User Reference Material .....	26
5.3.6 Command Line Utilities (Non-VM) .....	27
5.3.7 GUI Site Configuration Utility (Windows Only) .....	28
5.3.8 Line-mode site configuration utility (OpenVMS only) .....	28
5.3.9 Other files that will appear during use .....	29
5.4 Installing & Configuring LISTSERV's WWW Archive & Administration Interface .....	30
5.4.1 The WWW Archive Interface .....	30
5.4.2 The WWW Administration Interface .....	31
5.4.3 Installing a Web Server .....	32
5.4.4 Installing the Web Archive Interface Script .....	32
5.4.5 Creating a Subdirectory for the Archive Interface .....	34
5.4.6 Configuring LISTSERV to Activate the Web Archive Interface .....	34
5.4.7 Customizing the Web Pages .....	35
5.4.8 Enabling Individual Lists .....	35
5.4.9 Enabling Web-Based Bulk Operations .....	37
5.5 The Spam Detector and Anti-Subscription-Spoofing Feature .....	38
5.5.1 Spam Quarantine .....	38

5.5.2 "Anonymous" Spam Alerts	38
5.5.3 Subscription Anti-Spoofing Feature	39
5.6 Server Registration	40
5.6.1 Registering LISTSERV Classic and Classic HPO Servers	40
5.6.2 How do I find out if my server is already registered?	40
5.6.3 The LISTSERV Backbone	41
5.6.4 Automatic Registration for LISTSERV Lite Servers	42
5.7 Inter-Server Updates	42
5.8 Setting Up Archive and Notebook Directories for Use with LISTSERV	43
5.9 DBMS and Mail Merge Functions	44
5.10 Synonymous Host Name Registration via ALIASES NAMES	44
5.11 Real-Time Anti-Virus Scanning	44
5.12 LISTSERV DomainKeys Support	45
5.12.1 Creating DKIM Keys and Configuring DNS	46
5.12.2 LISTSERV Configuration	48
5.12.3 Starting LISTSERV with DKIM Support	48
5.12.4 Using DKIM with LISTSERV	49
5.12.5 Restrictions and Implementation Choices	49
<b>Section 6 LISTSERV Commands</b>	<b>51</b>
6.1 General Commands	51
6.1.1 List Subscription Commands	51
6.1.2 Other list-related commands	55
6.1.3 Informational commands	58
6.1.4 Commands Related to File Server and Web Functions	60
6.1.5 Other Advanced Commands	62
6.2 List Owner and File Owner Commands	65
6.2.1 File Management Commands (for file owners only)	65
6.2.2 List Management Functions	66
6.3 LISTSERV Maintainer Commands	69
6.4 Sending commands to LISTSERV	72
6.5 Defining Personal Passwords	73
<b>Section 7 Creating and Maintaining Lists</b>	<b>75</b>
7.1 Basic List Creation	75
7.2 Architecture-Specific Steps for List Creation	77
7.2.1 Unix: Creating Required Sendmail Aliases	77
7.2.2 OpenVMS: Creating Required PMDF Aliases	79
7.3 Sample Checklist for Creating Lists	79
7.4 Naming Conventions	80
7.4.1 The "-L" Convention	80
7.4.2 Reserved Names	80
7.4.3 Reserved Characters	81
7.4.4 Maximum Length of the List Name	82
7.4.5 Making the List Name User-Friendly	82
7.5 List Header Keywords	82
7.6 Retrieving and Editing a List	83

7.7 Adding a List Password	85
7.8 Storing a Modified List on the Host Machine	85
7.9 Fixing Mistakes	86
7.10 Sample List Header File	86
7.11 Deleting a List	87
7.12 Adding HTML to a List Header for the CataList	88
7.12.1 Update Latency	89
7.12.2 Inserting a Pointer to Another List	89
7.12.3 Restrictions on the Placement of Equal Signs	90
7.13 Setting Up Lists for Specific Purposes	90
7.13.1 Public Discussion Lists	90
7.13.2 Private Discussion Lists	91
7.13.3 Edited Lists	92
7.13.4 Moderated Lists	93
7.13.5 Semi-Moderated Lists	95
7.13.6 Self-Moderated Lists	95
7.13.7 Private Edited/Moderated Lists	96
7.13.8 Auto-Responders	96
7.13.9 Announce-Only Lists	97
7.13.10 Restricted Subscription Lists with Automatically-Generated Questionnaire	98
7.13.11 Peered Lists	99
7.13.12 Super-Lists and Sub-Lists	102
7.13.13 Cloning Lists	104
7.14 Merging existing LISTSERV lists	106
7.14.1 Merging List A into List B; List A User Options Not Preserved	106
7.14.2 Merging List A into List B; List A User Options Preserved	106
7.14.3 Merging List A and List B into List C	107
7.15 Migrating lists from one site to another	107
7.15.1 Migrating Lists From One LISTSERV Site to Another LISTSERV Site	107
7.15.2 Migrating Lists from Non-LISTSERV Sites	109
7.15.3 Migrating Lists from Sendmail Alias Files, Databases, Etc.	111
7.16 Changing the Name of an Existing List	111
7.17 Bulk Operations (ADD and DELETE)	112
7.17.1 Bulk ADD Operations	112
7.17.2 Bulk DELETE Operations	113
7.18 Content Filtering	113
7.19 DomainKeys Message Signing	116
<b>Section 8 File and Notebook Archives</b>	<b>117</b>
8.1 The File Archive	117
8.2 Starting a file archive for your list	117
8.3 Filelist Maintenance (VM Systems Only)	118
8.3.1 Creating a Filelist (VM Systems Only)	118
8.3.2 Adding FAC Codes (VM Systems Only)	118
8.3.3 Retrieving the Filelist (VM Systems Only)	118
8.3.4 Adding File Descriptors to the Filelist (VM Systems Only)	119
8.3.5 File Access Codes (FAC) for User Access (VM Systems Only)	120

---

8.3.6 Deleting File Descriptors from the Filelist (VM Systems Only)	121
8.3.7 Storing the Filelist (VM Systems Only)	121
8.4 The listname.CATALOG System on Non-VM Systems	121
8.4.1 Adding Files to the SITE.CATALOG	122
8.4.2 Delegating File Management Authority	123
8.4.3 Creating a Sub-Catalog	124
8.4.4 Updating the Sub-Catalog	124
8.4.5 Indexing the Sub-Catalog	126
8.5 Storing Files on the Host Machine	126
8.6 Deleting Files from the Host Machine	127
8.7 Automatic File Distribution (AFD) and File Update Information (FUI)	127
8.8 File "Packages"	128
8.9 Finding More Information on File Archives	129
8.10 Notebook Archives	129
8.10.1 Setting Up Notebook Archives for a List	129
8.10.2 Migrating Old Notebook Archives to a New Site (LISTSERV to LISYSERV)	130
8.10.3 Migrating Old Notebook Archives (Non-LISYSERV to LISYSERV)	130
8.10.4 Deleting Old Notebook Archives	132
8.10.5 Indexing existing notebook archives	133
<b>Section 9 Creating and Editing Mail and Web Templates</b>	<b>135</b>
9.1 Using LISYSERV Templates	135
9.2 Getting Copies of the Default Template Files	136
9.3 Types of Templates	136
9.3.1 Mail Templates	136
9.3.2 Message Templates	136
9.3.3 Message Fragments	137
9.4 Naming Conventions for Message Templates and Fragments	137
9.5 Mail Template Format and Embedded Formatting Commands	138
9.5.1 Mail Template Format	138
9.5.2 Common Variable Substitutions	138
9.5.3 Template Commands	140
9.5.4 Conditional Processing	143
9.5.5 The .QUIF Command	145
9.5.6 Using 8-Bit Characters in Templates	145
9.6 Editing List-Level Default Templates	146
9.6.1 The INFO Template Form	146
9.6.2 DEFAULT MAILTPL Templates	147
9.6.3 Tips for Using Templates	148
9.7 Using the DAYSEQ(n) Function	149
9.7.1 Rotating Bottom Banner	149
9.7.2 Rotating FAQ via the PROBE1 Template and "Renewal= xx-Daily"	150
9.7.3 Calculating the Value for DAYSEQ()	150
9.8 Storing the <listname>.MAILTPL File on the Host Machine	151
9.9 DIGEST-H and INDEX-H Template Files	151
9.10 WWW Interface Templates and Template Forms	152
9.10.1 Web Forms (Static) Contained in DEFAULT MAILTPL	152

9.10.2 The WWW_ARCHIVE.MAILTPL File .....	152
9.10.3 The DEFAULT.WWWPTL File (Dynamic Templates) .....	153
9.10.4 The SITE.WWWTPL File .....	153
9.10.5 National Language Template Files (idiom.mailtpl) .....	154
9.10.6 Template Precedence .....	154
9.11 Serving Up Custom Web Pages for your List .....	155
9.11.1 A Practical Example: ADMIN_POST .....	155
9.12 Modifying the Output of LISTSERV's HELP Command (non-VM) .....	156
9.13 The \$SITE\$.MAILTPL File .....	158
<b>Section 10 Interpreting and Managing Log Files .....</b>	<b>159</b>
10.1 Logs Kept by LISTSERV .....	159
10.2 Managing the Logs .....	159
10.2.1 Making Daily Logs .....	159
10.2.2 Cleaning Your Log Files .....	159
10.3 Interpreting the LISTSERV log .....	160
10.3.1 Expiring Cookies .....	161
10.3.2 Releasing and Reallocating a Disk Slot .....	161
10.3.3 Reindexing a List .....	161
10.3.4 Distributing a Digest .....	161
10.3.5 Daily Error Monitoring Reports .....	162
10.3.6 Processing Mail for Local Lists .....	163
10.3.7 Administrative Mail (X-ADMMAIL) .....	163
10.3.8 DISTRIBUTE Jobs from Remote Hosts .....	164
10.3.9 Requesting "OK" Confirmation for Commands .....	164
10.3.10 Subscription Summary Updates (SUPD Jobs) .....	164
10.3.11 Global List of Lists Updates (LUPD Jobs) .....	165
10.3.12 Valid "OK" Confirmation Received .....	166
10.3.13 Invalid "OK" Confirmation Received .....	167
10.3.14 User Already Subscribed to a Given List .....	167
10.3.15 Non-Command Text in Mailings to LISTSERV .....	167
10.3.16 Response to List Owner or LISTSERV Maintainer Commands .....	168
10.3.17 Response to Posts to a Held List or to a List with PRIMETIME .....	168
10.3.18 Command Forwarded via GLX from Another Host .....	168
10.3.19 Netwide DELETE (X-DEL Jobs) .....	168
10.3.20 FIOC Cache Notifications .....	169
10.3.21 Web Archive/Administration Interface Logging .....	169
10.3.22 X-SPAM Jobs .....	170
10.3.23 X-TBREG Jobs .....	170
10.3.24 Responses to LVMON@VM.SE.LSOFT.COM .....	171
10.3.25 MIME Parser Messages .....	171
10.3.26 Content Filter Rejection Message .....	173
10.4 Interpreting the SMTP Logs (Windows Servers Only) .....	173
10.5 Interpreting the SMTP Worker Log Entries (Non-VM Only) .....	174
10.6 Change Logs .....	174
10.7 Using LISTSERV Logs and SHOW CTR to Extract Server Statistics .....	176
10.7.1 Sample Log-Processing Scripts .....	176



10.7.2 Interpreting the Output of SHOW CTR .....	179
10.8 Using the System Changelog to Track Distributions .....	181
10.9 Logging Changelog Information to a DBMS .....	182
<b>Section 11 Using the Web Administration Interface .....</b>	<b>185</b>
11.1 The Default LISTSERV Home Page .....	185
11.2 Logging In .....	185
11.3 Setting a LISTSERV Password .....	187
11.4 Logging In and Setting Preferences .....	187
11.5 List Maintenance via the Web Interface .....	189
11.5.1 Examining or Deleting a Subscription .....	189
11.5.2 Adding a New Subscriber to the List .....	190
11.5.3 Bulk Operations .....	191
11.6 List Configuration .....	192
11.7 Maintaining Mail and WWW Templates via the Web Interface .....	193
11.8 Sending Interactive Commands via the Web .....	194
11.9 Mail-Merge .....	194
11.10 Server Administration Interface .....	195
<b>Section 12 Distribution Features and Functions .....</b>	<b>197</b>
12.1 Controlling the Default Level of Acknowledgement to User Postings .....	197
12.2 Controlling the Maximum Number of Postings Per Day .....	197
12.2.1 Controlling Total Postings to the List Per Day .....	197
12.2.2 Controlling the Number of Postings Per Day from Individual Users .....	197
12.3 Controlling "Prime" Time .....	197
12.4 "Holding" and "Freeing" a List .....	199
12.4.1 Automatic List Holds .....	199
12.4.2 Manual List Holds .....	199
12.5 Controlling the List Digest Feature .....	200
12.6 Defining List Topics .....	200
12.7 Allowing/Blocking MIME Attachments .....	201
<b>Section 13 Error Handling Features and Functions .....</b>	<b>203</b>
13.1 Defining List-Level Error Handling Addresses .....	203
13.2 The Auto-Deletion Feature .....	203
13.3 LISTSERV's Loop Detection Feature .....	204
13.3.1 The Anti-Spamming Filter .....	205
13.4 RFC822 Mail Header Parsing .....	205
13.5 Address Probing .....	207
13.5.1 Active Address Probing .....	207
13.5.2 Passive Address Probing .....	208
13.5.3 OS-Specific Issues with Probing .....	209
13.6 Defining Server-Level Error Handling Addresses .....	209
13.6.1 BOUNCES_TO= .....	209
13.6.2 Crash Reports and CRASH_MONITOR= .....	209
<b>Section 14 List Maintenance and Moderation Features and Functions .....</b>	<b>211</b>
14.1 Setting Up Edited/Moderated Mailing Lists .....	211



14.2 Restricting the Size of Messages Posted to the List .....	212
14.3 Restricting the Number of Posts Per User Per Day .....	212
14.4 Moving a List to a New Location (New-List= Keyword) .....	213
<b>Section 15 Security Features and Functions .....</b>	<b>215</b>
15.1 The VALIDATE= Keyword .....	215
15.2 Controlling Subscription Requests .....	216
15.3 Controlling the Service Area of the List .....	216
15.4 Controlling Who Reviews the List of Subscribers .....	217
15.5 Controlling Access to the Notebook Files .....	217
15.6 Controlling Who Can Post Mail to a List .....	218
15.7 The "OK" Confirmation Mechanism .....	219
15.7.1 Explicitly Cancelling "OK" Cookies .....	221
15.8 Denying Service to Problem Users .....	221
15.8.1 The "Filter=" List Header Keyword .....	221
15.8.2 The "FILTER_ALSO" Configuration File Variable .....	221
15.8.3 The "SERVE" Command .....	221
15.8.4 The POST_FILTER List Exit Point .....	222
15.9 Hiding Selected Header Lines .....	222
15.10 Tracking Subscription Changes with the Change-Log Keyword .....	223
<b>Section 16 Subscription Features and Functions .....</b>	<b>225</b>
16.1 Setting Up Subscription Confirmation .....	225
16.2 Defining Default Options for Subscribers at Subscription Time .....	225
16.3 Setting Up Subscription Renewal .....	226
<b>Section 17 Other Features and Functions .....</b>	<b>229</b>
17.1 Setting Up National Language Mail Templates .....	229
17.2 Translating Control Characters Included in List Mail .....	229
17.3 Communicating with List Owners .....	229
17.3.1 The Listname-REQUEST Alias .....	230
17.3.2 The ALL-REQUEST Alias .....	230
17.3.3 Required Configuration for Unix and VMS Servers Running PMDF .....	231
17.3.4 Other Aliases Used by LISTSERV .....	231
<b>Section 18 Special Functionality for ISP's .....</b>	<b>233</b>
18.1 Directory Quotas for Individual Lists .....	233
18.1.1 The QUOTA.FILE .....	233
18.1.2 Displaying Quota Information .....	233
18.1.3 Reloading Quota Information After Making Changes .....	234
18.2 Limiting the Number of Subscribers to a List .....	234
<b>Appendix A: Command Reference Card .....</b>	<b>235</b>
<b>Appendix B: Sample Boilerplate Files .....</b>	<b>249</b>
Subscription Requests Sent to the List .....	249
Sending Other Commands to the List or to the *-REQUEST Address for the List .....	249
Unsubscribed User Still Getting Mail .....	250
Quoted Replies Include Message Headers Causing them to Bounce .....	250
Delivery Error with Unknown User Account .....	251

Setting a User to DIGEST because of Bouncing Mail ..... 251  
A Sample "Your List has been Created" Boilerplate ..... 252  
**Index** ..... **253**

---

## List of Figures

Sample List Header .....	75
Sample List Header File for a List Called MYLIST .....	87
The Edited List Header File Ready to be Sent Back to the Server .....	87
Sample Filelist Retrieved with (CTL Option) .....	119
Adding a File Descriptor to the Filelist .....	120
Default Content of the INFO Template form for DEFAULT.MAILTPL .....	146
Sample of an Edited INFO Template Form .....	147
Typical Contents of a DIGEST-H or INDEX-H File .....	151
Sample DIGEST Output for a List with a DIGEST-H File .....	152
Sample of a CLEANLOG.REXX Script .....	160
Typical SMTP Log for SMTPL.EXE Listener .....	173
Typical SMTPS Log for the SMTPW.EXE SMTP "Workers" .....	174
LISTSERV Archives Screen .....	186
Login Screen .....	186
Registering a LISTSERV Password .....	187
Web Interface Toolbar .....	187
Preference Screen .....	188
Examining or Deleting a Subscription .....	189
Select Subscriber to Examine or Delete .....	189
Subscriber Management Screen .....	190
Adding a New Subscriber .....	191
The Bulk Operations Tab .....	192
The List Management Menu .....	192
The Administration Dashboard .....	196
A Typical Daily Error Monitoring Report .....	204
Sample RFC822 Parser Error .....	206
The Editor-Header for a List Set to Send= Editor, Hold .....	218
A Typical Command Confirmation Request .....	219
Typical Daily Subscription Renewal Monitoring Report .....	227
Typical Output of a SHOW QUOTA Command Issued by a Privileged User .....	234



## List of Tables

LISTSERV Classic Keywords disabled in LISTSERV Lite .....	4
LISTSERV Lite vs. LISTSERV Classic .....	4
Site Configuration Files .....	13
LISTSERV Site Configuration Variables .....	14
Creating Archive and Notebook Directories .....	43
Sample Output of an <i>INDEX Listname</i> Command .....	56
Formatting Commands for List Owners .....	140
Advanced Formatting Commands for List Owners .....	141



## Preface - About This Manual

Every effort has been made to ensure that this document is an accurate representation of the functionality of LISTSERV®. As with every software application, development continues after the documentation has gone to press so small inconsistencies may occur. We would appreciate any feedback on this manual. Send comments via email to:

[MANUALS@LSOFT.COM](mailto:MANUALS@LSOFT.COM)

The following documentation conventions have been used in this manual:

- Menus, options, icons, fields, and text boxes on the screen will be bold (e.g. the **Help** icon).
- Clickable buttons will be bold and within brackets (e.g. the **[OK]** button).
- Clickable links will be bold and underlined (e.g. the **Edit** link).
- Directory names, commands, and examples of editing program files will appear in Courier New font.
- Some screen captures have been cropped for emphasis or descriptive purposes.
- This symbol denotes an important note or warning.



- This symbol denotes optional advice that can help you save time.

## Editorial Note - New Version Numbering

With this release, L-Soft is aligning LISTSERV's version numbering with the rest of the e-mail industry. There have been 51 released versions of LISTSERV since 1986 – 15 major upgrades and 36 minor releases. Version 1.8e in the “traditional” numbering system corresponds to 14.0. The present update is version 15.0.

Because the old nomenclature is more familiar to our users, in this version of the documentation we will continue to refer to versions of LISTSERV inferior to version 14.4 by the old version system.

## LISTSERV Command Syntax Conventions

Generally, parameters used in this document can consist of 1 to 8 characters from the following set:

A-Z 0-9 \$#@+-\_:

Deviations from this include:

- *fformat* – Netdata, Card, Disk, Punch, LPunch, UUencode, XXencode, VMSdump, MIME/text, MIME/Appl, Mail.
- *full\_name* – *first\_name* [*middle\_initial*] *surname* (not your email address). Must consist of at least two space-separated words, e.g., "John Doe".
- *listname* – name of an existing list



- *node* – Either the fully-qualified domain name (FQDN) of an Internet host or the BITNET nodeid or Internet hostname of a BITNET machine which has taken care of supplying an 'internet' tag in its BITEARN NODES entry.
- *host* – Generally the same as *node*, but normally refers specifically to the fully-qualified domain name (FQDN) of an Internet host rather than to a BITNET nodeid.
- *pw* – a password containing characters from the set: A-Z 0-9 \$#@\_ -?!|%
- *userid* – Any valid RFC822 network address not longer than 80 characters; if omitted, the 'hostname' part defaults to that of the command originator.
- *internet\_address* – Similar to *userid*, but specifically refers to a complete RFC822 network address in *userid@fqdn* format. When we use this nomenclature a fully-qualified hostname is required.

Other deviations from the standard set will be noted along with the affected commands.

Also, the following conventions represent variable or optional parameters:

- *italic type* – Always indicates required parameter names that must be replaced by appropriate data when sending commands to LISTSERV.
- *< >* – Angle brackets may sometimes enclose required parameter names that must be replaced by appropriate data when sending commands to LISTSERV. Sometimes used for clarity when italic type is inappropriate.
- *[ ]* – Square brackets enclose optional parameters which, if used, must be replaced by appropriate data when sending commands to LISTSERV.

## Contacting L-Soft

### **Support**

L-Soft international recognizes that the information in this manual and the FAQ questions on our web site (<http://www.lsoft.com/lsv-faq.html>) are not going to solve every problem you may face. We are always willing to help diagnose and correct problems you may be having with your licensed LISTSERV server.

L-Soft strongly recommends that, for support purposes, it is best to use the technical support "lifebuoy" link from the Server Administration Dashboard to initiate a support ticket. This will help you create an email message to the support group that contains all the necessary information about the site configuration, license and so forth without requiring you to find the individual files or issue information commands.

If LISTSERV is not running, of course, this will not be possible. In that case, please try to use the following procedure:

- Make the subject line of your report indicative of the problem. L-Soft receives a great deal of mail with the subject "Help!", which is not very helpful when we receive them.
- Include any appropriate log entries. LISTSERV keeps logs of everything it does, and without the log trace back, it is often impossible to determine what caused a given error.

- If you're running a Unix server and LISTSERV dumps core, please run the debugger on the core file, produce a trace back, and include the results.
- Always send a copy of your site configuration files (with the passwords x'ed out). See Section 5.1 [Site Configuration Files](#) for the locations and names of the two site configuration files.
- Send along anything else that you think might be helpful in diagnosing the problem.

If the supporting documents (for instance, log files) are extremely large, please contact support first before sending everything through. The support group has alternative methods of handling large files that they will be happy to share with you.

If you are not currently an L-Soft customer and are running an evaluation version of our software, please send your trouble reports to the evaluation users' list, [LISTSERV@PEACH.EASE.LSOFT.COM](mailto:LISTSERV@PEACH.EASE.LSOFT.COM).

If you are running LISTSERV Lite, please send your trouble reports to the LISTSERV Lite support mailing list, [LISTSERV-LITE@PEACH.EASE.LSOFT.COM](mailto:LISTSERV-LITE@PEACH.EASE.LSOFT.COM). This includes users of the paid version of the software unless you have also purchased paid support.

If your LISTSERV Classic/Classic HPO server for VM, VMS, unix, or Windows has paid-up maintenance, you may send problems to [SUPPORT@LSOFT.COM](mailto:SUPPORT@LSOFT.COM) for a quick reply.

### **Sales**

To reach our worldwide sales group, simply write to [SALES@LSOFT.COM](mailto:SALES@LSOFT.COM). You may also call 1-800-399-5449 (in the US and Canada) or +1 301-731-0440 (outside the US and Canada) to speak to our sales representatives.



## Section 1 Who Should Read This Book?

This manual makes the following assumptions:

- You are a system administrator of a VM, VMS, unix (including MacOS), Windows 2000/XP/2003 or system (or in any case, a person with root- or system-level administrative privileges) whose assignment it is to be the LISTSERV maintainer;
- You have already installed the current version of L-Soft's LISTSERV on your system in accordance with the installation instructions that come with the package, and have it running;
- You have sufficient knowledge (or know where to find it) of your system mailer to fine-tune it without needing instructions from this manual.

In other words, we expect you already to be knowledgeable about the system on which you plan to install and run LISTSERV. This manual does not contain installation instructions; individual installation guides for the four general types of operating systems supported by L-Soft can be found at <http://www.lsoft.com/resources/manuals.asp>.

L-Soft international's LISTSERV software is designed to run on various platforms that have widely-differing configurations. Therefore it is not within the scope of this manual to describe in detail (for instance) how you can tune Sendmail 8.12.3 under Linux for optimum performance with LISTSERV. However, general tips that could work on all systems will be offered within these pages.

Overall you will find that LISTSERV works much the same way on a unix workstation or a VMS minicomputer or an Intel Pentium machine running Windows 2000 as it has since 1986 on VM mainframes. Where LISTSERV procedures do differ between platforms, we will detail those differences in order to minimize confusion.

### 1.1 Where to Find Changes and Updates to the Manual?

When we find a mistake in the manual, or when between-release features are added, we normally report changes to the manual to the announce-only mailing list [LISTSERV-DEVELOPERS@PEACH.EASE.LSOFT.COM](mailto:LISTSERV-DEVELOPERS@PEACH.EASE.LSOFT.COM), provided as a free service for our customers.



## Section 2 Differences Between Architectures and Implementations

This section outlines differences between how LISTSERV is implemented on VM and non-VM machines, and the differences between LISTSERV and LISTSERV Lite.

### 2.1 Differences Between Architectures and Implementations

In version 15, LISTSERV running under VM continues to differ in some regards from its counterparts on the other architectures. Here is a short list of these differences:

- VM: The web interface is not available.
- VM: Rotating change-logs are not available.
- Non-VM: Only a subset of the VM file server functions are available
- Non-VM: Certain rarely-used commands (e.g., STATS listname) are not available
- Non-VM: FUI (File Update Information) and AFD (Automatic File Distribution) are not available



**Note:** LISTSERV 15 running on non-VM systems actually has about 98% of the functionality of the VM version, and nearly 100% of the functionality that people actually use day-to-day.

#### 2.1.1 The File Server

There are actually two different file server systems in operation across the LISTSERV network. One is the original version running on VM, which includes the ability to create "filelists" (indexes) which point in turn to more files which can be stored on the server, and the AFD and FUI functions mentioned above. This file server system, while still quite powerful and easy to use, is unfortunately written in a non-portable language, making a complete rewrite from the beginning a necessity. There has been no change in the VM file server from 1.8b through 1.8e (and subsequently 15.x).

The second file server system currently in operation runs on the VMS, unix, and Windows ports of LISTSERV. This is in essence still a subset of the old system in which the LISTSERV maintainer creates entries in a SITE.CATALOG file for each file that will be made available to users. It is also possible for the LISTSERV maintainer to create hierarchical sub-catalogs, which can be maintained by list owners or other responsible people. Finally, the GIVE command and the ability to create file "packages" are also available. For more information, please see Section 8 [File and Notebook Archives](#).

L-Soft is still developing LISTSERV's file server, which will eventually include a super-set of the original VM file server command set. Pains are being taken to ensure that the most common commands will not change along the development path. This will help to keep a great deal of existing documentation that has been passed along the Internet from becoming obsolete overnight.

#### 2.1.2 The WWW List Archive and List Management Interface

LISTSERV 15 includes a significantly-rewritten web archive and management interface. The web interface continues to be unavailable on VM but is available on all other platforms on which the software runs.

## 2.2 Differences between LISTSERV and LISTSERV Lite

LISTSERV Lite is LISTSERV running with a special license activation key (LAK) which limits what you can do with the software. With the Free Edition of LISTSERV Lite (activated by a LAK which is both free and perpetual), you can run up to 10 mailing lists as long as you do not derive a profit from this activity. You can also purchase LISTSERV Lite LAKs that allow more (or unlimited) lists.

However, note carefully that LISTSERV Lite does not have all of the functionality of the full, Classic version--a list of the keywords and functions disabled in LISTSERV Lite follows this paragraph. For more information on the exact terms and conditions under which you may run LISTSERV Lite, please see L-Soft's World Wide Web site or contact L-Soft's sales department.

Table 2-1 LISTSERV Classic Keywords disabled in LISTSERV Lite

Change-Log	Confirm-Delay	DBMS	Default-Topics
Editor-Header	Exit	Files	Indent
Internet-Via	Language	List-Address	List-ID
Local	Long-Lines	Loopcheck	Mail-Merge
Mail-Via	Misc-Options	Moderator	New-List
Newsgroup	NJE-Via	Peers	Prime
Renewal	Sender	Service	Sizelim
Stats	Sub-Lists	Topics	X-Tags



**Note:** The fact that the keyword is disabled only means that the default value cannot be changed. For instance, loop checking is still present, you just cannot control the details of its operation. On the other hand, if the default value is that the function in question is disabled (as is the case with "Peers="), then the function is actually gone. See the [List Keyword Reference](#) document for more information.

Table 2-2 LISTSERV Lite vs. LISTSERV Classic

Feature	LISTSERV Classic	LISTSERV Lite
Moderated Lists	Yes	Yes
Moderation Sharing	Yes	No
DISTRIBUTE	Yes	No
Peered Lists	Yes	No
Topics (up to 23 different topics per list)	Yes	No
Validate Keyword (provides security)	Yes	Yes
Filter Keyword (screens mail)	Yes	Yes
Spam Detector	Yes	Yes



Feature	LISTSERV Classic	LISTSERV Lite
Spam Filter	Yes	No
Customization of Mail Templates	List based	Site based
Auto-Delete	Yes, full featured	Yes, not full featured
Probe	Yes(*)	No
List Exits	Yes	No
Networked Mode	Yes	(**)
Subscription Options: - RENEW - EDITOR - REVIEW - NOPOST - All other LISTSERV subscription options	Yes Yes Yes Yes Yes	No No No No Yes
File Server Functions	Yes, hierarchical	Yes, non-hierarchical
Database (Archive Search) Functions	Yes	No
WWW Archive Interface	Yes, with search interface	Yes, but no search functions
WWW Administration Interface	Yes	Yes
DBMS/Mail Merge Functions	Yes	No
Anti-Virus Scanning Feature (Windows NT/2000 and Linux Only)	Yes (requires special LAK and special AV software package; contact your sales representative for details.)	No
Message Content Filtering	Yes	No

(\*) The probe feature does not work with all MTAs (mail servers), or may only work with recent enough versions.

(\*\*) Networked and Standalone RUNMODEs are not available in the Free Edition of LISTSERV Lite, but are available in the commercial version of LISTSERV Lite.

### 2.3 Operating Systems and Architectures Supported

A comprehensive list of operating systems (and versions) under which LISTSERV is supported can be found at:

<http://www.lsoft.com/products/default.asp?item=listserv-ossupport>



## Section 3 Principles of Operation

LISTSERV® is software that allows you to create, manage, and control electronic "mailing lists" on a corporate network or on the Internet. Since its inception in 1986 for IBM mainframes on the BITNET academic network, LISERV has been continually improved and expanded to become the predominant system in use today. LISERV is now available for VM, OpenVMS, several types of unix, and the Microsoft Windows "family" (including Windows 2000 and later).

Consider for a moment what the users of your electronic mail system actually use electronic mail for. Do they discuss problems and issues that face your organization, down to the departmental level? In an academic setting, do your faculty and students communicate via electronic mail? As with "real world" distribution lists, electronic mailing lists can make it possible for people to confer in a painless manner via the written word. The electronic mail software simply replaces the copying machine, with its associated costs, delays and frustrations. In fact, electronic mail lists are easier to use than most modern copiers, and a lot less likely to jam at just the worst possible moment.

Because electronic mail is delivered in a matter of seconds, or occasionally minutes, electronic mailing lists can do a lot more than supplement the traditional paper distribution lists. In some cases, an electronic mailing list can replace a conference call. Even when a conference call is more suitable, the electronic mailing list can prove a powerful tool for the distribution of papers, figures and other material needed in preparation for the conference call. And, when the call is over, it can be used to distribute a summary of the discussion and the decisions that were made. What before might have been an exchange of views between two or three people can now become an ongoing conference on the issue or problem at hand. Announcement lists and even refereed electronic journals can be made available to your audience, which can be as small as a few people or as large as the entire Internet community.

LISERV accomplishes its design goals very efficiently and very quickly. This is due primarily to its use of the proprietary DISTRIBUTE algorithm (described in RFC1429, and in the [Advanced Topics Guide for LISERV](#), available separately) and to the large (and growing) network of LISERV servers.

The LISERV network of servers helps to enhance LISERV's performance by providing a "backbone" through which large quantities of mail can be quickly distributed. The backbone also allows LISERV servers to "talk" to each other and exchange information. Among other things, this exchange of information between servers allows your own local server to participate in the global List of Lists service and L-Soft's [CataList](#) service on the World Wide Web.

LISERV's nature as a distributed network of interconnected servers also makes it possible to identify and eliminate unsolicited advertisements sent to multiple lists (known as "spam") before they do much harm. While it is virtually impossible for a small isolated server to detect spam (unless the sender listed the thousands of lists he was targeting in the "To:" field), the LISERV network receives thousands of copies of the spam. By comparing notes with each other, the servers can quickly determine that a spam is occurring and raise a network-wide "spamming alert", stopping the message before it does much damage at all. Since the introduction of LISERV's anti-spam technology, the growing number of sites that are participating in the anti-spamming warnings have virtually stopped the distribution of such messages in their tracks. L-Soft is constantly upgrading and refining the anti-spam algorithms.

In addition to the anti-spamming filter, LISTSERV also incorporates an anti-spoofing filter, to keep mischievous (and often malicious) users from subscribing other users to mailing lists in order to "mailbomb" them.

LISTSERV makes it possible for you to offer the same mailing list in four different formats:

- **Individual mail messages** sent out as they are processed
- **Digest mode**, where a compendium of messages processed by the list is sent at specified intervals
- **Indexed mode**, where an index consisting of the message number, sender, and the subject line of each message is sent each day, along with instructions on how to retrieve postings from the server
- Users can read, search, and respond to postings via LISTSERV's **Web Archive Interface**.

These modes are set by sending SET commands to LISTSERV. Unlike some other mailing list management systems, LISTSERV does not require the user to unsubscribe from one version of the list and resubscribe to another just to change delivery modes.

LISTSERV includes database search capability for list archive notebooks. A fast reverse indexing feature is available for servers running lists with large archives. Users can use a simple search syntax to comb list archives for specific terms of interest. And L-Soft provides a World Wide Web archive interface (not currently available on VM for technical reasons unrelated to LISTSERV itself) with which the notebook archives for all public lists can be viewed and searched from a web browser. LISTSERV's WWW interface differs from (and has advantages over) "hypermail" style web archiving in that new postings are shown as soon as they are received; postings can be organized in a manner that best suits the reader; there is no duplication of effort, as the LISTSERV WWW interface works from the list's notebook archives rather than creating a separate HTML file for each posting; and the list owner can customize the main page for their list by simply modifying their mail template file.

LISTSERV also includes a number of list and server management functions in its WWW interface, including the ability to edit list headers and associated mail and WWW templates, and to manage subscribers via the Web. Almost every aspect of LISTSERV management, from the list subscriber level to the site administrator level, is available through the version 15 interface. See Section 11 [Using the Web Administration Interface](#) for details.

LISTSERV has contains DBMS and mail-merge support. These features are documented in the Advanced Topics Guide for LISTSERV, available separately. There is also much tighter tight integration between LISTSERV and L-Soft's campaign management software, LISTSERV Maestro, beginning with LISTSERV 15.0 and Maestro 3.0.

LISTSERV includes an Anti-Virus Scanning feature for messages passing through the server. This is a value-added enhancement which requires a special LAK and a special version of F-Secure Anti-Virus. At this writing the feature is only available for Windows 2000 and later, and Linux servers running LISTSERV Classic or LISTSERV Classic HPO. (Other OS platforms may be supported in the future; there is no intent to make this functionality available in the Lite version of the product.)

Many other enhancements have been introduced in LISTSERV 15.0. The full release notes are available at <http://www.lsoft.com/resources/manuals.asp>.

## Section 4 A LISTSERV How-To for Site Managers

This section is not intended to replace the LISTSERV FAQs available from [L-Soft's documentation web site](#); instead, it is an attempt to bring together certain basic operations and how to accomplish them in one place. However, note that some of these how-to answers will redirect you to existing external documentation or to other sections of this manual in order to avoid duplication of effort.

### 4.1 Installation and Startup Questions

- **How do I install LISTSERV?**

Installation guides are available on the web and are also shipped in the version-specific installation kits. You can read the guides on the web at:

<http://www.lsoft.com/resources/manuals.asp>

- **Why do I need a DNS A record and a static IP number for my LISTSERV machine?**

The best analogy is to consider why you need to put a return address on a piece of postal mail that you expect someone to respond to (or to be returned to you if the person you are trying to reach no longer lives at the address you have for him). In order for people to be able to send mail to your server, it must have a "street address" so that the "postman" can deliver mail to it, and that "street address" must be known to the "post office" so that mail can be properly routed. The DNS A record tells the world where your LISTSERV machine is located by both its name (e.g. LISTSERV.EXAMPLE.COM) and its IP address, so that other mail machines on the Internet can correctly route mail to it. If the IP address is not static, in other words if it changes every time you dial up, or whenever you disconnect and reconnect your DSL service, it is not possible to add an A record for it to DNS. This is why both a static IP address and a DNS A record are required in order for LISTSERV to work properly.

- **Is there any way to make LISTSERV work on a machine with a dynamic IP?**

Yes, if you subscribe to a service that provides dynamic DNS. There are several such services, some of them free. However, the configuration of dynamic DNS with your particular installation is outside of the scope of this manual, and L-Soft cannot help you with it.

- **Can LISTSERV read mail from POP mailboxes?**

No. LISTSERV is designed to work with SMTP mail servers and is not able to read POP mailboxes.

- **How do I install the web archive/administration interface?**

Please see either your version-specific installation guide or Section 5.4 [Installing & Configuring LISTSERV's WWW Archive & Administration Interface](#).

- **How do I start LISTSERV?**

This is version specific and documented in the version-specific installation guides.

- **How do I stop or stop/restart LISTSERV?**

LISTSERV site administrators can issue the STOP command from the "Issue a LISTSERV Command" page of the web administration interface.

Starting with LISTSERV 15.0, it is possible under all ports of the software to immediately restart LISTSERV in a similar manner (for instance, to pick up configuration changes that require a restart) by issuing the STOP REBOOT command.

Alternate supported methods are to send e-mail from your POSTMASTER= address to LISTSERV@your server with the command

```
STOP PW=personal_password
```

or

```
STOP REBOOT PW=personal_password
```

in the body of the message, where "personal\_password" is the personal password you have created for yourself for LISTSERV administration.<sup>1</sup>

Under Windows, assuming that LISTSERV is running as a system service (which is the recommended method), you can also stop LISTSERV from the Control Panel/ Services applet, or by issuing a NET STOP LISTSERV command from a DOS prompt (both of these assume that you are logged into the machine with administrative privileges).

Under unix, it is possible to stop LISTSERV by issuing a 'kill -TERM' command on the PID found in \$LSVSPool/listserv.PID. However, this is not 100% guaranteed to kill all of the existing 'lsv' processes which may be running at the time (for instance you may end up with zombie processes left over from web interface queries), so L-Soft recommends that the e-mail method using the STOP command as documented above be used in preference to the 'kill -TERM' method from a shell prompt. It is vital that all 'lsv' processes be stopped before restarting LISTSERV, as the web interface may not properly re-initialize if this is not done.

LISTSERV also stops automatically when the system is rebooted, and depending on platform support and whether or not you have the system set to do so, may also be started automatically at boot time.

## 4.2 Initial Configuration

- **How do I add, change, and delete LISTSERV Maintainers (aka postmasters)?**

LISTSERV Maintainers are defined by their e-mail addresses in the site configuration file, by setting the site configuration variable POSTMASTER=. This is normally done by opening the site configuration file in a text editor (never in a word

---

1. It should be noted that STOP and SHUTDOWN are synonymous, and REBOOT and REIPL may also be used interchangeably. Therefore, STOP REBOOT, STOP REIPL, SHUTDOWN REBOOT and SHUTDOWN REIPL all have identical outcomes. Please also note that REBOOT in this context means to restart LISTSERV, not to reboot the entire machine.

processor or other non-flat-ASCII editor) and changing the value in the variable, then saving the file and stopping and restarting LISTSERV.

Windows sites can alternatively use the SITE.EXE configuration GUI to make these changes, but must also stop and restart LISTSERV after making the change.



**Note:** The syntax for the `POSTMASTER=` variable (like all other site configuration variables) differs from one OS platform to another. See the [Site Configuration Keyword Reference](#) document for examples.

- **How do I create passwords for postmasters, and what are they used for?**

All commands are authenticated by the personal password associated with the LISTSERV maintainer's e-mail address. This password can be created in one of three ways:

- Via the web interface, where a clickable password creation link will appear when you try to log in for the first time; or
- Via mail, by using the `PW ADD` command documented elsewhere in this manual.
- By another LISTSERV maintainer, using the `PWC ADD` command documented elsewhere in this manual.



**Note:** Some mailing list commands do not always require password authentication, depending on the setting of the `Validate=` list header keyword for the list in question. See the [List Keyword Reference](#) document for more information on how the various `Validate=` settings affect command authentication.

- **How do I make my first list?**

Please see Section 7.1 [Basic List Creation](#).

- **How do I delete a list?**

LISTSERV maintainers can delete lists from the web administration interface.

- **Does LISTSERV have a GUI interface?**

LISTSERV's GUI interface is its web administration/archive interface. Most site-level and most list-level functions can be accessed via the web interface.





## Section 5 Configuring Your LISTSERV Site



**Note:** This manual is not intended to replace the individual installation manuals for LISTSERV on the various platforms supported by L-Soft. This is because the installation procedures vary radically from platform to platform and this manual is intended to assist LISTSERV maintainers on operational LISTSERV sites. The installation guides for all platforms are included in the software distributions, and are also available on L-Soft's World Wide Web and FTP sites.

For the purposes of this section, it is assumed that you have already installed LISTSERV on your host computer and have been able to start it in successfully in interactive mode.

### 5.1 Site Configuration Files

These files have different names depending on the platform. They are located in the same directory with the executable binaries.

*Table 5-1 Site Configuration Files*

Platform	Site Configuration File
VM	LOCAL SYSVARS
Open VMS	SITE_CONFIG.DAT
Unix (all)	go.user
Windows NT/2000	SITE.CFG
Windows 95/98/Me	SITE.CFG

These are the only configuration files that should be changed on any LISTSERV installation. Software upgrades may overwrite any other configuration files located in LISTSERV's home directory. They will never overwrite the files listed above. The intent is to help preserve your system settings from one version to the next so that you do not experience the inconvenience of having to reconfigure LISTSERV after an upgrade.

L-Soft international, Inc., is not responsible for system downtime or mis-operation occasioned by the loss of any changes that you make to configuration files other than the ones listed above.

All LISTSERV sites that use the web administration interface to modify the system configuration in real-time will also have a file called SITECFG.FILE (sitecfg.file under unix) in LISTSERV's A directory. This is where LISTSERV stores changes to the default or initial configuration when made via the web interface. This file SHOULD NOT be modified by hand.

At startup, LISTSERV 15.0 and following will first read the original site configuration file, and then will read the SITECFG.FILE for any changes to the original configuration.

L-Soft strongly recommends that, for support purposes, it is best to use the technical support "lifebuoy" link from the Server Administration Dashboard to initiate a support ticket. This will help you create an email message to the support group that contains all the necessary information about the site configuration without having to go to the trouble to find and attach both configuration files.

If LISTSERV is not running, of course, this will not be possible, and you should provide both the original configuration file and SITECFG.FILE to support when writing.

## 5.2 What can be configured?

Depending on the platform, a large number of control variables are available to "fine tune" the performance and behavior of LISTSERV. The following table indicates the variables, under which platforms they are supported, and briefly what they control. Please see the [Site Configuration Keyword Reference](#) document for details before setting any control variable. Some variables shown in the table are VM legacy settings and are not otherwise discussed in this manual.

Table 5-2 LISTSERV Site Configuration Variables

Variable	Short Description	VM	VMS	Unix	Win
ALL_REQUEST_ALLOWED_USERS	Specifies non-POSTMASTER users who are allowed to post to the ALL-REQUEST alias.	Yes	Yes	Yes	Yes
ANTI_VIRUS	Boolean value determining whether or not LISTSERV's anti-virus scanning is enabled.	Yes	Yes	Yes	Yes
AVS_DROP_SPAM	One of two variables that may be set on servers that submit mail to the external LISTSERV AVS for virus and spam scanning, which can help deal with DoS mail-bombing attacks.	Yes	Yes	Yes	Yes
AVS_DROP_VIRUS	One of two variables that may be set on servers that submit mail to the external LISTSERV AVS for virus and spam scanning, which can help deal with DoS mail-bombing attacks.	Yes	Yes	Yes	Yes
BITNET_ROUTE	Defines the hostname of a machine that knows how to route mail to BITNET addresses.	Yes	Yes	Yes	Yes
BOUNCES_TO	Tells LISTSERV where to send bounces not related to any particular list.	Yes	Yes	Yes	Yes
CHANGELOG_DBMS	Tells LISTSERV to mirror a copy of all or selected changelogs to DBMS.	Yes	Yes	Yes	Yes
CHANGELOG_DBMS_CONNECTION	Used in conjunction with CHANGELOG_DBMS. Sets the DBMS connection characteristics for mirroring changelog data to DBMS.				

Variable	Short Description	VM	VMS	Unix	Win
CHANGELOG_DBMS_TABLE	Used in conjunction with CHANGELOG_DBMS. Sets the DBMS table characteristics for mirroring changelog data to DBMS.				
CHECKMDISK	List of library mini-disks to be checked at startup.	Yes	No	No	No
CLI_*	Three configuration variables under the CLI_* rubric are available for use with DBMS/ Mail Merge. Please see the <a href="#">Advanced Topics Guide for LISTSERV</a> for more information.	No	No	Yes (AIX only)	No
CMDPIP_HOSTNAME	Defines the hostname used by the LCMD utility.	No	No	No	Yes
CMSNAME	The name of the CMS system to be used on IPL commands.	Yes	No	No	No
CRASH_MONITOR	Where to send VMS or NT crash reports.	No	Yes	No	Yes
CREATEPW	The password required to create new lists.	Yes	Yes	Yes	Yes
DATABASE	Indicates whether the (old) VM database functions are enabled or not.	Yes	No	No	No
DBMS_*	Several configuration variables under the DBMS_* rubric are available for use with the DBMS/ Mail Merge.	no	Yes	Yes <sup>a</sup>	Yes
DBRINDEX	Determines whether or not the new LISTSERV database functions use reverse indexing.	Yes	Yes	Yes	Yes
DEBUG_LOG_TCPGUI	Show fully-logged TCPGUI commands for debugging purposes.	No	Yes	Yes	Yes
DEBUG_SHOW_PW	Debug setting to display passwords in LISTSERV's log file.	No	Yes	Yes	Yes
DEFAULT_CHANGELOG_PERIOD	Sets a default period for rotating change-logs.	no	Yes	Yes	Yes
DEFAULT_DIST_BACKGROUND	Boolean value that sets the server default for background DISTRIBUTE (requires HPO).	Yes	Yes	Yes	Yes
DEFAULT_LANGUAGE	Sets the default national language template for use by all lists on the server.	Yes	Yes	Yes	Yes

Variable	Short Description	VM	VMS	Unix	Win
DEFAULT_LOOPCHECK	Sets the server default for the list-level Loopcheck keyword.	Yes	Yes	Yes	Yes
DEFAULT_MISC_OPTIONS	Sets the server default for the list-level Misc-Options keyword.	Yes	Yes	Yes	Yes
DEFAULT_PROBE	Sets default for passive probing.	Yes	Yes	Yes	Yes
DEFAULT_SPLIT	Provides a default value for the SPLIT= command line keyword.	Yes	Yes	Yes	Yes
DELAY	The delay between two reader-scan operations.	Yes	No	No	No
DIAGD4	Indicates whether LISTSERV should use diagnose X'D4' to mimic the RSCS origin on files it DISTRIBUTEs.	Yes	No	No	No
DIST_ALLOWED_USERS	Space-separated list of non-POSTMASTER users who are to be allowed to use DISTRIBUTE.	Yes	Yes	Yes	Yes
DIST_BACKGROUND_CHUNKSIZE	Tuning parameter for background DISTRIBUTE. (requires HPO)	Yes	Yes	Yes	Yes
DIST_BACKGROUND_JOB_SIZE	Tuning parameter for background DISTRIBUTE. (requires HPO)	Yes	Yes	Yes	Yes
DIST_FORWARD	Enables and tunes DISTRIBUTE "workers". This is a tuning feature for embedded mail merge. (requires HPO)	Yes	Yes	Yes	Yes
DIST_FORWARD_MIN_SIZE	The minimum message size, in kilobytes, to make use of DISTRIBUTE workers (requires HPO)	Yes	Yes	Yes	Yes
DIST_FORWARD_THRESHOLD	Enables and tunes DISTRIBUTE "workers". This is a tuning feature for embedded mail merge. (requires HPO)	Yes	Yes	Yes	Yes
DIST_OWNER_MAIL_MERGE	Determines whether or not list owners may use the mail-merge feature for their lists.	Yes	Yes	Yes	Yes
DIST_SECURITY	Boolean value which controls security validation feature for the DISTRIBUTE command. WARNING: See the <a href="#">Site Configuration Keyword Reference</a> document before changing this value.	Yes	Yes	Yes	Yes

Variable	Short Description	VM	VMS	Unix	Win
DKIM_SIGN	Specifies DomainKeys domains for which you are able and willing to sign	Yes	Yes	Yes	Yes
DKIM_SIGN_ALL	Boolean value which directs LISTSERV to sign all messages for which it has a suitable DomainKeys private key.	Yes	Yes	Yes	Yes
EMBEDDED_MAIL_MERGE	Boolean value which determines whether or not LISTSERV may use an SMTP mailer other than LSMTP to send mail-merge messages. The default is 1, which means LSMTP is not required.	Yes	Yes	Yes	Yes
FILEDISK	The filemode of the DEFAULT disk to be used for storing files via a PUT command.	Yes	No	No	No
FILEMAXL	The maximum number of lines for any incoming non-mail file to be accepted.	Yes	Yes	Yes	Yes
FILTER_ALLOW	Defines users or classes of users who should be exempt from LISTSERV's standard filter.	Yes	Yes	Yes	Yes
FILTER_ALSO	Defines users or classes of users who should not be allowed to post to any list on the server.	Yes	Yes	Yes	Yes
FIOC_INUSE_RETRY	Defines the number of seconds LISTSERV will try to open a file locked by an external process.	Yes	Yes	No	Yes
FIOC_TARGET	Defines (in kilobytes) the "target size" for LISTSERV's file cache.	Yes	Yes	Yes	Yes
FIOC_TRIM	Defines the threshold (in kilobytes) at which point LISTSERV should start trimming the cache.	Yes	Yes	Yes	Yes
FIOC_WARNING	Defines (in kilobytes) the cache size at which LISTSERV should write a warning to the console log.	Yes	Yes	Yes	Yes
FOREIGN_ANTI_VIRUS	(Boolean) Add anti-virus protection for those Windows sites that for policy or other reasons must run anti-virus systems other than F-Secure on their servers.	No	No	No	Yes

Variable	Short Description	VM	VMS	Unix	Win
GETQWAIVE	Internet addresses of persons to be granted an "infinite" GET quota.	Yes	No	No	No
HIDE_TRACEBACK	(Boolean) Determines whether or not error trace backs are shown to non-postmasters.	Yes	Yes	Yes	Yes
IGNORE	A list of userid@nodes whose messages and files are to be ignored.	Yes	No	No	No
IGNORE_EMAIL_CASE	Provided for DBMS lists with UEMAIL fields to work around RFC821 requirement that an email address local-part must be evaluated case-sensitively.	No	Yes	Yes	Yes
IGNORE_EXTERNAL_PRIME	Boolean value determining whether or not LISTSERV will ignore the PRIME setting on incoming DISTRIBUTE jobs.	Yes	Yes	Yes	Yes
INDEX_VIA_GETPOST	On VM, determines whether INDEX subscriptions use GET-POST or old-style database jobs by default.	Yes	No	No	No
INSTPW	The optional local "installation password" associated with the INSTALL command.	Yes	No	No	No
JOB_STAT_DEFAULT	Boolean value determining whether or not the "Summary of resource utilization" is generated or suppressed in a CJLI JOB command response.	Yes	Yes	Yes	Yes
LICENSE_WARNING	Toggles license warnings on and off. <b>Warning:</b> See the <a href="#">Site Configuration Keyword Reference</a> document before changing this value.	Yes	Yes	Yes	Yes
LIST_ADDRESS	Default value for the List-Address= keyword.	Yes	Yes	Yes	Yes
LIST_EXITS	Filenames of executable files that can be defined as exits by an Exit= list header keyword.	Yes	Yes	Yes	Yes
LMCPUT	A boolean variable indicating how PUT commands for datafiles associated with the LMC FAC are handled	Yes	No	No	No

Variable	Short Description	VM	VMS	Unix	Win
LOCAL	A list of nodes to be associated with the hardcoded LCL FAC. Also used as the default for the Local= list keyword.	Yes	Yes	Yes	Yes
MAILER	Internet address of the local mailer.	Yes	No	No	No
MAILMAXL	The maximum number of lines for an incoming MAIL file to be accepted.	Yes	Yes	Yes	Yes
MAXBSMTP	Maximum number of 'RCPT TO:' lines per BSMTP file sent to the mailer.	Yes	Yes	Yes	Yes
MAX_CONSECUTIVE_SUBS	The maximum number of lists to which consecutive subscription attempts will be accepted from a given subscriber, to prevent "spoofing" attacks.	Yes	Yes	Yes	Yes
MAXDISTL	(HPO only) The maximum number of recipients to be listed in the LISTSERV console log as recipients of an SMTP job.	Yes	Yes	Yes	Yes (NT)
MAXDISTN	Maximum number of recipients in forwarded DISTRIBUTE jobs.	Yes	Yes	Yes	Yes
MAXGET	Maximum number of GET requests a user can make per day.	Yes	No	No	No
MAXGETK	Maximum number of kilobytes of data a user may obtain a day via GET requests.	Yes	No	No	No
MDISK.cuu	Information about library mini-disks.	Yes	No	No	No
MSGD	Userid of the virtual machine running a RFC1312/MSP server, if "Internet TELL" support is desired.	Yes	No	No	No
MYDOMAIN	The list of domain names that are equivalent to NODE – e.g., MX addresses, CNAMEs, etc.	Yes	Yes	Yes	Yes
MYORG	Short organization name that appears in the RFC-822 "Sender:" line.	Yes	Yes	Yes	Yes
NDMAIL	Whether to send mail to the local MAILER in Netdata format.	Yes	No	No	No



Variable	Short Description	VM	VMS	Unix	Win
NODE	Internet address of the LISTSERV host.	Yes	Yes	Yes	Yes
NO_NJE_JOBS	Directs a VMS™ server running in NJE mode to send all outgoing server-to-server requests via the Internet.	No	Yes	No	No
OCI_*	Three configuration variables under the OCI_* rubric are available for use with LISTSERV's DBMS/Mail Merge functionality. Please see the <a href="#">Advanced Topics Guide for LISTSERV</a> for more information.	No	Yes	Yes	No <sup>b</sup>
ODBC_*	Three configuration variables under the ODBC_* rubric are available for use with LISTSERV's DBMS/Mail Merge functionality. Please see the <a href="#">Advanced Topics Guide for LISTSERV</a> for more information.	No	No	No	Yes
OFFLINETHR	Defines the system and RSCS spool thresholds for automatic offline/online control.	Yes	Yes	No	No
PLAIN_FROMLINE	Controls the "verboseness" of LISTSERV's administrative From: line.	Yes	Yes	Yes	Yes
POSTMASTER	A list of userid@nodes, of which the first one is the "main" postmaster (to receive transferred files).	Yes	Yes	Yes	Yes
PRIMETIME	Defines the "prime time" for your node.	Yes	Yes	Yes	Yes
QUALIFY_DOMAIN	Defines the domain to be appended to all non-qualified addresses.	Yes	Yes	Yes	Yes
RESMODES	Defines a list of filemodes which are to be considered as "reserved" and never available for dynamic ACCESS.	Yes	No	No	No
RSCS	A list of local userids which must be treated as RSCS virtual machines.	Yes	Yes	No	No
RUNMODE	The mode (NETWORKED, STANDALONE, or TABLELESS) that LISTSERV runs in.	Yes	Yes	Yes	Yes

Variable	Short Description	VM	VMS	Unix	Win
SEARCH_DISABLED	Determines whether or not the (new) SEARCH command is enabled.	Yes	Yes	Yes	Yes
SMTP_FORWARD	The Internet hostname of the server to which all outgoing SMTP mail should be forwarded for delivery.	No	Yes	Yes	Yes
SMTP_FORWARD_n	Defines "n" number of "SMTP workers" used to split up the SMTP forwarding load.	No	Yes	Yes	Yes
SMTP_LISTENER_IP	Dotted-decimal IP address that sets the IP address to which the SMTPL.EXE "listener" will bind at boot time.	No	No	No	Yes
SMTP_LISTENER_PORT	Integer value that sets the port number to which the SMTPL.EXE "listener" will bind at boot time.	No	No	No	Yes
SMTP_RATE_LIMIT	Defines bandwidth limits for the server. Typically used with delivery pools, but can be used to set a server-wide bandwidth limit. (Classic, Classic HPO)	No	Yes	Yes	Yes
SMTP_RESET_EVERY	Directs LISTSERV to reset open SMTP connections every "n" minutes.	No	Yes	Yes	Yes
SORT_RECIPIENTS	Determines whether or not to sort recipients in the RFC821 mail envelope.	Yes	Yes	Yes	Yes
SPAM_ALERT	Determines whether or not spam reports are sent to the postmaster.	Yes	Yes	Yes	Yes
SPAM_DELAY	Sets the server-wide value (in minutes) for the anti-spam quarantine period.	Yes	Yes	Yes	Yes
SPAM_EXIT	Sets the name of the "user provided" exit program used to call a third-party spam scanner. (VM/VMS require AVS)	Yes	Yes	Yes	Yes
SPAM_MAXSIZE	Sets the maximum size, in kilobytes, of any message to be handled by the spam scanner. Messages over the specified size are not scanned. (VM/VMS require AVS)	Yes	Yes	Yes	Yes

Variable	Short Description	VM	VMS	Unix	Win
SSI	Flag telling LISTSERV that it runs in a SSI system.	Yes	No	No	No
STARTMSG	Recipients of start and stop messages.	Yes	Yes	Yes	Yes
STOREPW	The password to be used by postmasters when executing CP/CMS commands and when storing files in the server by means of the PUTC command. <sup>c</sup>	Yes	Yes	Yes	Yes
SYSTEM_CHANGELOG	Enables a system-level changelog.	Yes	Yes	Yes	Yes
TCPGUI_IPADDR	Defines the IP address used by the TCPGUI interface.	No	Yes	Yes	Yes
TCPGUI_PORT	Defines the port number used by the TCPGUI interface.	No	Yes	Yes	Yes
TRAPIN	List of userid@node templates from whom LISTSERV should never accept mail.	Yes	Yes	Yes	Yes
TRAPOUT	List of userid@node templates to whom LISTSERV should never send mail.	Yes	Yes	Yes	Yes
TUNE_MANY_LISTS	(HPO only) Enables a suite of HPO functions that can speed up operations on servers with many lists (>100).	Yes	Yes	Yes	Yes
UODBC_*	Three configuration variables under the UODBC_* rubric are available for use with DBMS/Mail Merge.	No	No	Yes <sup>d</sup>	No
USE_LSMTMP_MAIL_MERGE	OBSOLETE as of 14.5, see EMBEDDED_MAIL_MERGE instead.	No	Yes	Yes	Yes
VM30091	Indicates whether or not the VM30091 message suppression functions are available.	Yes	No	No	No
WEB_BROWSER_CONFIRM	Indicates whether or not LISTSERV should require "OK" confirmation for commands sent from WWW browsers.	Yes	Yes	Yes	Yes
WWW_ARCHIVE_CGI	The (preferably) relative URL that leads to the WWW archive CGI script. (This is a URL, not an OS path name.)	No	Yes	Yes	Yes

Variable	Short Description	VM	VMS	Unix	Win
WWW_ARCHIVE_DIR	The full OS path name to the WWW archive directory.	No	Yes	Yes	Yes
WWW_AUTHINFO_DISABLE	Disable or enable the web archive interface's IP address verification function.	No	Yes	Yes	Yes
XFERTO	Userid of the virtual machine to which files found in the lists readers should be transferred.	Yes	No	No	No

- a. Depending on platform support for DBMS. See the full documentation for details.
- b. The native OCI interface was available for Windows servers only in version 1.8d. It was removed in LISTSERV 1.8e. See the [Site Configuration Keyword Reference](#) document for details.
- c. For non-VM systems, STOREPW is a secondary password that is functionally identical to CREATEPW. You should use the same value for both passwords, i.e., set STOREPW=%CREATEPW% (for Windows NT/2000 and 95/98), etc.
- d. Dependent on unix platform support for unixODBC.

There are also a number of configuration variables pertaining to the DBMS/Mail Merge functions. These variables are documented in the [Advanced Topics Guide for LISTSERV](#).

### 5.3 Files Used by LISTSERV

The proper operation of LISTSERV is dependent on LISTSERV's ability to find a number of files that belong to it. The following list of files are required to operate the product, and in most cases, it must be located in the same directory with the LISTSERV executables.



**Notes:** The exception is under unix, where all of the data files other than the 'go\*' files MUST be placed one directory below the executables, typically in ~listserv/home.

Under certain conditions, some required files aren't necessary; these will be noted where applicable. In addition, some files are not shipped with the distribution, but are generated automatically the first time you run LISTSERV.

#### 5.3.1 Program Executables

Depending on the platform, the required executables are:

- For VM: LSV EXEC
- For OpenVMS: LSV.EXE
- For Unix: lsv\*
- Windows (all): LSV.EXE

For OpenVMS and Windows systems, the executable SMTPW.EXE is also required.

For Windows systems, the executable SMTPL.EXE is also required.

The executables listed above belong in the following places (depending on the platform):

- For VM, the executable belongs on LISTSERV's A disk.
- For OpenVMS, the executables belong in LISTSERV's "A" directory, normally called `LISTSERV_ROOT:[MAIN]`.
- For unix, the executable belongs in the LSVROOT directory (`~listserv/`).
- For Windows, the executables belong in LISTSERV's "A" directory, normally `drive:\LISTSERV\MAIN`.

Finally, the Web Archive ('wa') interface CGI script is shipped in the "A" directory of the non-VM servers. This script must be copied into the appropriate script directory for your Web server (see Section 5.4 [Installing & Configuring LISTSERV's WWW Archive & Administration Interface](#)) if you plan to use the Web Archive interface. For OpenVMS and Windows servers, this file is `WA.EXE`, while on unix machines it is `wa*`.

### 5.3.2 BITNET Network Table Files

These files are not required when running LISTSERV with `RUNMODE=TABLELESS`, and are not shipped with LISTSERV Lite. Network table files include:

- `bitearn.nodes`
- `bitearn.linksum2`
- `bitearn.dbindex`
- `bitearn.nodesum3`
- `bitearn.distsum2`
- `bitearn.linkdef2`

With the exception of BITEARN NODES, all files are regenerated whenever BITEARN NODES is updated or when an explicit NODESGEN command is issued. For pre-1.8c servers (or non-registered 1.8c or later servers), BITEARN NODES must be downloaded on an approximately monthly basis from

`ftp://ftp.lsoft.com/listserv-data/bitearn.nodes`

or from the European mirror at

`ftp://seagate.sunet.se/listserv-data/bitearn.nodes`

BITEARN NODES on registered networked servers are updated using the same mechanism as PEERS NAMES and other LISTSERV tables. Note that this requires that your mail server support incoming files of at least 1.5M. VM sites have not been included as they typically maintain this file using the UPNODES procedure and store it on a public disk, applying change control procedures in the process.

### 5.3.3 Internet and Peer Networking Table Files

These files are used by LISTSERV to define its "backbone" and other peer servers, as well as to help determine the best routes for mail sent via the DISTRIBUTE algorithm.

- `aliases.names`
- `intlincs.file`

- `intpeers.names`
- `linkswt2.file`
- `peers.dbindex`
- `peers.dbnames`
- `peers.distsum2`
- `peers.names`
- `peers.namesum`
- `service.names`

For registered sites they are updated periodically by mail from other servers. The update process is automatic and does not require LISTSERV maintainer intervention unless a problem is noted.

For non-registered sites the files must be updated manually. See <http://www.lsoft.com/products/table-updates.asp> for information on how to accomplish this.

Sites running in TABLELESS or STANDALONE mode do not require these files. This includes all LISTSERV Lite and LISTSERV Shareware sites.

### 5.3.4 LISTSERV's External Data Files

LISTSERV uses these files for a number of purposes. The fact that they are external to the executables makes it easy to update them when needed. These files include:

- `country.file`
- `default.mailtpl`
- `default.wwwtpl`
- `errfac.file`
- `listkwd.file`
- `lsvhelp.file`
- `lsvinfo.file`
- `permvars.file`
- `signup.file`
- `sitecfg.file`
- `stdcmd.file`
- `syscfg.file`
- `sysff.file`
- `site.catalog`
- `system.catalog`



**Important:** PERMVARS.FILE is LISTSERV's main "permanent variables" file; among other things, this is where LISTSERV registers spammers and users that have been served off. This file should NEVER be modified manually. It is in a binary format and, if corrupted, LISTSERV will not start.



**Important:** SITECFG.FILE is where LISTSERV stores changes made to the main site configuration file via the web administration interface. This file should NEVER be modified manually.

The SIGNUP.FILEx files (initially there are 9 [or 31 if you are licensed for HPO], for example, SIGNUP.FILE1, SIGNUP.FILE2, etc.) are used to register users and their real name fields.

SYSTEM.CATALOG is used by LISTSERV to register system files; it should not be modified, as it is always shipped with new versions and will thus overwrite itself. Instead, SITE.CATALOG should be used to register files and list file archive catalogs (listname.CATALOG) for users to retrieve. (SITE.CATALOG is not shipped with LISTSERV; please Section 8 [File and Notebook Archives](#) for details.)

DEFAULT.MAILTPL and DEFAULT.WWWTPL are the files from which LISTSERV gets its default mail templates and default web templates for responses to user input. See Section 9 [Creating and Editing Mail and Web Templates](#) for details.

SYSCFG.FILE tells LISTSERV what site-level variables can be configured (and how) from the web administration interface. It should NEVER be altered. It will be overwritten in an upgrade.

### 5.3.5 User Reference Material

The following files are LISTSERV's online documentation.

- listdb.memo
- listfile.memo
- listkeyw.memo
- listmast.memo
- listpres.memo
- listjob.memo
- listlpun.memo
- listownr.memo
- listserv.memo
- listall.refcard

LISTALL.REFCARD is broken into three parts internally. Part 1 is the response to the INFO REFCARD command; Parts 1 and 2 are the response to a GET LISTOWNR REFCARD command; and the whole document is sent in response to a GET LISTMAST REFCARD command.

### 5.3.6 Command Line Utilities (Non-VM)

Depending on your platform, the following executables may have been shipped (under unix they must all be compiled from the corresponding \* .c files):

- LCMD.EXE (or lcmd\*)

LCMD is a command-line named-pipes interface to LISTSERV. You can use it to send commands directly to LISTSERV from the console and receive information in return, either on the console itself (Windows and OpenVMS) or via mail (unix). The syntax is:

Windows: `lcmd [\\computer[serverid]] command`

OpenVMS and unix: `lcmd command`

The user running LCMD must have appropriate permission (e.g., must be a list owner or LISTSERV maintainer) in order to issue the various protected commands.

- `LISTVIEW.EXE` (or `listview*`)

LISTVIEW is a utility that allows you to type the binary-format .LIST files to standard output so that they can be viewed and/or redirected to text files. The syntax is:

```
listview [-a] [-e[h]] [-h] [-r nnnn] [-s] file1 file2...
```

You can choose only one of the command line options at a time, except that you can specify one of the other options along with the `-r` option if needed. The options are:

- `-h` shows the header only.
- `-s` shows the header + the subscribers (without the option string in columns 81-100).
- `-e` shows the list of subscribers only (without the option string).
- `-eh` similar to `-e`, but show only the hostnames without `userid@`.
- `-a` shows the entire list file.
- `-r nnnn` generates two files (`listname.view1` and `listname.view2`) for each list file viewed with this option. The `view1` file contains `nnnn` subscriber addresses chosen at random from the list, where `nnnn` is an integer value between 1 and the number of users on the list. The `view2` file contains the rest of the subscriber addresses from the list, randomly sorted. (The `view2` file is useful in cases where you wish to pull `x` names at random from your mailing list, and then pull `x` more names at random without duplication. Note however that you would have to add the subscribers in the `view2` file to a regular LISTSERV list in order to be able to run `listview` against those subscribers.)

While you can specify one other option with `-r` to manipulate the output, the following caveats should be noted:

- `listview -h -r` results in a blank file
- `listview -s -r` does not output the list header
- `listview -eh -r` outputs a list of random hostnames; they are not unique.
- `listview -a -r` is the same as `listview -r`



Note carefully that running `listview -r` against a mailing list with a value of `nnnn` greater than the actual number of subscribers in the list will result in duplicates being written to the `view1` file and the generation of a `view2` file of length 0.

LISTVIEW executed with no option is the same as `'listview -a'`.

You can redirect the output of LISTVIEW with standard OS-dependent redirection symbols. For instance,

```
listview -h mylist > mylist.file
```

redirects the output to the ASCII file `'mylist.file'`.

- `JOBVIEW.EXE` (or `jobview*`)

JOBVIEW allows you to read the Base64-encoded spool files created by LISTSERV (see below for the types of files created in the spool directory that may be read with this utility). The syntax is simply `jobview file1 file2...`

### 5.3.7 GUI Site Configuration Utility (Windows Only)



**Notes:** The `SITE.EXE` and `SITE.HLP` files are obsolete as of LISTSERV 15.0 and no longer shipped or maintained. Please use the LISTSERV web administration interface to change the site configuration.

The Site Configuration Utility for LISTSERV allows you to easily configure LISTSERV's operation. While this can also be done by manually editing LISTSERV's `SITE.CFG` file, the GUI gives you an easier way to take care of this task. Online help for the various configuration variables is provided, and new LAKs can be entered. Basic optimization for various pre-calculated loads can also be performed.

### 5.3.8 Line-mode site configuration utility (OpenVMS only)

`LISTSERV_CONFIGURE.COM` is a very basic line-mode utility that allows you to modify the OpenVMS version of the site configuration file. It is useful for initial configuration, but most OpenVMS sysadmins will probably prefer to edit the `SITE_CONFIG.DAT` file by hand with a text editor.



**Note:** Newer configuration variables are not covered by `LISTSERV_CONFIGURE.COM`. Please use the LISTSERV web administration interface to change the site configuration.

### 5.3.9 Other files that will appear during use

While in use, LISTSERV creates various files for itself. On the A disk or in the MAIN or HOME directory, these are typically:

- `.AUTODEL` files – Maintain data for LISTSERV's autodeletion functions; one for each list that has Auto-Delete enabled. If no auto-deletion reports are pending, this file will not exist.

- **.CHANGELOG files** – Contain data regarding subscription changes for a given list if that list has the "Change-Log= Yes" list header keyword setting. These files are called *listname* CHANGELOG on VM.
- **.DIGEST files** – These files are the (volatile) digest files for each list that has digests enabled. They are deleted and restarted when the digest is cut. Note that if the location parameter of the Digest= keyword is not set to something that points to the MAIN or HOME directory, then the .DIGEST files will not appear in the MAIN or HOME directory, but rather in the directory specified.
- **.LIST files** – Mailing list files, including the header and subscriber information. Do not attempt to edit these files with a text editor; use the GET and PUT commands instead.
- **.OKxxxxxxx files** – Usually found for edited lists, but can also appear for non-edited lists if users are set to REVIEW. These are mail messages that are awaiting "OK" confirmation. If they are not confirmed, they are automatically deleted after about a week.
- **.OLDLIST files** – These files contain the last saved version of the list file. If you PUT a header and find that you've made a fatal mistake (like adding users "on the fly" and deleting everyone else on the list, or editing the list file by hand and corrupting the record structure) you can send the command GET listname (OLD to have the listname.OLDLIST file sent to you.
- **.SUBJECT files** – Maintain the list of subjects for the digest. Again, if digests are not enabled for a specific list, this file does not exist for that list. Also, the same note for the location of these files as for .DIGEST files applies. .SUBJECT files are deleted and restarted when the digest is cut.

In the SPOOL directory, the following file types will be found:

- **.DELETED** – These are processed JOB files that have been left in the spool when LISTSERV was running under a debugging regime, i.e., to compare JOB file input to LISTSERV output. Normally JOB files are deleted from the spool after processing. These files can be viewed with the jobview utility. Unless you are actively debugging something, files with this extension can be safely deleted.
- **.ERROR** – LISTSERV generates an .ERROR file in the spool when it encounters an error in a JOB file. These can be viewed with the jobview utility and are important for tracing certain errors back.
- **.JOB** – Files that have been received by LISTSERV and are queued for processing. These files are in Base64 format and can be viewed with the jobview utility.
- **.JOBH** – Held .JOB files. Such files are either being processed by LISTSERV (and are thus locked) or have generated an error message. These can also be viewed with the jobview utility.
- **.MAIL** – Files that have been processed through LISTSERV and are queued for delivery to the outgoing SMTP mail agent. These are plain-text files.

- `.MAIL-ERR` – Files that have been processed through LISTSERV and for which delivery has been attempted, but for which a "permanent" SMTP error has resulted. If you have reason to believe that the error was not actually "permanent", simply rename the file with the `.MAIL` extension and LISTSERV will pick it up for another try.

`JOBH` files containing the string `$NOJOB$` in the filename are typically waiting to be processed because the list they are going to has an explicit `Prime=` variable set and the non-prime time has not yet arrived.

## 5.4 Installing & Configuring LISTSERV's WWW Archive & Administration Interface

LISTSERV 15 includes an optional WWW archive and administration interface (not enabled by default). This interface is used to allow users to browse and search notebook archives for lists with the feature explicitly enabled, as well as to allow list owners to manage almost every aspect of their lists and to allow LISTSERV maintainers to perform a number of common site management tasks. The interface is secured by the use of LISTSERV personal passwords. List owners have administrative access only to their own lists; general users have access only to the archives of public lists or to private lists to which they are subscribed (in other words, there is no difference between the access one receives via the web interface and the access one receives via the mail interface).

### 5.4.1 The WWW Archive Interface

Postings can be organized by date, by topic or by author, and a search function with online help is provided. LISTSERV's WWW interface has the following advantages over "hypermail" style web archiving:

- The information on the web is always up to date. New postings are shown as soon as they are received.
- The postings can be organized in the manner that best suits the reader: by date, by author, by topic, with or without table of contents, with or without showing the author, etc.
- Only one copy of the information is kept, and in particular there is no need to create an individual HTML file for each posting. This design allows the interface to scale up gracefully to lists with hundreds of thousands of archived postings, which would otherwise require hundreds of thousands of individual HTML files, wasting disk space (each file takes up at least one disk block) and stressing the file system past reasonable limits.
- The search forms can be used to create search requests matching (for instance) all postings in the last X days. The resulting URL can then be bookmarked and reloaded on a regular basis.
- List owners can customize the main page for their lists without any intervention by the LISTSERV maintainer, by updating one of the mail template forms for their list (`WWW_INDEX`). The LISTSERV maintainer can customize common pages and header/trailer HTML statements by updating system templates.

To take advantage of the interface, you must first ensure that the "Notebook=" options for your list are compatible with the WWW interface. In most cases, you will not have to

do anything, but certain options are incompatible with the use of the WWW interface and may need to be changed:

- The archive frequency MUST be WEEKLY, MONTHLY or YEARLY. SEPARATE and SINGLE notebooks are not supported. L-Soft generally recommends converting lists with SINGLE notebooks to YEARLY unless there is a compelling reason to have all the messages in exactly one file.
- For optimal performance, the archive frequency may need to be adjusted to produce an "adequate" number of topics and messages in each archival period. The definition of "adequate" depends on your users, the kind of equipment they have, and how they connect to the Internet. As a rule, home users will prefer a larger number of smaller archives whereas office users with large screens and T1 or better connectivity will tolerate a larger table of contents.
- L-Soft strongly recommends that the directory in which your list archives are kept should be specified in the Notebook= list header keyword in absolute rather than symbolic form. Symbolic form is when the directory name is a single letter, for instance "Notebook= Yes,A,Monthly,Public" ("A" being a logical filemode defined in LISTSERV's site configuration which points to the directory where LISTSERV keeps its internal files). In most cases, your list header will probably read something like "Notebook=Yes,E:\LISTS\XYZ-L,Monthly,Public" and you will not have to worry about this.

If a logical filemode is used, LISTSERV will translate it into a full path, but L-Soft still strongly discourages the use of logical filemodes for the "where" parameter of the Notebook= keyword, primarily for security reasons but also to keep things orderly. A full path is always preferred, and each list should imperatively have its own subdirectory. See Section 5.8 [Setting Up Archive and Notebook Directories for Use with LISTSERV](#) and Section 8 [File and Notebook Archives](#) for details.

The LISTSERV maintainer must then enable the list for the WWW interface. This may require the installation of a web server and of the WWW interface code itself. You can then modify the WWW\_INDEX mail template form to customize the main archive page for your list. See Section 9 [Creating and Editing Mail and Web Templates](#) for more information on customizing mail templates.

#### 5.4.2 The WWW Administration Interface

Assuming that the WWW interface has been installed per the instructions below, the WWW administration interface is enabled automatically for all lists on the server that are not coded "Validate= Yes,Confirm,NoPW" or "Validate= All,Confirm,NoPW". The default entry URL is

```
http://hostname/script-directory/wa[.exe]
```

By default, this directs all users to the LISTSERV archives page. If the default has been changed by setting personal preferences, the archives page can be reached at

```
http://hostname/script-directory/wa[.exe]?INDEX
```

The basic URL for the list owner "dashboard" is

```
http://hostname/script-directory/wa[.exe]?OWNER
```

where hostname is the name of the LISTSERV host, and script-directory is the name of the directory where "wa" is installed. For unix you specify "wa?OWNER" and for Windows and VMS you specify "wa.exe?OWNER". With some non-unix web servers you may have to type `WA.EXE?OWNER` (that is, all in upper case) in order for this to work.

Site managers can reach the server administration dashboard at

```
http://hostname/script-directory/wa[.exe]?ADMINDASH
```

From here they may create lists, customize site-wide WWW templates, manage DBMS and mail-merge operations, and so forth.

See Section 11 [Using the Web Administration Interface](#) for more details on the Web Administration Interface.

### 5.4.3 Installing a Web Server



**Notes:** L-Soft cannot help you with the installation or configuration of your web server itself. L-Soft does not recommend or endorse specific web servers, nor does L-Soft have development machines with every possible web server installed. You should ensure that the web server software you choose is installed and operating properly before attempting to install the LISTSERV WWW interface script.

If you do not already have a World Wide Web server installed and operating on your LISTSERV machine, you will need to obtain and install one. There are quite a few free web servers available for downloading on the Internet for most systems; you may want to start your search for server software at the W3 Consortium's web site at <http://www.w3.org/>. Naturally, commercial web servers can also be used.

For security purposes you should always disable directory browsing if it is not disabled by default by your web server.

### 5.4.4 Installing the Web Archive Interface Script

The CGI script for the web archive interface must be installed in the directory where your web server normally keeps CGI scripts and from which they are authorized to run. If in doubt, please read the manuals that came with your web server and/or contact the web server manufacturer's support group; L-Soft cannot help you with this. LISTSERV cannot install the script for you because installation depends on which server you use, which operating system you are running, how the server has been configured, etc.



**Note:** The web interface is not designed to be run on a machine separate from the LISTSERV server. It **MUST** run on the same machine. This means that a web server **MUST** be installed on the LISTSERV machine or you will not be able to use the web interface.

System specific instructions:

- **Windows:** It is strongly recommended that the web interface be installed when LISTSERV is installed (the installation program prompts you to do so).

However, if it is necessary to install the web interface manually: Copy `WA.EXE` to the appropriate directory. For Microsoft's Internet Information Server (IIS), this is normally `C:\INETPUB\SCRIPTS` (not `C:\INETPUB\WWWROOT\SCRIPTS`). For Apache for Windows, this is normally `C:\apache2\cgi-bin`.

`WA.EXE` builds shipped with LISTSERV 15.0 communicate with LISTSERV via TCP/IP. If your `%SystemRoot%` directory (e.g., `C:\WINNT`) is on an NTFS partition, in order for this to work properly it may be necessary to grant the "Everyone" user (or at least the user that invokes `WA.EXE`, for example, `IUSR_XXX` under IIS) R/X permissions on the following files in the `%SystemRoot%\system32` directory:

- `MSAFD.DLL`
- `WS2_32.DLL`
- `WS2HELP.DLL`
- `WSHTCPIP.DLL`
- `WSOCK32.DLL`

Under IIS the invoker is normally the `IUSR_XXX` user created when you install IIS. Other web servers are probably different and you may have to check the logs to see what user is invoking `WA.EXE`.

This instruction can be ignored if your `%SystemRoot%` directory is on a FAT or FAT32 partition. However, using FAT or FAT32 partitions is insecure and is not recommended.

- **unix:** The LISTSERV installation kit will offer to install the web interface, but if it is necessary to do a manual install, you can copy 'wa' to the appropriate cgi-bin directory, change its owner to 'listserv' and set the suid bit (typically, 'chmod 4755 wa'). This authorizes the interface to read archive files. Please note that one of the most common problems with 'wa' under unix is that the installer has not followed this instruction.
- **OpenVMS:** A precompiled `WA.EXE` is provided. However, for some web servers, you may need to link `WA.OLB` with the CGI library provided with the web server to make a new `WA.EXE`. You then copy it to the appropriate cgi-bin directory. Make sure to arrange for the program to have read access to the archive files for the lists you want to serve on the web. Again, this may vary from one web server to another.

While the script can be renamed, a short name will help keep the HTML documents small and speed up the site.

### 5.4.5 Creating a Subdirectory for the Archive Interface

Create a subdirectory on your web server to contain the various files LISTSERV will be creating for the web archive interface. The suggested name (and the name LISTSERV will expect by default) for the subdirectory you will create in this step is 'archives'. Under IIS, you would typically make the directory `C:\INETPUB\WWWROOT\ARCHIVES` for this purpose. For a unix server running Apache it might be `/var/www/html/archives`.





**Important:** Please note the following restrictions carefully:

- Do not simply use your main HTML documents directory as LISTSERV will create quite a few files. It is much more orderly to keep the web archive interface's files and subdirectories in their own place in any case.
- Do not use the directory you keep the list's notebook archives in for this purpose. Notebook archives should always be kept separate from the web interface, preferably in a completely separate directory hierarchy.
- Do not set the `Notebook=` keyword for any list so that the list's notebook archives are kept in the subdirectory used by the web archive interface for the list.

For specifics on what should be kept in what directories, see Section 5.8 [Setting Up Archive and Notebook Directories for Use with LISTSERV](#).

System specific steps:

- **OpenVMS:** define the system-wide logical `LISTSERV_WWW_ARCHIVE_PATH` to point to the directory you just created, and `LISTSERV_WWW_ARCHIVE_URL` with the URL to the directory in question (preferably relative).
- **unix:** create a world-readable file called `/etc/lsv-wa.config` with the following two statements:

```
PATH xxx
```

```
URL yyy
```

where 'xxx' is the absolute path to the directory you've just created and 'yyy' is the URL to this directory (preferably relative). For instance:

```
PATH /usr/local/etc/httpd/htdocs/archives
```

```
URL /archives
```

- **Windows:** If necessary (and it shouldn't be), you can update the registry key `HKEY_LOCAL_MACHINE\SOFTWARE\L-Soft\LISTSERV\WWW_ARCHIVE_URL` to override the default URL to the directory you have just created. Again, this is not normally necessary and is only provided for weird web servers, etc. Don't do it unless it didn't work without it.

#### 5.4.6 Configuring LISTSERV to Activate the Web Archive Interface

This is done by modifying LISTSERV's site configuration file (see the [Site Configuration Keyword Reference](#) document) to add two variables:

- `WWW_ARCHIVE_CGI` is the (preferably) relative URL that leads to the CGI script you have just installed. Typically this will be something like `'cgi-bin/wa'` or `'scripts/wa.exe'`. This is a URL, not an OS path name.
- `WWW_ARCHIVE_DIR` is the full OS path name to the directory you created in the previous step (`C:\INETPUB\WWWROOT\ARCHIVES` or whatever).

Under unix, you may have to export these variables (you can check the 'go' script to see if they are already exported for you; in early versions of the interface they were not) by adding these lines:

```
export WWW_ARCHIVE_CGI
export WWW_ARCHIVE_DIR
```

at the end of go.user. Again, if these variables are already exported in the 'go' script, there is no need to do this.

LISTSERV will then create and maintain a file called:

```
http://localhost/archives/index.html
```

from which you can access all the postings. (This is made from the template WWW\_ARCHIVE\_INDEX – see below.)



**Note:** Proper operation of the interface requires that the 'wa' script be able to talk to LISTSERV via TCP/IP to the local port 2306 (LISTSERV's TPCGUI port). This may require a local firewall exception, but there is no technical need for the port to be opened to the world at large.

### 5.4.7 Customizing the Web Pages

The simplest (and recommended) way to make changes to the templates that contain the information for these pages is to use the tools provided in the WWW administration interface for changing the "look" of your site. Because the template system has become quite complicated, we strongly recommend using the web interface tools for this purpose rather than trying to edit the template files by hand.

Most of the templates for the web interface are found in the file default.wwwtpl. Templates that you edit are placed in a file called site.wwwtpl.

You should never make changes to default.wwwtpl itself as it is always overwritten when LISTSERV is upgraded.

### 5.4.8 Enabling Individual Lists

Once the interface is installed, LISTSERV will automatically make any mailing list with public archives available through it, provided that a subdirectory has been created for them in the 'archives' subdirectory created above, and provided that LISTSERV has read/write access to the subdirectory.



**Warning:** CONFIDENTIAL=YES does not limit access to the Archive. Public notebooks for any list coded "Confidential= Yes" will be available via the interface if a subdirectory under 'archives' is created for that list. However, the list will not appear on the main archive index page if you have coded "Confidential= Yes". In this case there are two avenues for users who know the list exists and want to access the web archives:

1.) Users can bookmark or manually enter the link to the list's archives, for example, <http://www.yourhost.com/archives/yourlist.html>

2.) Users can click on the link at the bottom of the main index page that pulls up an



"unlisted archive" form, into which they can type the name of the list.

On the other hand, if you code "Confidential= Service", your list will show up on the main archive index page for your server but will not show up in the CataList or global list of lists.

"Private" notebooks can be viewed via the WWW interface by following the same instructions as for "Public" notebooks. However, in order to view the notebooks, subscribers must log in with their subscribed email address and their LISTSERV password (set with the PW ADD command or via the WWW interface).

Please note carefully that if the user is subscribed as "joe@unix1.host.com" and tries to log in as "joe@host.com", he will be refused access. Also note that unless the list is coded "Confidential= Yes", there will be a link to its archives in the main archive index page.

If you do not want the confidential and/or "Private" list's notebooks available via the WWW interface at all, simply do not make a subdirectory for it under 'archives'.

Please note that when deleting a list without deleting the list's archive notebooks, you MUST delete the list's index directory under the web interface. Otherwise someone with a bookmarked URL may still be able to access some of the archives via the web.

Also note that under unix, if the 'wa' script does not have the suid bit set, the interface will appear to work normally until you try to read a message. If the suid bit is not set, you will receive a message saying the archives are not available and to try again in 30 seconds.

As an example, let's assume that you have a list called XYZ-L that you want to make available through the Web interface, and that so far you have used the defaults for the installation of the interface.

First, under the 'archives' directory you created above, you must create a directory with the same name as your list. Thus, in order to make the XYZ-L list accessible through the interface, you must create the directory 'archives/xyz-l'

Next, you would edit the XYZ-L header to indicate how you want the list to appear to the interface. If you want the archives to be wide open, you must code

```
* Confidential= No
* Notebook= Yes,where,interval,Public
```

If you want the archives to be "wide open" but don't want a link on the main archives page, you would code

```
* Confidential= Yes
* Notebook= Yes,where,interval,Public
```

If you want the archives to be accessible only by subscribers (with a password) and to have a link on the main archives page, you would code

\* Confidential= No

\* Notebook= Yes,where,interval,Private

And, if you want the archives to be accessible only by subscribers (with a password) but you do not want a link on the main archives page, you would code

\* Confidential= Yes

\* Notebook= Yes,where,interval,Private

Finally, if you want the archives to be available via the interface (either with or without a password), and you want a link on the main archives page, but you do not want your list to appear in the CataList or global list of lists, you would need to code

\* Confidential= Service

and Notebook= would be either Public or Private depending on your preference.



**Note:** Coding the Confidential= keyword has other implications. For instance, if you want your list to show up in the CataList or be available via the Global List Exchange (GLX), you must set Confidential= No. Thus advertising your list globally is not compatible with having your archives available via the web but not having a link on the server's main archives index page.

Finally, you would simply perform a GET and PUT of the XYZ-L header (or store it from the web interface without making any changes). When you PUT the header, LISTSERV will create the XYZ-L.HTML file in the 'archives' directory and build indexes for the list in the 'archives/xyz-l' directory.



**Note:** If you do not execute a list PUT operation after creating the directory for the list under 'archives' (for instance, if the list already had public archives and it was not necessary to edit the header), LISTSERV will wait until midnight to create the web archive files for the list rather than creating them immediately. (Naturally, stopping and restarting LISTSERV will also force a rebuild of all of the web interface files but is not recommended as the normal way to accomplish this.)

At this point you will be able to access XYZ-L's archives from the URL  
<http://www.yourhost.com/archives/xyz-l.html>.

### 5.4.9 Enabling Web-Based Bulk Operations

Bulk operations (part of the list owner administration section of the interface) are not enabled by default when the interface is installed. As the site manager, you must create a directory called "upload" under the directory specified in the WWW\_ARCHIVE\_DIR= site configuration variable, and give the userid under which the "wa" CGI program is run write permission in that directory. This is the only directory in which "wa" needs write authority, and only for this functionality. If you do not want the functionality, do not create the "upload" directory.



**Note:** Your browser MUST support the RFC1867 file upload extension or you will not be able to use the bulk operations page. Most current browsers support this extension.

If you get "error 2" when clicking on the **[Import]** button; then the "upload" directory has not been created.

If you get an "error 13" when you click on the **[Import]** button, then the "upload" directory has been created but the CGI program user does not have write permission.

## 5.5 The Spam Detector and Anti-Subscription-Spoofing Feature

L-Soft acknowledges that these features have been continually upgraded and enhanced throughout the development process, but in keeping with previously-announced policy, specifics are proprietary and will not be documented.

### 5.5.1 Spam Quarantine

One of the most arduous problems the spam detector has to face is the accurate detection of the first few copies of the spam. When the first copy reaches the first LISTSERV server worldwide, it is just a posting like any other. It will take repeated occurrences of this same posting for LISTSERV to realize that it is in fact a spam. However, it is desirable to block this very first copy as well, and this can only be accomplished by introducing a delay in the processing of "suspicious" messages. This "quarantine" gives the spam detector some time to gather the necessary evidence to determine if the message is a spam or not. The default value is 10 minutes, and can be changed by adding:

```
* Loopcheck= Spam-Delay(xxx)
```

in the list header (the value is in minutes). The LISTSERV maintainer can also change the system default by adding a `SPAM_DELAY` variable to the LISTSERV configuration with the desired value (also in minutes). A value of zero disables this part of the spam filter.

(It should be carefully noted that setting `SPAM_DELAY=0` does not turn off LISTSERV's spam filter. It turns off only the spam quarantine part of the overall filter. There is no setting to disable the spam filter server-wide; it can be turned off only at the list level, with "Loopcheck= NoSPAM" in the list header.)

The default value of 10 minutes is adequate in most cases; it can be lowered on fast, large, active servers and may need to be increased on servers with backlogs. Currently, LISTSERV determines whether a message is suspicious or not based on the sender's posting history. This however may be changed in future versions to further improve the efficiency of the spam detector.

### 5.5.2 "Anonymous" Spam Alerts

On occasion, you may receive a spam alert from LISTSERV where the offender's e-mail address is replaced with the word "anonymous". These alerts are generated by new detection algorithms where, for various reasons, it may sometimes be desirable to hide the identity of the potential offender, usually because there is a fair chance that the posting is in fact legitimate for the particular lists to which it was posted (for instance because these lists were configured to tolerate a high degree of cross-posting). In this case, information about the text of the message may be released and ultimately lead to a spamming alert that will block further copies of this same message, while the identity of the poster remains hidden.

### 5.5.3 Subscription Anti-Spoofing Feature

About a decade ago, a number of point and click utilities began to appear on anonymous FTP servers, allowing mischievous users to forge Internet mail on an industrial scale and subscribe an unfortunate victim to thousands of mailing lists. The resulting mail onslaught fills the victim's mailbox in minutes, rendering the account forever unusable. It also brings the mail server on which the account is hosted to its knees, causing, in some cases, tens of thousands of dollars in consequential damages as other users of the same system also lose precious e-mail.

In most cases, the account ends up being closed. Unfortunately, this usually doubles the load on the recipient's mail server, as a delivery error needs to be generated for every message received from the mailing list servers. Thus, it is not uncommon for the service provider to leave the account open and simply reconfigure it in such a way that incoming mail continues to be accepted, but is summarily discarded without generating a costly delivery error notification. While it is difficult to blame the service provider for wanting to minimize impact to their customers, the drawback is that the list owners may never be notified of the fact that the account was closed. On any large LISTSERV system, there are likely to be dozens of these addresses, each being sent hundreds or possibly thousands of messages a day which are simply discarded and waste resources.

Until now, the only defense against this attack was to configure mailing lists to require subscription confirmation:

```
* Subscription= Open,Confirm
```

LISTSERV will then send a confirmation request to the victim, who does not reply and thus is not added to the list. While this line of defense is 100% effective, it may not always be practical or desirable to configure the list to require confirmation.

As a result, L-Soft modified LISTSERV to be able to detect these "spoofed" subscription attacks automatically. When more than the default of 50 subscription requests are received from the same account in a short time frame, LISTSERV automatically undoes all the subscription requests and rejects any further subscription attempt for a certain period of time. This applies even to requests that LISTSERV forwarded to other servers; LISTSERV will then send a SIGNOFF request to the remote server for the address in question. Note that, in some cases, the subscription may not be undone, either because of a temporary condition (locked list, etc.) preventing LISTSERV from deleting the user, or because the list was configured with "Subscription= By\_Owner" and the owner manually added the victim after the arrival of the undo request.

This mechanism offers a very good degree of protection against the adverse effects that dead "spoofed accounts" can have on the performance of the LISTSERV host system. It does not, unfortunately, mean that people no longer have to fear subscription spoofing, as only LISTSERV lists are monitored and protected by the "spoofer detector". Requests to subscribe to lists hosted by other mailing list managers are sent directly to the list managers in question, and LISTSERV can only act on the requests that it does receive.

If desired, the maximum number of consecutive subscription attempts can be adjusted by setting the LISTSERV site configuration variable MAX\_CONSECUTIVE\_SUBS .

## 5.6 Server Registration

### 5.6.1 Registering LISTSERV Classic and Classic HPO Servers



**Notes:** This section and Section 5.6.2 [How do I find out if my server is already registered?](#) do not apply to evaluation kits or to LISTSERV Lite kits. Evaluation copies of LISTSERV should not be registered because they are (presumably) temporary servers running test lists, whose existence should not be broadcast.

LISTSERV Lite Free Edition runs only in TABLELESS mode and therefore cannot be registered in the same manner as LISTSERV Classic. LISTSERV Lite paid sites may run only in TABLELESS or STANDALONE runmode, and as such cannot be registered either. In addition, LISTSERV Lite sites, whether Free Edition or not, may not participate in the LISTSERV backbone. For information about how LISTSERV Lite servers are registered, please see Section 5.6.4 [Automatic Registration for LISTSERV Lite Servers](#).

Also note that only those LISTSERV Classic and Classic HPO servers running in NETWORKED mode may be registered.

Once the server is ready for production use (that is, once you have installed a permanent License Activation Key, and once you have arranged for LISTSERV to be started automatically when the system boots), you should register it with L-Soft by filling in a server registration form, and returning it to [SUPPORT@LSOFT.COM](mailto:SUPPORT@LSOFT.COM). Registering the server is necessary to broadcast its existence to the other LISTSERV servers. Once you have registered, your server will be sent periodic updates about the lists hosted by other LISTSERV sites and updates to the files whose versions are shown in the output of the RELEASE command, among other things, and, similarly, other LISTSERV sites will receive information about the public lists you are hosting.

The LISTSERV site registration request form is found at the URL

<http://www.lsoft.com/regform.html>

along with the requirements that must be met for registration.

### 5.6.2 How do I find out if my server is already registered?

If your server is already registered, it will have a :node. entry in LISTSERV's INTPEERS.NAMES file. If your node name is LISTSERV.EXAMPLE.COM, you can look for the line

```
:node.LISTSERV.EXAMPLE.COM
```

in that file.

If you are still unsure, simply contact L-Soft support and ask.

### 5.6.3 The LISTSERV Backbone

The last question on the registration form is whether or not you wish for your site to participate in the LISTSERV backbone.

The LISTSERV backbone is a collection of servers which are operating 24 hours and maintained on a regular basis by their respective operation staffs. This backbone is used to support the DISTRIBUTE command and to host heavy-traffic network-wide peered lists.

LISTSERV servers can fall into one of two categories:

- **Local Server** – A local server has by definition no obligation towards the rest of the network. It can run any release of the code, with or without local modifications. Its operation staff can update it at irregular intervals, place it offline 14 hours a day, or do just anything they might see fit to do. The server will appear in the network routing files but it will be flagged as being a local server.

The only two restrictions placed on local servers are:

- If the server's operation staff encounter a problem with the software and the latest available release of the code has not yet been installed on the server, in general L-Soft support will recommend upgrading to the latest release before trying to diagnose a problem which may have been fixed between releases.
- Local servers are not allowed to create peer lists. Note that the term "peer list" should be interpreted as meaning "network-wide public peer list". A closed peer list local to the various nodes of an institution does not fall into that category.

A local server can create a network-wide list by means of the Mail-Via=DISTRIBUTE feature. Local servers can submit DISTRIBUTE jobs to the backbone, but will not receive any. If a peer sub-backbone is required for the list (e.g. if large archive files are to be made available), the local LISTSERV operations staff should try to find hosts in the backbone or should join the backbone.

- **Backbone Server** – A backbone server can do all of the above, can create peer lists and is supposed to receive DISTRIBUTE jobs. The restrictions placed on the backbone sites are the following:
  - A backbone server should always be at the latest available level. This means that the operations staff must take whatever steps are necessary to update it in a timely basis. The average delay should not exceed one week, with the deadline being two weeks.
  - A backbone server cannot be placed offline on a regular basis. It must operate 24 hours/7 days. It can of course be placed offline manually under particular conditions, lists can be held by their respective owners, etc.
  - (VM) A backbone server must be AUTOLOG-ed when the system is IPL-ed, and ought to be automatically restarted at regular intervals in case it logs off due to some hardware failure (e.g. paging error). This applies only if such a restart facility is already available at the site hosting the server. In any case, local operators should be able to restart it if they are also able to restart RSCS and other service machines. That is, the host site should do its best to ensure that the server is restarted on a regular basis in case it crashes.



- (Non-VM) A backbone server must start automatically whenever the system is rebooted, and should have some facility to restart if it crashes during operation. As with VM servers, the host site should do its best to ensure that the server is restarted on a regular basis in case it crashes.
- A backbone server should have the latest version of `BITEARN NODES`, or in the worst case, the version from the previous month. This applies only if the `NODUPD` files are received in due time of course.

Sites which are willing to become part of the LISTSERV backbone should indicate it in the `:backbone` tag of the registration form returned to `support@lsoft.com`. However, please note that unless you have a large number of lists, or a number of very large lists, it is probably not necessary for you to join the backbone. Sites running a few small support or hobby lists, for instance, don't need to be on the backbone; sites running hundreds of lists both large and small do need to be on the backbone. Also, sites running one or two huge lists (greater than, say, 50K subscribers each) probably should be on the backbone; such sites should contact L-Soft for more information.

#### 5.6.4 Automatic Registration for LISTSERV Lite Servers

LISTSERV Lite servers are registered automatically when you start the software for the first time. This auto-registration is not optional for Free Edition servers, and can only be disabled for non-Free Edition Lite servers by specifying `STANDALONE` runmode (see `"RUNMODE="` in the [Site Configuration Keyword Reference](#) document).

The auto-registration allows you to take part in the global List of Lists and CataList services maintained by L-Soft. Registrations are verified on a regular basis by a central L-Soft server, which sends out several informational commands that return non-privileged information about your server (anyone can issue these commands). Since these registrations are maintained by regular communication with your server, please note that, should you decommission the server, registration verifications will continue to be mailed to your server for several days until the central server decides that your server is actually gone, and not simply unable to receive mail for some reason. Please note carefully that it is not possible for L-Soft to stop these registration queries manually even if you write to us and tell us that the server has been shut down permanently. They will stop after several days without a response.

### 5.7 Inter-Server Updates

Because networked LISTSERV servers operate as part of a distributed system, they do a certain amount of inter-server "chatting". This "chatting" takes the form of `DISTRIBUTE` jobs, `X-LUPD` jobs, `X-SPAM` jobs, and so forth. Some of the jobs are requests for statistics for the LISTSERV network; other jobs are updates to the list of lists; still other jobs are spam alerts. None of these jobs contain privileged or private information; L-Soft does not query your server for licensing information or any non-LISTSERV-related data, and in fact, all data sent out regarding your server can be retrieved by any user with documented LISTSERV commands.

If you are running LISTSERV Classic, and you do not want to participate in the full-fledged LISTSERV network for whatever reason, you can make a change in your site configuration file to run your server in "standalone" rather than "networked" mode. Simply set the `RUNMODE=` variable to the value "STANDALONE" and stop and restart your

server (see the [Site Configuration Keyword Reference](#) document for the syntax applicable to your OS). Note that this will remove your server and all otherwise-public lists running on it from the CataList and the global List of Lists.

LISTSERV Lite (Free Edition) sites are not allowed to change their runmode. If this is a security issue for your site, L-Soft suggests purchasing either the commercial edition of LISTSERV Lite or LISTSERV Classic and running in "standalone" mode.

## 5.8 Setting Up Archive and Notebook Directories for Use with LISTSERV

We have found that often people get confused about the difference between the directories where the mailing list's notebook archives are kept and the directories where the mailing list's web archive interface files are kept. Here are a few guidelines:

L-Soft's STRONG RECOMMENDATION is that each list be given a separate directory in which its notebook archives and any files available via LISTSERV's file server are kept. On VM this is not always practical, but the security concerns are different and (to date) the 'wa' interface is not available in any case. For OpenVMS, unix, and Windows systems, our STRONG RECOMMENDATION is that a separate directory tree be established for the purpose of storing list notebook archives and other related files. For instance, you might create

*Table 5-3 Creating Archive and Notebook Directories*

System	Location of base directory	Create this directory
OpenVMS	LISTSERV_ROOT:[MAIN]	LISTSERV_ROOT:[LISTS]
unix	/home/listserv	/home/listserv/lists
Windows	C:\LISTSERV	C:\LISTSERV\LISTS

Then, under the main "lists" directory, you would create further subdirectories, one for each list that has archives.

- An example for OpenVMS is `LISTSERV_ROOT:[LISTS.MYLIST-L]`
- An example for unix is `/home/listserv/lists/mylist-l`
- An example for Windows is `C:\LISTSERV\LISTS\MYLIST-L`



**Note:** Under unix, the directory path specification for notebook archives MUST IMPERATIVELY be in lower case. LISTSERV will not write notebooks to directory paths specified in upper- or mixed-case.

Where you locate list archives is particularly important with regard to LISTSERV's file server functions. You MUST NOT set up a file server sub-catalog entry in `site.catalog` that points to LISTSERV's A directory! The catalog owner will be given FULL ACCESS to all the files in the directory you specify in the sub-catalog entry. Therefore in order to maintain security you MUST make a separate directory for each list catalog you define in `site.catalog`. For simplicity's sake, it is generally best to use this directory for the list's notebooks as well as any files the list owners may want to store except for the web archive interface files.



Files generated by LISTSERV for the web archive interface MUST NOT be stored in the notebook directories (or vice versa). You MUST make a separate directory tree where the HTML files and index files for the 'wa' interface are kept. Our best recommendation is to place this directory tree under your web server's document root directory, so that the HTML files for the web archive interface are reached by using a URL such as `http://yourserver/archives/`. The location of this directory naturally varies from platform to platform and web server to web server; the guidelines in Section 5.4.5 [Creating a Subdirectory for the Archive Interface](#) will give you a start on finding this location.

## 5.9 DBMS and Mail Merge Functions

For information on installing and using these features, please refer to the [Advanced Topics Manual for LISTSERV](#).

## 5.10 Synonymous Host Name Registration via ALIASES NAMES

LISTSERV has supported hostname aliases since BITNET added support for this function in 1990. You could define that BITNET node (say) VTVM1 was the same as VPIVM1 and VPIVM2 (old names) and was also known as VTVM1.CC.VT.EDU. Since this was designed into the first major rewrite of LISTSERV, it is very efficient and there is almost no performance penalty. It also works globally, i.e., once the equivalence is defined, it works for all lists and all users.

However, the Internet did not have a similar mechanism for registering aliases, so this function was only useful to BITNET sites, although the underlying code would also have worked with Internet aliases if there had been a way to register them.

LISTSERV supports a centralized list (called ALIASES NAMES) of synonymous Internet hostnames. The main purpose is to address problems with ISPs where the "From:" line is rewritten from (say) "joe@isp.net", which is what Joe wanted, to "joe@alpha.isp.net", "joe@beta.isp.net", "joe@gamma.isp.net" and so forth, where "alpha", "beta" and so on are known machine names (with the understanding that they may add machines from time to time) and "joe" is the same in all cases. In this way it is possible for LISTSERV to match joe@alpha.isp.net with his actual subscribed address of joe@isp.net or any of the other cluster hosts in his domain.

This can also handle a situation where an ISP changed name and for instance "joe@oldname.net" is rewritten to "joe@newname.net". However this does not handle "joe@isp.net" -> "J.Doe@isp.net" and the like.

Requests for additions to ALIASES NAMES are handled by a web form:

<http://www.lsoft.com/regaliases.html>



**Note:** While it is possible to add entries to your local copy for your local host, you should NOT do this as they will not be propagated through the network and will simply be overwritten by the next update.

## 5.11 Real-Time Anti-Virus Scanning

*This feature is not available in LISTSERV Lite.*

*LISTSERV Classic or LISTSERV Classic HPO running on Windows 2000 or later, or on Linux, is required. In addition, current LISTSERV maintenance is required. Other OS platforms may be supported in the future.*

LISTSERV 15 includes optional real-time, on-the-fly anti-virus scanning of all messages sent to LISTSERV mailing lists. All parts of such messages, including inline uuencoded binaries and MIME attachments in those messages, are scanned and bounced back to the sender if viruses are present. Messages sent to the \*-request and owner-\* pseudo-mailbox addresses used by LISTSERV (see Section 17.3.4 [Other Aliases Used by LISTSERV](#)) are also scanned and discarded if they contain viruses. This includes mail sent to the listserv-request and owner-listserv pseudo-mailboxes.

This is a value-added feature which, in addition to a regularly-licensed LISTSERV Classic or LISTSERV Classic-HPO installation, requires the following:

- A "maintenance" LAK in addition to your regular LAK, meaning that you must purchase maintenance (which includes automatic anti-virus signature updates for the term of the LAK) for LISTSERV in order to use the feature; and
- A copy of F-Secure Anti-Virus, which is available for download from L-Soft's web site. This download is free for licensed LISTSERV sites with annual maintenance.

The anti-virus scanning feature includes:

- A 45-day warning when maintenance/AV support is about to expire.
- A 5-day warning when virus databases have not been updated.

The above warnings are controlled by the site configuration variable `LICENSE_WARNING=` as usual. And, as usual, it is not recommended to disable the license warnings as you may miss an expiration date without any warning and/or your anti-virus signature databases may not be kept up to date without your knowledge.

The site configuration variable, `ANTI_VIRUS=`, defaults to 1 (enabled) if the supported AV system is detected and 0 (disabled) if it is not. Attempts to manually enable antivirus scanning by setting the variable to 1 are ignored if the supported AV system is not detected.

The virus checking capability is enabled if (1) the supported AV system is present, (2) a maintenance LAK is loaded and not expired, and (3) `ANTI_VIRUS=0` is not specified in the site configuration file. List owners may not turn off AV checking (design decision -- security). Messages containing viruses are always returned to the sender (design decision – the sender ought to be warned) even if filtering is otherwise enabled. However, attachments which have been filtered are not scanned for viruses (they are simply discarded).

## 5.12 LISTSERV DomainKeys Support

In order for DKIM support to work, we assume that DKIM support has already been configured in DNS for the domains you will be signing for, per the DomainKeys documentation. For your convenience, we have excerpted the relevant section from the Internet Draft for DomainKeys below.

DKIM support is available for LISTSERV Classic and HPO, on all operating systems except for VM. It is not available in LISTSERV Lite.

### 5.12.1 Creating DKIM Keys and Configuring DNS

The following is an excerpt from the controlling Internet Draft for DomainKeys.

### 3.2.2 Public-key signing and verification algorithm

The default signature is an RSA signed SHA1 digest of the complete email.

For ease of explanation, the openssl command is used throughout this document to describe the mechanism by which keys and signatures are managed.

One way to generate a 768 bit private-key suitable for DomainKeys, is to use openssl like this:

```
$ openssl genrsa -out rsa.private 768
```

Which results in the file rsa.private containing the key information similar to this:

```
-----BEGIN RSA PRIVATE KEY-----
MIIBYQIBAAJhAKJ21zDLZ8X1VambQfMXn3LRGKOD5o6lMIgulclWjZwP56LRqdg5
ZX15bhc/GsvW8xW/R5Sh1NnkJNyL/cqY1a+GzzL47t7EXzVc+nRLWT1kwTvFNGIo
AUsFUq+J6+OprwIDAQABAmBOX0UaLdWWusYzNo1++nNZ0RLAtr1/LKMX3tk1MkLH
+Ug13EzB2RZjjDOW1UOY98yxW9/hX05Uc9V5MPo+q2Lzg8wBtyRLq1ORd7pfxYCn
Kapi2RPMcR1CxEJdXOkLCFEcMQDTo0fzuShRvL8q0m5sitIHL1LA/L+0+r9KaSRM/
3WQrmUpV+fAC3C31XGjhHv2EuAkCMQDE5U2nP2ZWV1SbxOKBqX724amoL7rrkUew
ti9TEjfaBndGKF2yYF7/+g53ZowRkfcCME/xOJr58VN17pejSl1T8Icj88wGNHCs
FDWGAH4EKNwDSMnflMG4WMBqd9rzYpkvGQIwLhAHDq2CX4hq2tZAt1zT2yYH7tTb
weiHAQxeHe0RK+x/UuZ2pRhuoSv63mwbMLEZAJAP2vy6Yn+f9SKw2mKuj1zLjEhG
6ppw+nKD50ncnPoP322UMxVNG4Eah0GYJ4DLP0U=
-----END RSA PRIVATE KEY-----
```

(...)

To extract the public-key component from the private-key, use openssl like this:

```
$ openssl rsa -in rsa.private -out rsa.public -pubout -outform PEM
```

Which results in the file rsa.public containing the key information similar to this:

```
-----BEGIN PUBLIC KEY-----
MHwwDQYJKoZIhvcNAQEBBQADAwAwaAJhAKJ21zDLZ8X1VambQfMXn3LRGKOD5o6l
MIgulclWjZwP56LRqdg5ZX15bhc/GsvW8xW/R5Sh1NnkJNyL/cqY1a+GzzL47t7E
XzVc+nRLWT1kwTvFNGIoAUsFUq+J6+OprwIDAQAB
-----END PUBLIC KEY-----
```

This public-key data is placed in the DNS.

(...)

### 3.2.3 Public-key representation in the DNS

There is currently no standard method defined for storing public-keys in the DNS. As an interim measure, the public-key is stored as a TXT record derived from a PEM format [PEM], that is, as a Base64 representation of a DER encoded key. Here is an example of a 768 bit RSA key in PEM form:

```
-----BEGIN PUBLIC KEY-----
MHwwDQYJKoZIhvcNAQEBBQADAwAwAJhAKJ21zDLZ8X1VambQfMXn3LRGKOD5o6l
MIgulclWjZwP56LRqdg5ZX15bhc/GsvW8xW/R5Sh1NnkJNyL/cqY1a+GzzL47t7E
XzVc+nRLWT1kwTvFNGIoAUsFUq+J6+OprwIDAQAB
-----END PUBLIC KEY-----
```

To save scarce DNS packet space and aid extensibility, the PEM wrapping **MUST** be removed and the remaining public-key data along with other attributes relevant to DomainKeys functionality are stored as tag=value pairs separated by semicolons, e.g.:

```
brisbane._domainkey IN TXT "g=; k=rsa; p=MHww ... IDAQAB"
```

Verifiers **MUST** support key sizes of 512, 768, 1024, 1536 and 2048 bits. Signers **MUST** support at least one of the verifier supported key sizes.

The current valid tags are:

g = granularity of the key. If present with a non-zero length value, this value **MUST** exactly match the local part of the sending address. This tag is optional.

The intent of this tag is to constrain which sending address can legitimately use this selector. An email with a sending address that does not match the value of this tag constitutes a failed verification.

k = key type (rsa is the default). Signers and verifiers **MUST** support the 'rsa' key type. This tag is optional.

n = Notes that may be of interest to a human. No interpretation is made by any program. This tag is optional.

p = public-key data, encoded as a Base64 string. An empty value means that this public-key has been revoked. This tag **MUST** be present.

t = a set of flags that define boolean attributes. Valid attributes are:

y = testing mode. This domain is testing DomainKeys and unverified email **MUST NOT** be treated differently from verified email. Recipient systems **MAY** wish to track testing mode results to assist the sender.)

This tag is optional.

## 5.12.2 LISTSERV Configuration

LISTSERV's DKIM support is configured by doing two things.

1. Supply one or more private keys.

Each private key is stored as a text file in LISTSERV's main or home directory (that is, the directory where the \*.list files are) and must be named xxx.dkim, where xxx is the arbitrary name you choose to give the key. If you only use one key, it is recommended to name it default.dkim.

The file is created in the usual openssl/RSA format, with one minor modification. Here is an example:

```
d=example.com; s=test
-----BEGIN RSA PRIVATE KEY-----
MIIBYwIBAAJhAM5MtvnHlLhPzQj itdBctkJTRbJ/
YkbGzcxHP701mHrlMdVeTI3M
(...)
QJEE65afJ4PS8yqM10hZ0p2euKrVZGgUDDdLzgPo2w==
-----END RSA PRIVATE KEY-----
```

The first line in the file must include a specification for the 'd=' and 's=' parameters of the DomainKeys signature (in whatever order, as long as they are both there). Per the controlling Internet Draft for DomainKeys, these variables specify the domain for which you are signing ("d=") and the "selector" that is used to form the query for the public key ("s="). For instance, let's say that your public key is registered as follows in the DNS:

```
brisbane._domainkey.example.com IN TXT "g=; k=rsa; p=MHww ... IDAQB"
```

Please see the DomainKeys documentation for more information.

2. Supply a DKIM\_SIGN Configuration Variable

In your site configuration file, add a DKIM\_SIGN= variable containing a blank-separated list of domains that you are able and willing to sign for. You can use wildcards, but only of the form '\* .EXAMPLE .COM'. You can't use, for instance, 'SALES .EXAMPLE . \*'. For each entry in the list, specify the key to be used, as follows:

```
DKIM_SIGN=EXAMPLE.COM *.EXAMPLE.COM EXAMPLE.CA(CA) *.EXAMPLE.CA(CA)
```

(Under unix, don't forget to export DKIM\_SIGN .)

By default, the key called DEFAULT is used (if one exists). So, in the sample above, the key for EXAMPLE.COM will be fetched from DEFAULT.DKIM whereas the key for EXAMPLE.CA will come out of CA.DKIM.

## 5.12.3 Starting LISTSERV with DKIM Support

LISTSERV loads the keys at startup and makes simple verifications.

```
8 Dec 2005 12:07:42 Loading DomainKeys private keys...
```

```
8 Dec 2005 12:07:42 -> Loaded DEFAULT (d=EXAMPLE.COM; s=TEST; RSA-768)
```

```
8 Dec 2005 12:07:42 -> Loaded CA (d=EXAMPLE.CA; s=TEST; RSA-768)
```

```
8 Dec 2005 12:07:42 DomainKeys support enabled
```

In particular, the 'd=' parameter in the key must match or be a parent of the domain you want to sign for. Thus, the key for EXAMPLE.COM can be used to sign for EXAMPLE.COM and \*.EXAMPLE.COM, but not for EXAMPLE.CA. LISTSERV will skip any invalid entries. Keys are kept in memory so you can have as many as you want.

If there is no DKIM\_SIGN variable or if you are running a LISTSERV version without DKIM support, LISTSERV does not attempt to load any keys and the DKIM feature is bypassed.

### 5.12.4 Using DKIM with LISTSERV

#### With mailing lists:

- Incoming DomainKeys signatures submitted to a mailing list will be suppressed unless "Misc-Options= KEEP\_DKIM\_SIGNATURE" is set in the list configuration.

The KEEP\_DKIM\_SIGNATURE option is experimental and not meant for general use. As DKIM is specified today, signatures DO NOT survive posting to mailing lists (LISTSERV or otherwise), so LISTSERV removes them by default to avoid triggering alerts for Yahoo subscribers.

- For announcement lists, "Misc-Options= ADD\_DKIM\_SIGNATURE" is available to have LISTSERV sign all messages from the list (including digests and indexes). This option also tells LISTSERV to sign administrative messages, welcome messages, and the like for the list.

If desired, both options can be specified for all lists via the DEFAULT\_MISC\_OPTIONS= site configuration variable.

#### In DISTRIBUTE jobs:

A DKIM=NO|YES option is available for the DISTRIBUTE command (default: NO). This will fail if running a LISTSERV version without DKIM support, but otherwise it always succeeds. Messages originating from domains for which LISTSERV has been configured to sign will be signed, while those originating from other domains won't be.

#### In other types of messages:

Once you have become comfortable with DomainKeys signatures, you may want to have LISTSERV sign every message that it generates, regardless of its source. Setting DKIM\_SIGN\_ALL=1 in the site configuration file tells LISTSERV to try to sign every message for which it has a suitable private key, as defined in the DKIM\_SIGN configuration parameter

(Under unix, don't forget to export DKIM\_SIGN\_ALL if you use it.)

### 5.12.5 Restrictions and Implementation Choices

LISTSERV will not sign messages that already have a DomainKeys signature. Double DKIM signatures are disallowed in most cases and, even when allowed, there is a risk that they may not be handled correctly by all implementations. This does not apply if the incoming DomainKeys signature has been discarded (e.g. mailing list without "Misc-

Options= KEEP\_DKIM\_SIGNATURE"). In that case, the message can be signed without risk of false positive.

DKIM can be used to sign mail-merge messages, but in that case LISTSERV's Embedded Mail Merge (EMM) feature **MUST** be enabled. Using EMM is the only way to guarantee that the signing engine will see the exact text being sent to the recipient, and that the signature will match.



## Section 6 LISERV Commands

(For a quick reference card of LISERV commands, see Appendix A: [Command Reference Card](#).)

This section is divided into five parts. The first three correspond to those commands available for use by the general user, list owners and file owners, and the LISERV maintainer. The last two describe how to send commands to LISERV and how to register LISERV passwords. Non-privileged users can send commands by mail or by interactive commands. (Note that interactive commands can only be sent if a two-way NJE or MSGD connection exists.) Privileged users can send commands by mail, interactive commands (subject to the same restriction previously noted) or via the console (VM) or the LCMD utility (non-VM).

Unless otherwise noted, commands are listed in alphabetical order, with the minimum acceptable abbreviation in capital letters. Angle brackets are used to indicate optional parameters. All commands which return a file accept an optional 'F=fformat' keyword (without the quotes) that lets you select the format in which you want the file sent; the default format is normally appropriate in all cases. Some esoteric, historical or seldom-used commands and options have been omitted.



**Note:** Some commands are not available on all platforms; these commands are marked appropriately.

Continuation cards (see the [Advanced Topics Guide for LISERV](#) regarding LISERV's Command Jobs Language) can be used to split long commands (for instance, ADD commands for users with long X.500 addresses) into two or more 80-character cards. In that case you must insert "// " (two slashes followed by a space) before the command text and a comma at the end of each line of the command so that CJLI considers it as a control card and performs the required concatenation. For instance,

```
// QUIET ADD MYLIST someone.with.a.real.long.userid.that.wraps@hishost.com ,
His Name
```

or, for instance, for a large GETPOST job,

```
// GETPOST MYLIST 10769-10770 10772 11079 11086 11095 11099-11100 11104 ,
11111 11115 11118 11121 11124 11131 11144 11147 11153 11158 11166 11168
```

Be sure to put a space before the comma at the end of the first line, as LISERV will not add the space for you.

### 6.1 General Commands

#### 6.1.1 List Subscription Commands

The following commands are listed from most to least important.

**SUBscribe listname [full\_name | ANONYMOUS] [WITH options]**

The SUBscribe command is LISERV's basic command, issued by users to join mailing lists. This command can also be used to change one's "full\_name" field in LISERV's SIGNUP database (simply reissue the command with the changed name). Note that the full\_name is not required if the user has previously signed up to lists on the same



LISTSERV server, or if the user has previously registered in LISTSERV's SIGNUP database by using the REGISTER (q.q.v.) command.

The following syntax:

```
SUBSCRIBE listname ANONYMOUS
```

indicates that the user wishes to join the list anonymously, that is, without specifying a name. The CONCEAL subscription option is automatically set, granting the subscriber the maximal level of protection available.

The following additional syntax:

```
SUBSCRIBE listname full_name WITH option1 option2 ...
```

allows you to "preset" subscription options at subscribe time. For instance, you might want to subscribe to MYLIST-L in order to be able to search its archives, but don't want to receive postings. You would use the command

```
SUBSCRIBE MYLIST-L Joe User WITH NOMAIL
```

Or you might want to receive individual postings with the SUBJecthdr option and receive copies of your own postings instead of the standard acknowledgement that your message was distributed to the list:

```
SUBSCRIBE MYLIST-L Joe User WITH SUBJecthdr REPRO NOACK
```

Also,

```
QUIET SUBSCRIBE listname full_name WITH option1 option2 ...
```

suppresses the command response normally sent by LISTSERV that looks like this:

Confirming:

```
> SUBSCRIBE MYLIST-L Joe User
```

```
You have been added to the MYLIST-L list.
```

**JOIN listname [full\_name | ANONYMOUS] [WITH options]**

JOIN is a synonym for SUBscribe.

**SIGNOFF listname|\*|\* [(NETWIDE)]**

The SIGNOFF command allows the user to cancel his or her subscription to lists. SIGNOFF requires a qualifying parameter, as follows:

```
listname      Sign off of the specified list
*             Sign off of all lists on that server
* (NETWIDE    Sign off of all lists in the LISTSERV network
```

The "\*" (NETWIDE" parameter causes the LISTSERV server to forward a copy of the signoff request to all other registered LISTSERV servers. This is a good option for a user who is changing service providers or otherwise losing a specific address that will not be forwarded. Please note that this parameter will not remove the user from non-LISTSERV lists or from LISTSERV lists running on non-registered sites. It will also not work if the server to which you issue it is running in STANDALONE runmode.

LISTSERV will attempt to sign off the address it finds in the RFC822 "From:" line and will not "fuzzy match" for "similar" addresses. The single exception is when the hostname part of the address is aliased to another hostname in LISTSERV's centrally-maintained ALIASES NAMES file.

**UNSUBscribe listname|\*|\* [(NETWIDE)]**

UNSUBscribe is a synonym for SIGNOFF.

**CHANGE listname|\* newaddr**

This form of the CHANGE command can be used by any subscriber. It must be sent from the currently-subscribed address and results in an OK confirmation request being sent back to that address. This cookie then MUST be confirmed by the currently-subscribed address, exactly as it was entered, or the command will fail. This is the only case where a LISTSERV cookie must be confirmed by a specific address. Note that this assumes that the user still has login access to both addresses, or at least the ability to send mail from the old address.

**SET listname option1 [option2 ...]**

Allows the user to change his or her subscription options without administrative intervention. The options available to be changed are as follows:

- ACK – A mail message acknowledging the receipt and distribution of the user's posting is sent back to the user.
- NOACK – No posting acknowledgement is sent. In general, this setting should only be used if the user has also set himself to REPRO, as it is desirable in most cases that some indication of whether or not the posting was received by LISTSERV be sent.
- MSGack – (Obsolete) An interactive message is sent to acknowledge receipt and distribution. Note that this works only if both the machine running LISTSERV and the user's machine have NJE connectivity (e.g., BITNET). If NJE connectivity is not available on both ends, this option is effectively the same as NOACK.
- CONCEAL – Allows the user to be concealed from the REVIEW command. Note that the list owner or LISTSERV maintainer can always get the complete list of subscribers, regardless of this setting.
- NOCONCEAL – "Unhides" the user.
- Files/NOFiles – (Obsolete) These options toggle the receipt of non-mail files from the list. Note that this is NJE-specific, and thus obsolete for systems without NJE connectivity, but retained for compatibility.
- Mail/NOMail – These options toggle the receipt of mail from the list. Users who will be away from their mail for an extended period of time may prefer to simply turn the mail off rather than to unsubscribe, particularly if subscription to the list is restricted in some way.



**Note:** For backward compatibility, the command SET listname MAIL sent by a user who is set to DIGEST but not also set to NOMAIL will cause the user to be set to NODIGEST (the behavior is identical for users set to INDEX but not to NOMAIL). SET listname MAIL sent by users set to DIGEST/NOMAIL or INDEX/NOMAIL will simply remove the NOMAIL setting and leave the user set to DIGEST or INDEX as the case may be.

- DIGests/INDEX/NODIGests/NOINDEX – These options change the format in which list mail is received by the subscriber. DIGEST turns on digest mode, in which the subscriber receives a digest of postings at set times dependent on how the

"Digest=" keyword of the list is set. INDEX turns on index mode, in which the subscriber receives a daily listing of subjects posted to the list, from which he or she may order postings of interest. NODIGEST and NOINDEX toggle the mode back to individual postings sent as received by LISERV. Note that these options are interrelated; setting one will negate another.

- REPRo/NOREPRo – Causes LISERV to send you a copy of your own postings as they are distributed. Some users may prefer this behavior to the ACK option (see above).
- MIME/NOMIME – Toggles MIME options on and off. Currently only digests are available in MIME format. If DIGEST mode is set, the user will receive a MIME digest instead of the regular plain-text digest. Note that you must have a mail client that supports MIME digests (Pegasus is one that does) or this setting will do you little good. This option is automatically set at subscribe time for users who send their subscription command using a MIME-compliant agent, unless "Default-Options=NOMIME" is specified for the list.
- HTML/NOHTML – Toggle the HTML function for digests and indexes on and off.
- TOPICS: ALL | [+/-] *topicname* – For lists with topics enabled (see the Topics= list header keyword), subscribe or unsubscribe to topics. For instance, if a list has topics SUPPORT and CHAT, a user could subscribe to CHAT by sending SET TOPICS +CHAT. Or the user could unsubscribe to SUPPORT by sending SET TOPICS -SUPPORT. Finally, the user can subscribe to all available topics by sending SET TOPICS ALL.

Options for mail headers of incoming postings (choose one):

- FULLhdr – "Full" mail headers, (default) containing all of the routing information.
- IETFhdr – Internet-style headers.
- SHORThdr – (Obsolete) Short headers (only basic information about the message - Date, From, Subject, To -- is preserved). Setting SHORThdr will break MIME-encoded messages, so it should be used only on lists where MIME and HTML messages are not allowed.
- DUALhdr – Dual short headers, useful with older mail programs which do not preserve the RFC822 return email address. Same caveat as with SHORThdr.
- SUBJecthdr – "Full" mail headers (like the default) except that setting this option tells LISERV to add the list's default subject tag to the subject line of mail coming from the list. (See the listing in the [List Keyword Reference](#) document for "Subject-Tag=" for more information.) Note that if the user is set to SHORThdr (or any other header option other than FULLhdr), LISERV will automatically switch the user to FULLhdr, as subject tags require full headers. A subject tag is generated (for subscribers with the SUBJecthdr option set) even if the original message had no "Subject:" header. To turn the subject tagging off, the user simply sends a new SET command with any of the other header options (e.g., SET *listname* FULLhdr) and the SUBJecthdr option is reset.

- `FULL822` – Essentially the same as "full" mail headers, but with the important difference that the recipient's email address is specified in the "To:" line rather than the address of the list. "`FULL822`" headers should be used with extreme caution, as they cause LISERSERV to create a separate mail envelope with a single RFC821 RCPT TO: for each address so set. This behavior can significantly affect the performance of both LISERSERV and of your external mail system.
- `SHORT822` – Essentially the same as "short" mail headers, with the same caveats as noted for `FULL822`.



**Note:** `FULL822` and `SHORT822` headers should only be used if a specific problem indicates that they might solve the problem. One possible use would be to determine which subscriber from a specific site is causing the site to throw back delivery errors if that site does not specify which RCPT TO: is generating the error. These headers should never be used by default.



**Documented Restriction:** The use of the `SHORTHDR` or `DUALHDR` options will break messages that depend on MIME encoding, because these options strip the RFC822 headers that identify the message as a MIME message. `SHORTHDR` and `DUALHDR` were designed for the non-MIME mail clients which prevailed in LISERSERV's early history. As most mail clients today support MIME, the use of these options is now deprecated.

`CONFIRM listname1 [listname2 ]...`

The `CONFIRM` command should be issued when LISERSERV requests it. A request for `CONFIRM` should not be confused with a "command confirmation request" which requires an "OK" response. The `CONFIRM` command is used in two cases:

- When the list in question requires periodic subscription renewals (controlled by the `Renewal= keyword`). In this case, the amount of time between the request for confirmation and termination of the subscription is controlled by the `Delay()` parameter of the `Renewal= keyword`; the default is seven days.
- When LISERSERV's automatic address probing function fails and you receive a message to that effect. The response time is controlled by the settings of the `Auto-Delete= keyword` for the list in question.

### 6.1.2 Other list-related commands

`INDEX [listname]`

The `INDEX` command sent to LISERSERV without further qualification sends back the contents of the "root" level archive filelist on VM systems (`LISTSERV FILELIST`) or archive catalog on non-VM systems (`SITE.CATALOG` plus the contents of `SYSTEM.CATALOG`).

If the `INDEX` command is sent with the name of a list (e.g., `INDEX MYLIST`) or the name of a special filelist or catalog file (e.g., `INDEX TOOLS`, if `TOOLS FILELIST` on VM or `TOOLS.CATALOG` on non-VM exists), LISERSERV sends back the contents of the specified filelist or catalog. Several possibilities exist:

- For mailing lists without an associated filelist or catalog, LISTSERV creates an index "on the fly" containing entries for the accumulated notebook archives for that list. If notebook archives are not enabled for the list, LISTSERV will respond, "This server does not have any file by the name '*listname.filelist*'."
- For mailing lists with an associated filelist or catalog, LISTSERV will append the "on the fly" index of notebook archives to the entries in the associated filelist or catalog. For instance, for a list called MYLIST with associated catalog MYLIST.CATALOG, INDEX MYLIST might return:

Table 6-1 Sample Output of an INDEX Listname Command

```

*
* MYLIST FILELIST from LISTSERV@LISTSERV.MYCORP.COM
*
* ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
*
* The GET/PUT authorization codes shown with each file entry describe
* who is authorized to GET or PUT the file:
*
*     ALL = Everybody
*     CTL = LISTSERV administrators
*     OWN = List owners
*     PRV = Private, ie list members
*     LMC = LISTSERV master coordinator
*     N/A = Not applicable - file is internally maintained by LISTSERV
*     MSC = Miscellaneous - contact administrator for more information
*
* ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
*
* Information files for MYLIST
*
* filename      filetype      GET PUT size (bytes) date      time
* -----
* MYLIST        FAQ           ALL MSC      22,528 1996-02-09 21:30:10
* MYLIST        WELCOME      ALL MSC        279 1998-02-02 09:59:44
* MYLIST        FAREWELL     ALL MSC         92 1998-02-05 11:06:14
*
* Archive files for the MYLIST list at LISTSERV.MYCORP.COM
* (monthly logs)
*
* filename      filetype      GET PUT size (bytes) date      time
* -----
* MYLIST        LOG9603      LOG OWN       8,668 1998-05-27 15:29:57
* MYLIST        LOG9605      LOG OWN       7,865 1998-06-29 08:43:26
* MYLIST        LOG9606      LOG OWN      17,298 1998-07-23 12:46:20

```

- Lastly, for catalogs or filelists without an associated list, the INDEX command returns only the entries in the catalog or filelist, since there are no associated list archives to be indexed.

Under VM, instead of the size in bytes, three separate VM-specific columns are used. Please note the following definitions for them:

- `rec -fm` – Indicates whether the file is in a fixed or variable record format.
- `lrecl` – Logical record length. For a file with fixed record format (F), this is the length of each record. For a file with variable record format (V), this is the maximum record length.
- `nrecs` – Number of records (lines) in the file.

#### **Lists [option]**

Access the global list of lists maintained by LISTSERV. If no options are specified, then LISTSERV returns only local lists, one line per list. The available options are:

- `Detailed` – All local lists, complete with full header information.
- `Global xyz` – Only those whose name or title contains 'xyz'.
- `SUMmary [host]` – Membership summary for all lists on specified host, or the host to which the command is sent if no host is specified.
- `SUMmary ALL` – Membership summary for all hosts (long output, send request via mail).
- `SUMmary TOTAL` – Membership totals only.

"Lists Global" without a search string, which returns the entire list of lists, may no longer be issued by general users. General users should use the "Lists Global /xyz" format to search the list of lists, or (preferably) use L-Soft's CataList service at <http://www.lsoft.com/catalist.html>.

"Lists SUMmary", when issued to an unregistered host or to a host running in STANDALONE mode will generate the response "No information available yet - please try again later." because the file required for this function does not exist.

#### **Query listname**

Query your subscription options for a particular list (use the SET command to change them). Using the "\*" wildcard in place of the name of a single list queries subscription options on all lists on the server.

#### **REGister full\_name | OFF**

Register's the user's full name field in LISTSERV's SIGNUP files, or changes the current value of that field. When a user's name is registered, he or she can omit the full name field from subsequent SUBscribe requests to that server. Sending "REGISTER OFF" to LISTSERV deletes the user's entry from the SIGNUP file.

#### **REView listname [(options)]**

Get information about a list, assuming the list header keyword "Review=" is set appropriately. For general users, REVIEW does not return address information about subscribers who are set to CONCEAL. The options are:

- `BY sort_field` – Sort list in a certain order:
  - `Country` – Sorts by country of origin (ISO country codes).
  - `Date` – Sorts by subscription date (newest to oldest).

- Name – Sorts by user name (last, then first).
- NODEid – Sorts by hostname/nodeid
- Userid – Sorts by userid
- BY (*sort\_field1 sort\_field2*) – You can specify more than one sort field if enclosed in parentheses. For instance, BY (NODE NAME).
- Countries – Synonym of BY COUNTRY.
- Topics – Adds a breakdown of subscribers per topic (if TOPICS= is defined in the list header) at the end of the subscriber list. If you just want the breakdown, use REVIEW *listname* SHORT TOPICS. This does not show topics by individual subscribers (see the QUERY command instead). If TOPICS= is not enabled for a given list then this option is ignored.
- LOCAL – Don't forward request to peers. This is only useful if the list is peered; normally it should not be necessary to issue this option.
- Msg – Send reply via interactive messages (BITNET users only).
- NOHeader – Don't send list header, just send the subscriber list.
- Short – Don't list subscribers, just send the header and the membership summary for the list.



**Note:** You can get a quick read of the number of subscribers on the list by sending the command REVIEW *listname* SHORT NOHEADER.

List owners and site maintainers may also use the additional option:

- ALL – List concealed members (who will show up with "[concealed]" next to their entry) as well as non-concealed members. (NOT available to general users even if Review= Public.)

#### SCAN *listname text*

Scan a list's membership for a name or address. Helpful if a user attempts to send a SET command or an UNSUB command and is informed that their address is not subscribed to the list. At the non-privileged level, this command will show all non-concealed entries that match the search text, assuming that the list is set to "Review= Public".

Stats *listname* [(options)]

*The following command is available on VM servers only.*

Get statistics about a list. This command is VM-specific, and was originally intended to return BITNET data. The single option is:

- LOCAL – Doesn't forward to peers.

### 6.1.3 Informational commands

#### Help

Obtain a list of commonly-used LISYSERV commands. Also explains how to get the comprehensive reference card and tells who the (non-hidden) server manager(s) are.



**Info** [**topic**|**listname**]

Order a LISERSERV manual, or get a list of available ones (if no topic was specified); or get information about a list. For Info listname, the text in the INFO template form of listname.MAILTPL is used; however, if listname.MAILTPL does not exist or does not contain an INFO template form, the INFO template form of DEFAULT.MAILTPL is used.

**Query File** **fn ft** [**filelist**] [(**options**)]

(Available only on VM) Get date/time of last update of a file, and GET/PUT file access code. The single option is:

- **Flags** – Get additional technical data (useful when reporting problems to experts).

**RELEASE**

Find out who maintains the server, the version and build date of the software, and the version of LISERSERV's network data files.

**SHOW** [**function**]

Display information as follows:

- **ALIAS** *node1* [*node2* [...]] – BITNET nodeid to Internet hostname mapping.
- **DISTRIBUTE** – Statistics about DISTRIBUTE.
- **HARDWARE** or **HW** – Hardware information; what kind of machine is LISERSERV running on?
- **LICENSE** – License/capacity information and software build date.
- **LINKS** [*node1* [*node2* [...]]] – Network links at the BITNET node(s) in question.
- **NADS** [*node1* [*node2* [...]]] – Addresses LISERSERV recognizes as node administrators for the specified site(s).
- **NODENTRY** [*node1* [*node2* [...]]] – BITEARN NODES entry for the specified node(s).
- **NODENTRY** *node1* /*abc*\*/*xyz* – Just the ':xyz.' tag and all tags whose name starts with '*abc*'.
- **POINTS** [ALL | *list1* [*list2*...]] – Graduated (LISTSERV Classic) license point information. This information can help you plan orderly expansion of your site if you are running with a graduated LISERSERV Classic license. Under Lite this command shows Classic point usage.
- **STATS** – Usage statistics for the server (this is the default option).
- **VERSION** – Same output as RELEASE command.

If no function is specified, the output is per SHOW STATS.

*The following options are available for VM servers only:*

- **BITEARN** – Statistics about the BITEARN NODES file.
- **DPATHS** *host1* [*host2* [...]] – DISTRIBUTE path from that server to specified host(s).



- `DPATHs *` – Full DISTRIBUTE path tree.
- `FIXes` – List of fixes installed on the server (non-VM see SHOW LICENSE).
- `NETwork` – Statistics about the NJE network.
- `PATHs snode node1 [node2 [...]]` – BITNET path between 'snode' and the specified node(s).

### 6.1.4 Commands Related to File Server and Web Functions

#### GET

- VM Syntax

```
GET fn ft [filelist] [(options) [F=fformat] [SPLIT=integer]]
```

- Non-VM Syntax

```
GET fn ft [catalogname] [(F=fformat) [SPLIT=integer]]
```

Order the specified file or package from LISYSERV. The single option is for VM servers only and is:

- `PROLOGtext xxxx` – Specify a 'prolog text' to be inserted on top of the file

Examples:

```
GET MYFILE TEXT
GET MYFILE TEXT MYLIST-L
```

Typically the filelist name or catalog name is the same as the name of the list to which the files belong. A password (`PW=xxxxx` at the end of the command) is required to retrieve any file that does not have a GET FAC of ALL. For more information on FAC (File Access Control) codes, see Section 8.3.5 [File Access Codes \(FAC\) for User Access \(VM Systems Only\)](#) or Section 8.4.1 [Adding Files to the SITE.CATALOG](#).

Do not use dots (periods) in the file specification when specifying the filelist or catalog name, as this will result in an error. For instance

```
GET MYFILE.TEXT MYLIST-L
```

will result in an error, whereas

```
GET MYFILE TEXT MYLIST-L
```

will not.

To control the format in which LISYSERV returns the file(s) to you, you can specify the `F=fformat` parameter. Supported formats are `Netdata`, `Card`, `Disk`, `Punch`, `LPunch`, `UUencode`, `XXencode`, `VMSdump`, `MIME/text`, `MIME/ Appl`, `Mail`.

To split very large files into manageable chunks, you can specify the `SPLIT=integer` parameter. The integer value is the size you want the chunks to be generated, in kilobytes. For instance if you were ordering a 2MB notebook log and wanted to break it into 100KB chunks, you would specify `SPLIT=100`. This is

handy for people whose mail systems place a limit on the size of an individual mail message that may be received by a given user.

The following syntax is accepted for **TCPGUI parameters for Change-Logs**:

```
GET listname CHANGELOG (MSG [COLUMNS(colspec1 colfilter1
[colspec2 colfilter2[...]])
```

The design goal was to provide access via the TCPGUI (and thus the web interface) for change-log data. Further information can be found in the [Advanced Topics Guide for LISYSERV](#).

#### **GIVE**

- **VM Syntax:** GIVE fn ft [filelist] [TO] userid@host
- **Non-VM Syntax:**

```
GIVE fn.ft [TO] userid@host
```

or

```
GIVE fn ft catalogname [TO] userid@host
```

Sends a file stored in a LISYSERV file archive to someone else. For instance, you may want to send LISYSERV REFCARD to a new user. Rather than retrieving LISYSERV REFCARD and then forwarding it to the user, you simply issue a GIVE command to tell LISYSERV to send it directly. Note that the token "TO" is optional.

Examples:

- For LISYSERV running under VM:

```
GIVE LISYSERV REFCARD joenewuser@hishost.com
GIVE LISYSERV REFCARD TO joenewuser@hishost.com

GIVE README TEXT MYLIST-L joenewuser@hishost.com
GIVE README TEXT MYLIST-L TO joenewuser@hishost.com
```

- For LISYSERV running on non-VM hosts there are two syntaxes, depending on whether or not you need to specify a catalog name for the file in question. Note that the only real difference is whether or not you are required to specify a dot between the filename and the extension.

```
GIVE LISYSERV.REFCARD joenewuser@hishost.com
GIVE LISYSERV.REFCARD TO joenewuser@hishost.com

GIVE README TXT MYLIST-L joenewuser@hishost.com
GIVE README TXT MYLIST-L TO joenewuser@hishost.com
```

#### **INDEX [filelist|catalog]**

Same as GET xxxx FILELIST. If no filelist is specified, the default is LISYSERV FILELIST (on non-VM, SITE CATALOG is returned as LISYSERV FILELIST in this case).

#### **PW function**

Define/change a "personal password" for protecting AFD/FUI subscriptions, authenticating PUT commands, and so on.

- `ADD firstpw` – Define a password for the first time, or after a PW RESET. Requires confirmation via the "OK" confirmation method.
- `CHange newpw [PW=oldpw]` – Change your existing password. If you do not include your old password for authentication, LISTSERV will require confirmation via the "OK" confirmation method.
- `REP password` – Starting with 1.8d, this function is a hybrid of "ADD" and "CHange". If a password does not exist for the user, one will be added. If a password does exist for the user, it will be changed (with confirmation required via the "OK" confirmation method). "REP" was added primarily for use by the web archive and administration interface but can be used in e-mailed PW commands as well.
- `RESET` – Reset (delete) your password. This function always requires confirmation via the "OK" confirmation method.

**SENDme**

Same as GET.

**AFD**

*Available on VM servers only.*

Automatic File Distribution. The functions are as follows:

- `ADD fn ft [filelist [prolog]]` – Add file or generic entry to your AFD list.
- `DELeTe fn ft [filelist]` – Delete file(s) from your AFD list (wildcards are supported).
- `List` – Displays your AFD list.

For node administrators:

- `FOR user ADD/DEL/LIST etc` – Perform requested function on behalf of a user you have control over (wildcards are supported for DEL and LIST).

**FUI**

*Available on VM servers only.*

File Update Information: same syntax as AFD, except that FUI ADD accepts no 'prolog text'.

**6.1.5 Other Advanced Commands****DISTRIBUTE type source dest [options]**

**Note:** Starting with 1.8d, the ability to send DISTRIBUTE jobs is limited to LISTSERV Maintainers by default, and requires a password. This section is retained for compatibility with 1.8c and earlier, and for 1.8d and later servers which have the DISTRIBUTE security feature turned off.

Distribute a file or a mail message to a list of users (see the Advanced Topics Guide for LISTSERV for more details on the syntax). The various parameters are, briefly:

Type:

- **MAIL** – Data is a mail message, and recipients are defined by '<dest>'.
- **MAIL-MERGE** – Data is a mail-merge message. See the [Advanced Topics Guide for LISTSERV](#) for specifics.
- **FILE** – Data is not mail, recipients are defined by '<dest>'.
- **POST** – (non-VM only) Same as MAIL except that the message is pre-approved. See the [Advanced Topics Guide for LISTSERV](#) for specifics.
- **RFC822** – Data is mail and recipients are defined by the RFC822 'To:/'cc:' fields.

**Source:**

- **DD=ddname** – Name of DD holding the data to distribute (default: 'DD=DATA').

**Dest:**

- **<TO> user1 <user2 <...>>** – List of recipients.
- **<TO> DD=ddname** – Use a DD called ddname for the destination addresses, one recipient per line.

**Options for the general user:**

- **ACK=NOne/MAIL/MSG** – Acknowledgement level (default: ACK=NONE).
- **CANON=YES** – 'TO' list in 'canonical' form (uid1 host1 uid2 host2...).
- **DEBUG=YES** – Do not actually perform the distribution; returns debug path information.
- **INFORM=MAIL** – Send file delivery message to recipients via mail.
- **TRACE=YES** – Same as DEBUG=YES, but file is actually distributed.
- **AV=YES [, FORCE]** – Check the message for viruses. See the [Advanced Topics Guide for LISTSERV](#) for specifics.
- **DKIM=NO/YES** – Sign the message with a DomainKeys signature (default: DKIM=NO) See Section 5.12 [LISTSERV DomainKeys Support](#) for specifics.

**Options requiring privileges:**

- **FROM=user** – File originator
- **FROM=DD=ddname** – One line: 'address name's
- **PRE-APPROVED=YES** – Pre-approve message (with DISTRIBUTE POST only)

**GETPost listname post\_number [post\_number [...]] [NOMIME]**

GETPost is used after receiving the output of a SEARCh command to retrieve the postings you want from the SEARCh output. For instance, if you want postings numbered 1730, 1731, 1732, and 1840 from the MYLIST list, send the command

```
GETPost MYLIST 1730-1732 1840
```

GETPost is analogous to the VM database command PRINT.

In previous versions, the GETPost command returned messages that contained MIME attachments in their "raw" form, which could not be extracted automatically by MIME-

aware mail clients. Customers who wished to use list notebooks to archive word-processing documents (for instance) found this to be a problem. From LISERV 1.8e, attachments returned in messages by way of the GETPost command will now display as inline clickable links in the individual messages.

Users of certain email clients (specifically Pine, which handles attachments in a secondary viewing area) may find the new format difficult to use. If preferred, the pre-1.8e behavior may be reverted to by specifying "NOMIME" as the last parameter of the GETPost command.

### Search

For lists running on VM servers, see also below at DATABASE.

The Search command syntax is similar to that of the SEARCH/SELECT commands in the "old" database functions. A very basic Search command for list MYLIST would look like this:

```
Search search_string IN MYLIST
```

You can also restrict your search by date, sender, or other criteria, for example,

```
Search search_string IN MYLIST SINCE 96/01/01
```

```
Search search_string IN MYLIST WHERE SENDER CONTAINS ERIC
```

The specific syntax is outlined in LISTDB MEMO (available from LISERV with the command "INFO DATABASE") and in the [Advanced Topics Guide for LISERV](#).



**Note:** The "new" Search command does not require a CJLI job framework to operate; simply send the Search command in the body of an email message to the appropriate server. LISERV will respond with an index of the postings matching your criteria and instructions on how to use the GETPost command to retrieve the posts you want.

### SERVE user

Restore service to a user whose access to LISERV has been disabled. This generally occurs when a user has sent 51 incorrect commands in a row to LISERV, which LISERV interprets as a possible mail loop. (Note also that certain mail packages that send "Read:/Not Read:" notifications back to LISERV will trigger this scenario after 51 iterations. The best solution would be for the user to disable receipt notifications.) The user in question cannot restore his or her own service; this command must be issued from another userid. Note that if the user has been manually served out by privileged user (a LISERV maintainer), the SERVE command must be issued by a similarly-privileged user (who must also be a LISERV maintainer, although naturally the same user who issued the SERVE OFF command can issue the SERVE command).



**Note:** The THANKS command will not reset the serve-off counter (so vacation messages or auto-replies that contain a sentence starting with something like "Thanks for writing" will not defeat the system and users sending them will eventually be served off instead of continuing to loop ad infinitum).

### THANKS

You can send this command to check to see if the server is alive. If it is, the server politely responds, "You're welcome!".

**DATABASE function**

*This command is only available on VM servers:*

Access LISYSERV database(s). The functions are explained in detail in the version of LISTDB MEMO available from VM servers, but the basic syntax is:

- `Search DD=ddname <ECHO=NO>` – Perform database search (see the VM version of LISTDB MEMO for more information on this)
- `List` – Get a list of databases available from that server
- `REFRESH dbname` – Refresh database index, if suitably privileged

**Dbase**

*This command is only available on VM servers:*

Same as DATABASE

**6.2 List Owner and File Owner Commands****6.2.1 File Management Commands (for file owners only)**

```
PUT fn ft <filelist <NODIST>>
```

Update a file you own. Options are:

- `<PW=password>` – Supply your password for command authentication.

*The following options are VM-specific and will not work on the non-VM servers.*

The "NODIST" option prevents AFD and FUI distributions when the file is updated. Other available VM only options include:

- `<CKDATE=NO>` – Accept request even if the current version of the file is more recent than the version you sent
- `<DATE=yymmddhhmmss>` – Set file date/time.
- `<RECFM=F <LRECL=nnn>>` – Select fixed-format file (not to be used for text files).
- `<REPLY-TO=userid>` – Send reply to another user.
- `<REPLY-TO=NONE>` – Don't send any reply.
- `<REPLY-VIA=MSG>` – Request reply via interactive messages, not mail (Requires NJE connectivity).
- `<"parameters">` – Special parameters passed to FAVE routine, if any.

Standard parameters supported for all files:

- `TITLE=file title` – Change file "title" in filelist entry.

**AFD/FUI**

*This command is only available on VM servers.*

Automatic File Distribution privileged commands. In addition to the AFD/FUI functions listed above, a file owner may use the following function:

- `GET fn ft <filelist>` – Get a list of people subscribed to a file you own.

**GET *fn* FILELIST <(options)>**

*This command is only available on VM servers.*

Special options for filelists:

- CTL – Return filelist in a format suitable for editing and storing back.
- NOLOCK – Don't lock filelist (use in conjunction with CTL).

**REFRESH *filelist* <(options)>**

*This command is only available on VM servers.*

Refresh a filelist you own. The single option is:

- NOFLAG – Don't flag files which have changed since last time as updated (for AFD/FUI).

**UNLOCK *fn* FILELIST**

*This command is only available on VM servers.*

Unlock filelist after a GET with the CTL option if you decide not to update it after all

## 6.2.2 List Management Functions

Commands that support the QUIET keyword are marked with an asterisk (\*).

**ADD(\*) *listname user [full\_name]***

**ADD(\*) *listname DD=ddname [IMPORT [PRELOAD]]***

The first syntax is used to add an individual user to one of your lists, or update his name field. Note that you can substitute an asterisk ("\*") for *full\_name* and LISERSERV will substitute "<No name available>" in the list.

The second syntax is used for bulk ADD operations where a dataset (DD=ddname) is used add multiple users, one address/name pair per line. For bulk operations you may also use the IMPORT option, which implies a QUIET ADD (in other words you do not need to specify QUIET if you use IMPORT) and otherwise vastly speeds up the ADD process by loosening syntax checking and omitting success messages. The IMPORT PRELOAD option is used to direct LISERSERV to preload the existing e-mail keys in memory before starting the transaction, which speeds the operation up considerably. This option is used primarily with DBMS lists to speed up bulk adds. PRELOAD is not necessary for traditional LISERSERV lists and does not normally lead to a significant performance improvement. However, when importing a new list (no existing subscribers), it does reduce CPU usage somewhat.

For a bulk ADD operation, the users are defined in a separate dataset beginning on the line following the ADD command. For instance,

```
ADD listname DD=ddname IMPORT
//ddname DD *
userid@host.com User Name
userid2@host.com User2 Name
... more address/name pairs, one per line ...
/*
```



Please see Section 7.17 [Bulk Operations \(ADD and DELETE\)](#) for specific instructions for bulk ADD operations.

**ADDHere (\*)**

Same as **ADD**, but means "add the user on this peer, do not forward this request to a (possibly) closer peer". For non-peered lists, is functionally identical to **ADD**.

**CHANGE(\*) listname|\* newaddr**

**CHANGE(\*) listname|\* oldaddr|pattern newaddr|\*@newhost**

The first form can be used by any subscriber and results in a cookie being sent to the new address. This cookie **MUST** be confirmed by the new address, exactly as it was entered, or the command will fail. This is the only case where a **LISTSERV** cookie must be confirmed by a specific address.

The list owner form does not use cookies but simply applies the standard "Validate=" rules (as for a **DELETE** command). You can specify a wildcard pattern for the old address and **\*@newhost** for the new address to rename certain addresses to a new hostname. The **CHANGE1** template is sent unless you specify **QUIET**.

Change log entries are made (**CHANGE oldaddr newaddr**) and there is a **CHG\_REQ** exit point which allows you to reject the operation.

**DELeTe(\*) listname user [(options)]**

**DELeTe(\*) listname DD=ddname [BRIEF]**

The first syntax is used to remove a single user from one of your lists, or from all local lists if listname is **'\***. The available options are:

- **Global** – Forward request to all peers
- **LOCaL** – Don't try to forward request to closest peer if not found locally
- **TEST** – Do not actually perform any deletion (useful to test wildcard patterns)

The second syntax is used for bulk **DELETE** operations (similar to a bulk **ADD** operation). See Section 7.17 [Bulk Operations \(ADD and DELETE\)](#) for details. The single available option is:

- **BRIEF** – Good for deleting wildcard patterns (such as **\*@\***) when you don't want a "userid@host has been deleted from list xxxx" for each user deleted. Returns instead only a count of the users that were deleted.

**FREE listname <(options)>**

Release a held list. The single option is:

- **Global** – Forward request to all peers.

**GET listname <(options)>**

Get a copy of a list in a form suitable for editing and storing the list and lock it so that other list owners can't modify it until you store it back (or until you or they issue an **UNLOCK** command). The options are:

- **Global** – Forward request to all peers.
- **HEADer** – Send just the header; on the way back, only the header will be updated. This is the recommended way to modify your list header.



- `NOLock` – Do not lock the list.
- `OLD` – Recover the "old" copy of the list (before the last PUT).

**HOLD listname <(options)>**

Hold a list, preventing new postings from being processed until a FREE command is sent. The single option is:

- `Global` – Forward request to all peers

**Lists [option]**

Additional options available for list owners and moderators:

- `OWNed` – Returns a list of local lists owned by the invoker.
- `MODerated` – Returns a list of local lists that are moderated by the invoker.

**MOVE(\*) listname user <TO> node**

Move a subscriber to another peer. Do NOT use this command to move users from one list host site to another during migration. It is strictly for moving subscribers from one peer to another peer.

- `listname DD=ddname` – Move several subscribers to various peers

**PUT listname LIST**

Update a list from the file returned by a GET command. This is the standard "PUT command" or "list PUT" referred to throughout this document.

Use of the PUT command to store a list header with new subscriber data at the bottom (e.g., an attempt to add subscribers "on the fly") will result in only the header of the list being stored, and in the generation of the following warning:



**Warning:** New subscriber data was found in the replacement list you sent, possibly due to the use of a signature file with an unusual separator line. If you really meant to update the subscriber data, please resend your request with the word "PUT" replaced with "PUTALL". For now, only the list header will be updated.

**PUTALL listname LIST**

This command allows you to PUT an entire list file, that is, the list header followed by the list of subscribers.

**Documented Restriction:** `PUTALL` does not work with DBMS lists; only the header information is replaced. Subscriber information in the DBMS table is not changed. For DBMS lists where the subscriber information needs to be replaced *in toto*, either the DBMS should be manipulated with your regular DBMS tools or you should use `ADD IMPORT`.

**Query listname <WITH options> FOR user**

Query the subscription options of another user (wildcards are supported).

- `* <WITH options> FOR user` – Searches all the lists you own for the specified user(s) with the specified option(s).

**SET(\*) listname options <FOR user>**

Alter the subscription options for other users (wildcards are supported when setting options for another user or set of users).

Additional options for list owners:

- NORENEW/RENEW – Waive subscription confirmation for this user.
- NOPOST/POST – Prevent user from posting to list.
- EDITor/NOEDITor – User may post without going through moderator.
- REView/NOREView – Postings from user go to list owner or moderator even if user is otherwise allowed to post.

**UNLOCK listname**

Unlock a list after a GET, if you decide not to update it after all, or unlock a list if it has been locked by another list owner or by the LISTSERV maintainer. Note that if you are not the person who originally locked the list, it is considered good practice to ask the person who originally locked the list whether or not they are done with the list before you unlock it.

**EXPLODE listname <(options)>**

*This command is only available on VM servers.*

Examine list and suggest better placement of recipients, returning a ready-to-submit MOVE job.

- BESTpeers *n* – Suggest the N best possible peers to add.
- Detailed – More detailed analysis.
- FOR *node* – Perform analysis as though local node were 'node'.
- PREFer *node* – Preferred peer in case of tie (equidistant peers).
- SERvice – Check to see that service areas are respected.
- With(*node1* <*node2* <...>>>) – Perform analysis as though specified nodes ran a peer.
- WITHOut(*node1* <*node2* <...>>>) – Opposite effect.

**Stats listname (RESET)**

*This command is only available on VM servers.*

Resets (BITNET) statistics for the list.

### 6.3 LISTSERV Maintainer Commands

All LISTSERV maintainer commands require a password for validation when issued by email. Commands issued by TELL or SEND from the local host or via the LCMD utility do not require password validation. (Commands issued by LCMDX do require password validation. LCMDX, the LISTSERV TCPGUI demonstration program, is not the same as the LCMD utility shipped with LISTSERV.)

**FOR user command**

Execute a command on behalf of another user (LISTSERV maintainers only). Note that this command is provided for debugging purposes only -- it provides a method for a LISTSERV maintainer to send commands "from" the specified user. It is not recommended to use this command syntax in production, for instance to issue SET or SUBSCRIBE or UNSUBSCRIBE commands on a user's behalf. For instance, the LISTSERV maintainer should use, respectively, the "SET listname options FOR userid@host", "ADD listname userid@host", or "DELeTe listname userid@host" syntaxes in preference to the "FOR userid@host command" syntax.

**Lists [option]**

- **Global** – All known lists, one line per list, sent as a (large!) file. Only LISTSERV maintainers may request this list, as it has become a favorite pastime of Internet mailbombers to issue LIST GLOBAL commands on behalf of users whose mailboxes they wish to bomb. You should direct users who request "the whole list of lists" to L-Soft's CataList service at <http://www.lsoft.com/catalist.html>.

Additional options available for site maintainers are:

- **OWNED BY internet\_address** – Returns a list of local lists owned by the userid@host specified. Wildcards are acceptable.
- **MODerated BY internet\_address** – Returns a list of local lists moderated by the userid@host specified. Wildcards are acceptable.

**NODESGEN [WTONLY]**

Regenerate all LISTSERV network tables, or just compile the links weight file (debugging command). This happens automatically when LISTSERV is rebooted if a new BITEARN NODES file is found. Otherwise you should issue a NODESGEN whenever you update BITEARN NODES.

**PUT listname LIST**

Create a new list. Requires the CREATEPW for validation when issued from a remote node. You may specify initial subscribers, one per line, following the list header when creating a list. See also the PUTALL command in Section 6.2.2 [List Management Functions](#).

**PWC function**

Password file management:

- **ADD user newpw** – Define a password for the specified user.
- **DELeTe user** – Delete password for that user.
- **Query user** – Query the password of the specified user.

**REGister name|OFF FOR user**

Set or delete a user's SIGNUP FILE entry.

**SERVE user OFF [DROP] | LIST**

*SERVE user OFF* permanently suspends access from an abusive user or gateway (restore service with *SERVE user*).

Adding "DROP" (for example, `SERVE user OFF DROP`) to the command is identical to `SERVE user OFF` except that the postmaster will not receive any notification messages from LISTSERV when/if the user continues to try to post.

Issuing a `SERVE LIST` command causes LISTSERV to return a list of all users who are currently served off or who are spam-quarantined. For instance,

```
> serve list
JOE@EXAMPLE.COM          DROP 2003-08-20 15:51:20 by nathan@EXAMPLE.COM
FOOBAR@EXAMPLE.EDU      HARD 2003-04-07 14:55:29 by NATHAN@EXAMPLE.COM
BLAB@FOO.EXAMPLE.COM    SOFT 2004-09-14 10:53:18
SPAMMER@SPAMDOMAIN.COM  SPAM 2003-08-20 15:50:55

4 matching entries.
```

### **SHUTDOWN [REBOOT|REIPL]**

Stop the server, and (optionally) restart it immediately (by specifying either `REBOOT` or `REIPL` -- the two options are synonymous). Starting with LISTSERV 15.0, this is available on all platforms.

#### **STOP**

Same as `SHUTDOWN`.

#### **CMS *command\_text***

*This command is only available on VM servers.*

Issue a CMS command and get the last 20 lines of response sent back to you, the rest being available from the console log.

#### **CP *command\_text***

*This command is only available on VM servers.*

Issue a CP command and get up to 8k of response data sent to you (the rest is lost).

#### **DATAbase *function***

*This command is only available on VM servers.*

Control operation of databases:

- `DISABLE` – Disable interactive database access, without shutting down existing sessions
- `ENABLE` – Re-enable interactive access
- `SHUTDOWN` – Shut down all interactive database sessions, and disable interactive access

#### **INSTALL *function***

*This command is only available on VM servers.*

Software update procedure (LISTSERV-NJE only):

- `CLEANUP shipment` – Remove an installed shipment from the log
- `CLEANUP BEFORE dd mmm yy` – Remove all shipments installed before that date

- `PASSWORD shipment PW=instpw` – Confirm installation of a shipment, when requested by LISTSERV
- `RELOAD shipment` – Attempt to reload a shipment which failed due to a disk full condition
- `STATus` – Get a list of installed "shipments"

**OFFLINE**

*This command is only available on VM servers.*

Suspend processing of reader files and disable the `GET` command

**ONLINE**

*This command is only available on VM servers.*

Cancel `OFFLINE` condition

`PUTC fn ft <fm|cuu|dirid> <RECFM=F LRECL=nnn>`

*This command is only available on VM servers.*

Update a CMS file on one of LISTSERV's R/W mini-disks; note that this is similar to `SENDFILE + RECEIVE` or `LINK + COPYFILE` and should NOT be used to update file-server files

`SENDfile fn ft <fm|cuu|dirid>`

*This command is only available on VM servers.*

Request the server to send you a file from one of its disks

**SF**

*This command is only available on VM servers.*

Same as `SENDFILE`

`SHOW <function>`

*This command is only available on VM servers.*

In addition to the standard `SHOW` functions available on other servers, VM servers support the following functions:

- `BENCHmarks` – CPU/disk/paging benchmarks.
- `EXECLoad` – Statistics about `EXECLOADED` REXX files.
- `LSVFILER` – Statistics about `LSVFILER` file cache.
- `PREXX` – Statistics about `PREXX` functions usage.
- `STORage` – Information about available disk space and virtual storage.



**Note:** Some debugging commands and options have been omitted.

## 6.4 Sending commands to LISTSERV

You will see numerous references to "sending commands to LISTSERV" in this and other L-Soft manuals. All LISTSERV commands are sent to the server either by email or via the web administration interface described in Section 11 [Using the Web Administration Interface](#). For mailed commands, this means that you must create a new mail message

using whatever command this requires for your mail client (click on "New message" or its equivalent for most mail clients) addressed to the LISTSERV address. Let's say for the sake of argument that the list you are managing is running on a server called `LISTSERV.MYCORP.COM`. In order to send a command to that server, you would create a new message and address it to `LISTSERV@LISTSERV.MYCORP.COM`, and place the command(s) in the body (not the subject) of the message.

Depending on how you have security set up for your lists, some or all commands may require that you validate them with a personal LISTSERV password.

## 6.5 Defining Personal Passwords

The passwords recognized by LISTSERV for various operations (assuming that the `NO PW` parameter is not used with the "Validate=" keyword) are of two distinct types:

- **Personal Passwords** – LISTSERV can store a personal password in its signup files corresponding to your userid. This password not only can be used for list maintenance operations, but also protects your FUI (file update information) and AFD (automatic file distribution) subscriptions (if available on your server) and must be used to store your archive files, if any, on the server.
- **List Passwords** – List passwords are technically obsolete except in one particular case (peered lists, which require each peer to have the same password). We mention them here only because users upgrading from earlier versions will be aware of their existence. You should define and use a personal password for all protected operations.

To add a personal password, send mail to LISTSERV with the command

```
PW ADD newpassword
```

in the body of the message. LISTSERV will request a confirmation via the "OK" mechanism (see above) before it adds the password.

If you want to remove your password altogether, send the command

```
PW RESET
```

This command will also require confirmation.

And finally, if you simply want to change your personal password, send the command

```
PW CHANGE newpassword [PW=oldpassword]
```

If you do not include the old password in the command (e.g., you've forgotten it), LISTSERV will request an "OK" confirmation. Otherwise, it will act on the command without need for further confirmation (unless, of course, the *oldpassword* provided is incorrect).

Personal passwords may also be defined via the web administration interface at login time.



## Section 7 Creating and Maintaining Lists

You can create and maintain lists from any userid listed in the POSTMASTER keyword of LISTSERV's site configuration file. Note that a LISTSERV maintainer has the authority to GET and PUT any list, filelist, catalog, or archive file on the server (although for any list not set to "Send= Public", the LISTSERV maintainer must be subscribed to the list in order to post to it, and must additionally be a list Editor if the list is set to "Send= Editor...").

This section deals with the "manual" method of creating lists that has been available since LISTSERV first was written. Most users will probably prefer to use the web administration interface to create lists, for which, please see Section 11 [Using the Web Administration Interface](#).

### 7.1 Basic List Creation

At its simplest, creating a list is a matter of setting certain keywords to desired values in a file (called the "list header file") and storing the file in a place where LISTSERV can find it. The format of a typical list header file is relatively free-form, with only a few basic rules:

1. All header lines (including those inserted for "white space") must begin with the character "\*" (ASCII 0x2A).
2. Header lines can be up to 100 characters long (including the initial "\*" character). However in practice you will probably want to limit them to no more than 80.
3. All words ending with the character "=" (ASCII 0x3D) are evaluated as keywords.
4. The first non-"white space" line of the header file is evaluated as the descriptive name of the mailing list, and will be displayed as such by the LIST command.

Additionally, for PUT operations, you must add a line of the format

```
PUT listname.LIST PW=password
```

to the top of the file before mailing it. This PUT line does not begin with an asterisk. (Note that the filename for the list can be either in the format listname LIST or listname.list. The "." character is not necessary, but the word LIST is always necessary.)

Here is a sample of a basic list header with its PUT command at the top:

*Figure 7-1 Sample List Header*

```
PUT SAMPLE LIST PW=CCCCCCCC
*
* Title of sample LISTSERV list
*
* Review= Public      Subscription= Open      Send= Public
* Notify= Yes        Reply-to= List,Respect      Validate= No
* Notebook= Yes,A,Monthly,Public
*
* Owner= someone@somewhere.com
*
```

The preferred method of creating a new list is as follows:



- Using a text editor, prepare a "list header", for instance using the sample in figure above. You can also get the header of an existing (L-Soft) LISTSERV list and use it as a sample.
- The first line of the list header **MUST** be as follows:

```
PUT LISTNAME.LIST PW=CCCCCCCC
```

Replace "LISTNAME" with the name of your list, for example,

```
PUT MYLIST-L.LIST PW=CCCCCCCC
```

Then replace "CCCCCCCC" after "PW=" with the value of "CREATEPW=" in your site configuration file. If your CREATEPW is FIATLUX, then your complete PUT line for a list called MYLIST-L would be as follows:

```
PUT MYLIST-L.LIST PW=FIATLUX
```



**Note:** One of the most common errors made by new LISTSERV users is to leave out the ".LIST" part of the PUT command. If you leave this part out, LISTSERV will bounce the header back to you with the comment that it does not have any file by the name "MYLIST-L PW=FIATLUX".

- Following the PUT line, you insert as many "list header" lines as you need (see the sample). Each of these lines must begin with an asterisk in column 1, for example,

```
* Notebook= Yes,C:\LISTS\PUBLIC,Monthly,Public
```

If your mail software indents paragraphs by default, you must turn off paragraph indentation, or an attempt to store the list will be returned to you with a message that there did not appear to be any list header lines.

Each "list header" line contains information needed by LISTSERV to operate your list. Most of this information is provided by you in the form of values for standard keywords. You can use the sample header provided above as an example; a complete list of keywords recognized by LISTSERV along with descriptions of their functions can be found in the [List Keyword Reference](#) document.

- Mail the resulting file to the LISTSERV address.

The "LISTSERV address" is the address formed by "LISTSERV@" + the value you defined in the site configuration file for NODE=. For instance, if you defined NODE=XYZ.COM, the LISTSERV address would be LISTSERV@XYZ.COM.

This mail must be sent as Internet mail from a username defined as a "postmaster" in the LISTSERV configuration. For instance, from a VMS™ system, you would save your list file (say, in a file called 'newlist.create'), and then do:

```
$ mail
MAIL> send newlist.create
To: in%"listserv@xyz.com"
Subj:
MAIL>
```

Or, from a unix® system:

```
$ mail listserv@xyz.com < newlist.create
```

On a PC, you would use your POP client or other GUI-based mail program. Make sure to cut+paste the file via the Clipboard and not send it as an "attachment" or use drag and drop. "Attachment" mechanisms are often proprietary or PC-specific and cannot be guaranteed to work. Sending plain text pasted from the Clipboard always works.

The above is the preferred method for creating and editing list headers. LISTSERV will respond with a report that either the list has been successfully created or that various problems (fatal and/or non-fatal) have been detected. If only non-fatal problems are detected, the list will be stored anyway (non-fatal problems include no list password having been defined). Any fatal problem detected will abort the storage operation.



**Notes:** A less-desirable method of creating lists is to copy the list header file into LISTSERV's main directory and restart LISTSERV. LISTSERV will log a message to the effect that the list is not formatted properly and will then reformat the list. This assumes that the list header has been constructed properly and that there are no errors in the file that will cause LISTSERV to crash or to reject the list file. This method is useful only for creating lists; never attempt to edit a production list file in place and restart the server. The GET and PUT operations are the only supported methods for editing list files. Particularly under unix and Windows, LISTSERV will not always accept the edited list file because some editors will insert control characters or CR-LF combinations that LISTSERV cannot parse. Under VM or VMS, it is always possible that hand-editing the list will introduce some sequence that will cause an operational error. L-Soft suggests that this method be used sparingly, if at all, and does not support it. The first method is always preferable to the second.

BITNET users may also use the LSVPUT utility to store lists on BITNET-connected servers. However, LSVPUT is not documented here as the number of sites with BITNET connectivity is dropping rapidly and fewer and fewer users will be using LSVPUT.

## 7.2 Architecture-Specific Steps for List Creation

### 7.2.1 Unix: Creating Required Sendmail Aliases

*This section is only for use by Unix sites running Sendmail.*



**Note:** The file you need to edit in this step and the commands you need to issue will require root privileges. Also, while the procedure for manually modifying the sendmail aliases file is described below, you can also enter "make list name=listname" (where listname is the name of the list) to have the installation program complete this step automatically. The automated procedure assumes that your sendmail stores aliases in the file /etc/aliases, that the "newaliases" command will rebuild the aliases database, and finally that "kill -HUP `cat /etc/sendmail.pid`" will cause Sendmail to read in the updated alias list (in case running "newaliases" doesn't do that itself).

LISTSERV accepts and responds to several e-mail addresses. Even before you setup mailing lists, mail sent to listserv and owner-listserv should be handed to LISTSERV (see the installation guide for details). The link between LISTSERV and your mail system is

the `lsv_amin` program. If you are running Sendmail, the best way to route incoming mail to `lsv_amin` is by adding entries to your "aliases" file. Refer to the manual pages for sendmail on your system if you are not sure where the alias file is stored. On many systems the file will be called `/etc/aliases`. If you are not running sendmail, please see the LISTSERV installation guide for unix for further information.

Once you have constructed a list header file, and sent it to your Unix LISTSERV server, you need to instruct your mail system to route mail for that new list to the LISTSERV mail interface. That involves adding entries to your aliases file, much as you did when installing the server itself. For each new list, you'll need to add eight entries to the aliases file. The format of those lines is as follows,

```
NAME: "|/BBB/lsv_amin /SSS NAME"
owner-NAME: "|/BBB/lsv_amin /SSS owner-NAME"
NAME-request: "|/BBB/lsv_amin /SSS NAME-request"
NAME-search-request: "|/BBB/lsv_amin /SSS NAME-search-request"
NAME-server: "|/BBB/lsv_amin /SSS NAME-server"
NAME-signoff-request: "|/BBB/lsv_amin /SSS NAME-signoff-request"
NAME-subscribe-request: "|/BBB/lsv_amin /SSS NAME-subscribe-request"
NAME-unsubscribe-request: "|/BBB/lsv_amin /SSS NAME-unsubscribe-request"
```

where "NAME" is the name of the mailing list, "/BBB" in the directory where the mail interface was installed (BINDIR in the Makefile), and "/SSS" is the LISTSERV spool directory (LSVSPOOL in the Makefile). Note that "/SSS" can be either:

- An explicit directory definition, for example, `/var/spool/listserv`; or
- The switch `-t`, which is equivalent to the value in LSVSPOOL. (Note that the "make list" command makes aliases using `-t`.)



**Note:** If you use the precompiled copy of `lsv_amin` from the distribution kit rather than compiling your own from the source at install time, you cannot use the `-t` switch because the LSVSPOOL value is not compiled into the precompiled program. For this reason, the default installation uses the "paths" option, which provides the explicit directory definition.

For example, assuming the default values were chosen for BINDIR and LSVSPOOL, the aliases for a new list called "mylist" (using the default explicit path option) would be,

```
mylist: "|/usr/local/bin/lsv_amin /home/listserv/spool mylist"
owner-mylist: "|/usr/local/bin/lsv_amin /home/listserv/spool owner-mylist"
mylist-request: "|/usr/local/bin/lsv_amin /home/listserv/spool mylist-request"
mylist-search-request: "|/usr/local/bin/lsv_amin /home/listserv/spool
mylist-search-request"
mylist-server: "|/usr/local/bin/lsv_amin /home/listserv/spool mylist-server"
mylist-signoff-request: "|/usr/local/bin/lsv_amin /home/listserv/spool
mylist-signoff-request"
mylist-subscribe-request: "|/usr/local/bin/lsv_amin /home/listserv/spool
mylist-subscribe-request"
mylist-unsubscribe-request: "|/usr/local/bin/lsv_amin /home/listserv/spool
mylist-unsubscribe-request"
```



**Note:** The aliases may not wrap to the next line in `/etc/aliases`.

If you should decide to use the `-t` definition for the LSVSPOOL parameter, then the aliases would look like this instead:

```
mylist: "|/usr/local/bin/lsv_amin -t mylist"
```

and so forth.

Once you've added the new aliases to the file, you need to issue the "newaliases" command and (on some systems) send your Sendmail daemon a hangup (HUP) signal before they will take effect.

### 7.2.2 OpenVMS: Creating Required PMDF Aliases

*This section is for use only by OpenVMS sites running Innosoft International, Inc.'s PMDF® product, version 4.2 or later.*



**Note:** You will require system level privileges to edit the file in this step.

If PMDF is installed, in addition to the `listserv` and `owner-listserv` aliases which you've created in `PMDF_ROOT:[TABLE]ALIASES` at install time, you will need to add the following eight aliases for each new mailing list you create, where `listname` is the name of the list:

```
listname:                listname@LISTSERV
owner-listname:          owner-listname@LISTSERV
listname-request:        listname-request@LISTSERV
listname-search-request: listname-search-request@LISTSERV
listname-server:         listname-server@LISTSERV
listname-signoff-request: listname-signoff@LISTSERV
listname-subscriber-request: listname-subscribe-request@LISTSERV
listname-unsubscribe-request: listname-unsubscribe-request@LISTSERV
```



**Note:** You can get around this (and also solve a problem with address probing under VMS with PMDF as documented in Section 13.5.3 [OS-Specific Issues with Probing](#)) simply by creating a dedicated domain for LISTSERV (e.g., LISTSERV.XYZ.COM) and adding a rewrite rule to redirect all traffic for that host to the LSV channel. This also simplifies the creation of new lists since it is no longer necessary to make all of the PMDF aliases shown above every time you make a new list.

### 7.3 Sample Checklist for Creating Lists

1. Check to see that the list name is legal and not duplicated elsewhere (particularly if the list will be publicly and generally accessible). You can use the [CataList](#) as one resource for the latter.
2. If Notebook= Yes, then make the appropriate directory and make sure that LISTSERV has appropriate r/w permissions in it.
3. If Notebook= No but Digest= Yes, then make the appropriate directory and make sure that LISTSERV has appropriate r/w permissions in it.
4. Optionally, add the list to the quota file (ISP scope licenses only).
5. VM: Optionally, make a `listname.FILELIST` for this list (see Section 8 [File and Notebook Archives](#) for details). Non-VM: Optionally, make an entry in `SITE.CATALOG` for a sub-catalog belonging to this list (see Section 8 [File and Notebook](#)

[Archives](#) for details) and create a dummy `listname.CATALOG` in the specified directory.

6. Non-VM: Optionally, assuming that `Notebook= Yes` and you have installed the web archive interface as described in [Section 5 Configuring Your LISTSERV Site](#), create the `listname` directory under the base 'archives' directory. If you do this now, you won't have to GET/PUT the list header later to initialize things.
7. Create and store the list header with the list owner and you as the only subscribers.
8. Architecture-specific steps:
  - a. Unix, running Sendmail: Create the required Sendmail aliases for the list, either by hand or by using `'make list name=listname'`. Note that this is a required step for unix servers; if you don't make the Sendmail aliases, the list won't work. See [Section 7.2.1 Unix: Creating Required Sendmail Aliases](#) for details.
  - b. OpenVMS, running PMDF@: Create the required PMDF aliases for the list in `PMDF_ROOT: [TABLE]ALIASES`. Note that this is a required step for VMS servers running PMDF; if you don't make the aliases, the list won't work. See [Section 7.2.2 OpenVMS: Creating Required PMDF Aliases](#) for details (and an alternative work around).
9. Send a boilerplate "your list has been created" message to the list as the final test that the list works--if it doesn't, go back and find out why, then return here. See [Appendix B Sample Boilerplate Files](#) for a sample boilerplate message for this step, or use your own.
10. Delete yourself from the list (assuming you don't want to be subscribed).

At this point the list should be ready for use.

## 7.4 Naming Conventions

When naming a list, there are a few conventions and restrictions that you should keep in mind.

### 7.4.1 The "-L" Convention

The "-L" convention isn't required, but it can help people to realize that the mail is coming from a mailing list rather than from a real person. The people we are referring to here are people who run Internet mail systems, who may see a great deal of mail coming from a single host and begin to wonder why. If it comes from a userid that ends in a "-L", they will be more likely to recognize it as list mail.

### 7.4.2 Reserved Names

You may not create lists whose names match the following wildcards:

```
owner-*
*-request
*-search-request
*-server
*-signoff-request
```

```
*-subscribe-request
*-unsubscribe-request
```

For instance, lists cannot be made with names like "owner-loyalty", "linux-server", and "donation-request". While it is physically possible to create a list with a name that matches one of the above wildcards, attempts to send mail to the list (for example, a list called "linux-server") will result in an error, logged as follows in the LISTSERV log:

```
4 Dec 2001 11:47:02 -> Invalid list (LINUX), generating bounce.
```

These "pseudo-mailboxes" have a special meaning to LISTSERV, which has internal rules that govern how mail sent to these addresses is handled. See Section 17.3 [Communicating with List Owners](#) for more information on what happens to mail sent to these special addresses.

### 7.4.3 Reserved Characters

In general, you want to avoid "special" characters such as the ones above the number keys on your keyboard. For example, don't use:

- ! can be confused for "bang-path" addressing, for example, UUCP
- @ a reserved character
- # can cause problems with some mail software that uses it for addressing
- \$ may have a special meaning to the unix shell
- % another addressing character that could cause problems
- & sometimes reserved by non-unix systems (specifically on NT, it has a special meaning to the shell). However, please note that use of this character in the name of a list or in a sendmail alias for a list will cause LISTSERV on unix to choke. Note that it is possible under unix to create a list with a "&" character in the name quite easily, and it is also possible to create a sendmail alias with a "&" character in the alias. That does not mean it will work.
- \* the wildcard character
- ( ) generally reserved and can't be used in file names
- + should be avoided because recent versions of sendmail deliver mail addressed to "user+whatever@somedomain" to "user@somedomain." Whether or not this is an intelligent thing to do on sendmail's part is left as an exercise for the user, but it can affect mail being sent to a list with a "+" character in the listname.
- / reserved and can't be used in file names
- . Although on some systems it is physically possible to create lists with a dot character in the name, LISTSERV will not accept this nomenclature. The only place a dot can or should be used is before the word "LIST" in the PUT command; for example, PUT MYLIST-L.LIST is equivalent to PUT MYLIST-L LIST.
- " not allowed

It is best if you avoid the use of special characters altogether and stick exclusively to the letters A-Z, numbers 0-9, and the underscore and hyphen characters when naming lists.



Note that the "\_" (underscore) character may cause problems with some non-compliant receiving systems. Also note that the space character (ASCII 0x20) is illegal in a list name, and L-Soft recommends that, although apostrophes (aka "single-quotes", ASCII 0x27) are valid in an RFC822 username, they should not be used in list names since some mail programs may not accept them.

If you have any question about the validity of a particular name, you can of course refer to RFC822 or the updated RFC2822 for the Internet standards for e-mail addressing.

#### 7.4.4 Maximum Length of the List Name

The length of the list name (that is, the name of the list file and thus the "official" name of the list) is restricted as follows:

- VM: 8 characters
- Non-VM: unlimited (but see below)

If you need a longer list name for a list running on a VM server, then you should use the `List-ID=` keyword (see the [List Keyword Reference](#) document).



**Note:** L-Soft recommends using names of 32 characters or less whenever possible as they provide for correct alignment of the results returned by certain commands. Very long (for example, program-generated) list names are likely to conflict with mail system limits and L-Soft recommends other solutions to the problem of dynamically generated lists. As a rule, list names in excess of 70 characters are likely to result in mail delivery problems.

#### 7.4.5 Making the List Name User-Friendly

While you can (within limits) name a LISTSERV mailing list just about anything you want, you will probably want to follow a couple of simple guidelines:

1. Keep the name simple.
2. Keep the name as short as possible without causing confusion.

No doubt you could name a list MY-LIST-FOR-MATH-STUDIES, but who wants to type that? Conversely, MLFMS-L wouldn't mean much to Joe Random User. Somewhere in the middle is a reasonable compromise, for example, MATH-STUDIES (or even just MATH-S).

### 7.5 List Header Keywords

How a LISTSERV mailing list performs its tasks is defined by its header keywords. There are several different categories of keywords, each of which is discussed below in general terms. We will discuss these keywords in detail in subsequent sections, and a complete alphabetical listing of list header keywords, including default settings and all options available, is provided in the [List Keyword Reference](#) document.

- **Access Control Keywords** – These keywords designate the level of "openness" for a list. They determine who can post to the list, who can review the list of subscribers, and whether or not the list is open to general subscription.

- **Distribution Keywords** – This group has to do with how LISTSERV distributes postings to subscribers, including whether or not acknowledgments are sent back to posters, how many postings may go through the list daily, whether or not the list is available in digest form and whether it is available to USENET through a gateway. These keywords also determine whether or not list topics are enabled, and how LISTSERV will configure outgoing postings for replies.
- **Error Handling Keywords** – Included under this group are the keywords controlling automatic deletion, loop-checking, and to whom error messages are sent for disposition when received by LISTSERV.
- **List Maintenance and Moderation Keywords** – A fairly large group of keywords having to do with how the list is operated, including definitions for the list owner, list editor, and the list archive notebook; whether or not (and who) to notify when users subscribe and sign off; how often subscriptions must be renewed, and so forth. These are perhaps the most basic keywords that can be set for a given list, and one of them ("Owner=") must be set for a list to operate.
- **Security Keywords** – These keywords control who can "see" the list (that is, whether or not the list appears in the List of Lists for a given user, based on the user's host site), whether or not the list is protected by a password, and the level of security necessary for changes to the list itself. The "Exit=" keyword is also contained in this group.
- **Subscription Keywords** – These control whether or not the list is open to general subscriptions, whether or not a mailing path confirmation is required, and what user options are set by default upon subscription.
- **Other Keywords** – These control other aspects of list management that are not generally changed from their defaults, and which do not fit readily into the categories listed above.

## 7.6 Retrieving and Editing a List



**Warning:** Never attempt to hand-edit a production list file in place and restart the server. The GET and PUT operations are the only supported methods. Particularly under unix and Windows, LISTSERV will not always accept the hand-edited list file because some editors will insert control characters or CR-LF combinations that LISTSERV cannot parse. Under VM or VMS, it is always possible that hand-editing the list will introduce some sequence that will cause an operational error. L-Soft suggests that this method be used sparingly, if at all, and does not support it.

Once the list has been created, you can have a copy of the list sent to you for editing purposes. Simply issue the `GET listname` command to LISTSERV. This will cause the server to mail you a copy of the entire list (header and subscriber list).

If you want to change header keyword settings only, it is probably advisable to issue the GET command with the (HEADER switch):

```
GET listname (HEADER
```

The GET command automatically locks the list so that no changes can be made to the operating copy on the server until you do one of two things:



- Issue the `UNLOCK listname` command (if you decide no changes are needed)
- Send the list back to the server with the `PUT` command.

Leaving the list locked also prevents new subscribers from signing up. It is therefore not advisable to leave the list locked for long periods of time. This necessitates remembering to issue the `UNLOCK` command if you decide not to make any changes.

It is possible to request that `LISTSERV` not lock the list when it is sent to you. This is accomplished by adding the `(NOLOCK` switch to the `GET` command. You can use `(NOLOCK` and `(HEADER` together as in the following example:

```
GET listname (HEADER NOLOCK
```



**Note:** The "(" switch character is used only once.

**Caution:** It is not advisable to use the `(NOLOCK` switch in at least two cases:

Don't use the `(NOLOCK` switch if you are not the sole owner of the list. This prevents conflicting GETs and PUTs by different list owners. For instance, Owner(A) GETs the list without locking it. Owner(B) then also GETs the list. The owners make differing changes to the list header. Owner(B) PUTs his changes back first. Owner(A) then PUTs his changes back, erasing every change Owner(B) made. If Owner(A) had not used the `(NOLOCK` switch, Owner(B) would not have been able to GET a copy of the list until Owner(A) either unlocked the list or PUT his copy back. (Owner(B) could also unlock the list himself, but it would be advisable to ask Owner(A) if he was finished editing the list header before doing so.)

Don't use the `(NOLOCK` switch if you get the entire list rather than just the header. You will erase all subscriptions for users who subscribed between the time you GET the list and PUT the list back. It is easier to deal with questions as to why they got the "listname has been locked since time by list-owner" message than to explain why they got a subscription confirmation and now aren't getting list mail.



**Note:** A `PUT` command containing new subscribers added "on the fly" will result in only the header of the list being updated and a warning being generated that says if you really wanted to `PUT` the entire list, subscribers and all, that you should use the `PUTALL` command.

`LISTSERV` maintainers should note one further caution: It is considered extremely inadvisable to "hand-edit" subscriber lists, as columns at the far right of each subscriber's entry contain list control codes corresponding to the subscriber's personal option settings. The only case in which it might be appropriate to "hand-edit" would be to delete a user entirely, and then only if all attempts to delete the user via the `DELETE` command fail. For instance, X.400 or X.500 addresses can cause `DELETE` to fail because of their use of the "/" character. You can use wildcards to delete these subscriptions:

```
DELETE XYZ-L *ADMD=ABC*PRMD=DEF*@X400.SOMEHOST.COM
```

You can also enclose the address in double quotes:

```
DELETE XYZ-L "/ADMD=ABC/PRMD=DEF/...../@X400.SOMEHOST.COM"
```

## 7.7 Adding a List Password

This section is obsolete since 1.8c, but it is retained for compatibility with those sites still running 1.8b or earlier, or for sites running peered lists (all peers must have the same password).

When creating the list, the LISTSERV maintainer should assign a password for the list. However, note that in 1.8c and later, if the LISTSERV maintainer does not assign a password at the time of the list's creation, LISTSERV will generate a random password for the list. This random password can be changed later, but until and unless it is changed, administrators must provide their personal LISTSERV password (created with the "PW ADD *password*" command) when updating the list.



**Compatibility Note:** When upgrading to LISTSERV 1.8d and later from 1.8b or earlier, lists without passwords will not be altered during the upgrade. However, the first PUT operation for such lists after the upgrade will cause LISTSERV to add the random password to the list. List owners should be encouraged prior to the upgrade to create personal passwords for themselves with the "PW ADD *password*" command (if they have not done so already) and plan to use those passwords after the upgrade.

The list owner can change this password when storing the list (with the "PW=" keyword), but the first time the list owner stores the list, the original password or the list owner's personal password must be used. Note that not all LISTSERV maintainers assign list passwords by default; the new random password feature addresses that. However, for pre-1.8c servers it is highly recommended that one be assigned by adding a "PW=" header line as follows:

```
* PW=MYPASSWD
```

Replace "MYPASSWD" with the word chosen. Note that there should not be a space between "PW=" and the password. The list password is never changed unless specified explicitly in the list header when it is stored on the server. For additional security, the list password will not appear in the list header on subsequent GETs; to all intents and purposes it is invisible once it is assigned.

L-Soft's position on list passwords is that they have become obsolete with version 1.8c (they were actually obsolete as far back as 1987) except for the single exception of peered lists, and that personal passwords should be used instead to validate commands (such as the PUT command).

## 7.8 Storing a Modified List on the Host Machine

(If you are creating a list, see Section 7.1 [Basic List Creation](#). These instructions are for storing a list once it already exists on the server, for instance, if changes have been made to the list header after a GET operation.)

When you are ready to store your list on the host, include the list file in a mail message to LISTSERV. Ensure that the PW=XXXXXXXXX command is in the first line of the mail body. Then send the message.

If LISTSERV has trouble processing the edited list file, it will return a discrepancy report to you with each error noted. If the errors are categorized as "warnings only", LISTSERV will go ahead and store the list. However, if any one error is categorized as a serious error that could actually affect the correct operation of the list, the list will not be stored and the old version will be retained. (For instance, creating a list with no list password defined in

the header will generate a "soft" error under 1.8b and before, and the list will be stored. On the other hand, setting a list to "Send= Editor" and not defining an editor with "Editor=" is considered a "hard" error, and you will have to fix the error before LISTSERV will accept the list for storage.)



**Caution:** If you are using a mailer such as Eudora, Pegasus, Pine, or Microsoft Mail that allows "attachments" to mail, do not "attach" the list file to your mail message. It must be in plain text with the PUT line at the top. LISTSERV will not translate encoded attachments.

If your mail software inserts page formatting (margins) or quoting characters (such as ">") in forwarded mail, you need to either turn these features off or you must cut and paste the header into a new mail message. The PUT line MUST be on the first line of the message, and all header lines including the PUT MUST start in column 1. Specific problems have been noted with cc:Mail (where top and left margins get inserted) and with certain POP clients including Eudora and Microsoft Exchange (where forwarded mail is quoted with ">" by default).

Also, be sure to turn off your signature file (if you use one) before sending a PUT command to LISTSERV. If you don't, LISTSERV will attempt to parse the data in your signature file as RFC822 addresses to be added to the list, and you will receive either an error to the effect that the file includes invalid RFC822 addresses and it has therefore not been stored, or a warning that your PUT operation contains new subscriber information and only the list header has been stored (see Section 7.6 [Retrieving and Editing a List](#) for information on the PUTALL command).

## 7.9 Fixing Mistakes

LISTSERV always backs up the current list file before it stores a new copy. Should you discover that you have made a mistake (for instance, you have deleted all users by storing a header and adding users "on the fly"), it is possible to retrieve the previous copy of the list by issuing a `GET listname (OLD` command to the host server. You must then add the `PUTALL listname LIST PW=XXXXXXXX` command to the top of the file and store it.

It is also possible for the LISTSERV maintainer to restore the list by deleting or moving the `listname.LIST` file from LISTSERV's A directory and renaming the `listname.OLDLIST` file to `listname.LIST`. Naturally this method requires that the LISTSERV maintainer in question have appropriate access to LISTSERV's files and directories or be able to log in as the 'listserv' user.

## 7.10 Sample List Header File

A basic list header file for a list to be created might look like this (CREATEPW must be replaced with the appropriate password):

Figure 7-2 Sample List Header File for a List Called MYLIST

```

PUT MYLIST.LIST PW=CREATEPW
* The Descriptive Title of My List
*
* Owner= NATHAN@EXAMPLE.COM (Nathan Brindle)
* Notebook= Yes,E:\LISTS\MYLIST,Monthly,Public
* Errors-To= Owner
* Subscription= Open,Confirm
* Ack= Yes                Confidential= No
* Validate= No
* Reply-to= List,Respect  Review= Public
* Send= Public
* Default-Options= NoRepro,NoMIME
*
* This list installed on 96/06/02, running under L-Soft's LISTSERV-TCP/IP
* for Windows NT.
*
* Comment lines...
*

```

A list owner might take the created list and modify it as shown below. Note that the PUT command has been modified to include the password you've assigned with the PW ADD command.

Figure 7-3 The Edited List Header File Ready to be Sent Back to the Server

```

PUT MYLIST.LIST PW=MYPASSWD
* The Descriptive Title of My List
*
* Owner= NATHAN@EXAMPLE.COM (Nathan Brindle)
* Owner= Quiet:
* Owner= nathan@linus.dc.example.com
* Owner= ncbnet@linus.dc.example.com
* Notebook= Yes,E:\LISTS\MYLIST,Monthly,Public
* Auto-Delete= Yes,Full-Auto
* Errors-To= ncbnet@linus.dc.example.com
* Subscription= Open,Confirm
* Ack= Yes                Confidential= No                Notify= No
* Mail-Via= Distribute    Validate= No                Send= Public
* Reply-to= List,Respect  Review= Public                X-Tags= Yes
* Default-Options= NoRepro,NoMIME
*
* This list installed on 96/06/02, running under L-Soft's LISTSERV-TCP/IP
* for Windows NT.
*
* Comment lines...
*

```

## 7.11 Deleting a List

For security reasons, LISTSERV does not have an explicit command for deleting lists, although lists can be deleted through the web administration interface (see Section 11 [Using the Web Administration Interface](#)).

For those who prefer to delete lists the old, manual way, the LISTSERV administrator simply deletes the list file from the system command prompt with the appropriate file

system command (CMS ERASE for VM, DEL for VMS, ERASE for Windows, rm for Unix). A suggested procedure for deleting an established list (one with archives and so forth) follows:

1. Back up any files you wish to keep, such as notebook archives
2. For a digested list, you may want to send a QUIET SET *listname* NODIGEST FOR \*@\* command. This will cause LISTSERV to send out its accumulated digest to those who were set to DIGEST mode. If the list hasn't been active or if it's not "digested", you don't need to take this step.
3. Delete the *listname*.LIST file with the appropriate file system command.
4. If the list has web archives, then delete the */archives/listname.html* file and the */archives/listname/listname.ind\** files. You can also remove/delete the */archives/listname* directory at this time.
5. Although it is not absolutely necessary, stopping and restarting LISTSERV will complete the procedure. If you do not stop and restart LISTSERV, LISTSERV will fairly quickly notice that the list is gone, and will take care of this on its own.

## 7.12 Adding HTML to a List Header for the CataList

L-Soft's CataList service allows users to search the global list of LISTSERV lists via the World Wide Web. Adding an HTML description to a list is easy, and can do a lot to enhance the appearance of a list in the database. All the list owner or LISTSERV maintainer has to do is update the list header and add the text of your choice. Here is an example:

L-Soft's CataList service allows users to search the global list of LISTSERV lists via the World Wide Web. Adding an HTML description to a list is easy, and can do a lot to enhance the appearance of a list in the database. All the list owner or LISTSERV maintainer has to do is update the list header and add the text of your choice. Here is an example:

```
* The coffee lovers' list
*
* Review= Public      Subscription= Open          Send= Public
* Notify= Yes        Reply-to= List,Respect
* Notebook= Yes,L,Monthly,Public
*
* Owner= claudia@espresso.xyz.it (Claudia Serafino)
*
* <HTML>
* COFFEE-LOVERS is an open list for, well, coffee lovers! Our
* motto is: <cite>"Instant - just say no!"</cite>
* That's pretty much our whole charter, although there are a
* few other <a href="http://www.coffee.org/charter.html">
* rules</a> that you may want to read before joining. For
* instance, we don't allow flame wars about decaf: if you like it,
* well, it's your body after all.
*
* <p>The list is maintained by
* <a href="http://www.coffee.org/claudia.html">Claudia
* Serafino</a> (that's me!) and you will find all sorts of
* useful info about coffee on my home page.
* </HTML>
*
```

In other words, you just insert your HTML text in the list header and bracket it with `<HTML>` and `</HTML>` tags (these tags tell the web interface where the HTML text begins and ends – they are not actually sent to the web browser). There are three simple rules that you must follow when inserting your HTML data:

1. The `<HTML>` and `</HTML>` tags must appear on a separate line, as shown in the example above. You cannot have anything else on that line and, in particular, you cannot mix keyword definitions with HTML data.
2. The HTML data you are providing is embedded into the document shown by the web interface when users query your list. Because you are given some space between two horizontal rules on an existing page, rather than a whole new page, you should not include tags that affect the whole document, like for instance `<TITLE>`.
3. While this procedure is compatible with all versions of LISTSERV, there are a few restrictions on the placement of equal signs within your HTML text with versions that do not have any specific support for the `<HTML>` and `</HTML>` markers. In practice, you can ignore this rule unless you get an error message while storing your list.

When reformatting your list header description for HTML, bear in mind that the text will not always be viewed using a web browser. It is best to keep the formatting as clear as possible and minimize the usage of HTML tags, since there are still many people without WWW access. For instance, do not hesitate to use white space between paragraphs for clarity.

### 7.12.1 Update Latency

Barring network outages, a list header update takes a maximum of 24h to be reflected in the distributed LISTS database. Database updates are usually scheduled to be broadcast at night, so the changes take place overnight. Once the LISTS database has been updated, it can take a maximum of 24h for the frozen copy of the database used by the web interface to be updated. In most cases, both the LISTS database and its frozen copy on the web server will be updated overnight. However, if the site hosting your lists is several time zones west of the site hosting the web server, and if that server only updates itself once a day, you may have to wait two days for your update to be reflected.

### 7.12.2 Inserting a Pointer to Another List

Sometimes it may be useful to link a number of related lists together so that the viewer can quickly examine all the lists without having to go back to the search screen and retyping the names you are providing. You can do this using the special HTML sequence:

```
<!--#listref listname@hostname-->
```

This sequence is internally translated to an `<a>` tag with a URL that will bring up information about the list you indicated. You must then provide a suitable caption and a closing `</a>` tag. Example:

```
Don't forget to take a look at  
<!--#listref COFFEE-L@COFFEE.ORG-->  
the coffee list!</a>
```



### 7.12.3 Restrictions on the Placement of Equal Signs

While all versions of LISTSERV are supported, servers which have no specific support for the `<HTML>` and `</HTML>` tags will process your HTML data as an ordinary list header line and attempt to determine whether it contains a list header keyword or descriptive text. The exact algorithms vary from one version to another, but in general the parser looks for a single word followed by an equal sign. With HTML text, it is possible (if unlikely) to generate such patterns. Here is an example:

```
*
* Sample list with problem pattern
*
* <HTML>
* For more information on the list, just check <a
* href="http://www.xyz.edu/mypage.html">my home page.</a>
* </HTML>
*
```

In that case, you can just reorder the HTML data so that the equal sign does not appear in this position. Alternatively, if the equal sign was meant to be actually displayed as an equal sign (as opposed to being part of some HTML tag), you can use the HTML escape sequence `&#61;`; instead.

## 7.13 Setting Up Lists for Specific Purposes

You can create certain types of lists from standard templates via the web administration interface. See Section 11 [Using the Web Administration Interface](#) for information on how to access the web administration interface.

### 7.13.1 Public Discussion Lists

Public discussion lists have always been the "classic" type of LISTSERV mailing list. Such lists are available to discuss just about everything imaginable. In the last few years it has become desirable to secure mailing lists against random spamming and mail bombing, but no discussion of different types of lists would really be complete without talking about this kind of list.

Typically, a public discussion list is wide-open (although some things, like the ability to review the subscribership, may be restricted). Anyone can subscribe (with a confirmation to verify the mailing path), anyone can post, anyone can read the messages in the archives, and security is set fairly low. Very large lists (hundreds or even thousands of users with hundreds of postings every week) may likely be set up this way as it is a "low-maintenance" way to run a list (and most spams tend to be caught by LISTSERV's anti-spamming filters anyway). For instance, you might have

```
* My public discussion list (MYLIST-L)
* Subscription= Open,Confirm
* Ack= Yes
* Confidential= No
* Validate= No
* Reply-to= List,Respect
* Review= Owners          Send= Public          Errors-To= Owner
* Owner= joe@example.com
* Notebook= Yes,E:\LISTS\MYLIST-L,Weekly,Public
```

For more security, you might want to code

```
* Validate= Yes,Confirm
```

and if you want to cut down on the amount of "me-too"ism on the list, you could set

```
* Reply-to= Sender,Respect
```

to force the default Reply-To: header to point back to the original poster instead of to the list. Note that the ",Respect" option means that if a user sends mail to the list that contains a "Reply-To:" header pointing back to the list (unlikely that this may be), LISTSERV will "respect" that header and use it. If you absolutely do not want this to be possible, you should code the following instead:

```
* Reply-to= Sender,Ignore
```



**Caution:** "Reply-To:" are not universally honored!

**Note:** There is one major caveat with regard to the use of the Reply-To= list header keyword. Setting this parameter guarantees only one thing -- that LISTSERV will generate an appropriate RFC822 Reply-To: header in the mail it distributes to subscribers. THERE IS UNFORTUNATELY NO GUARANTEE THAT THE MAIL TRANSFER AGENT (MTA) OR MAIL CLIENT ON THE RECEIVING END WILL HONOR THE Reply-To: HEADER. This is because some mail clients, out-of-office robots, and Internet MTAs either simply do not recognize the existence of Reply-To: or do not implement it properly. Specifically RFC2076 "Common Internet Message Headers" reports that the use of Reply-To: is "controversial", that is, "The meaning and usage of this header is controversial, meaning that different implementors have chosen to implement the header in different ways. Because of this, such headers should be handled with caution and understanding of the different possible interpretations." (RFC2076, page 4). While L-Soft recognizes that it is sometimes important to provide an explicit Reply-To: header to indicate a response path, L-Soft cannot and will not be held responsible for problems arising from the inability of a remote server to properly process Reply-To: headers.

### 7.13.2 Private Discussion Lists

Private discussion lists are similar to public discussion lists, but with varying restrictions on who may subscribe, who may post and who may view the archives. Such lists are relatively safe from random spamming since typically only a subscriber can post (but note that a spammer spoofing mail from a subscriber's address will probably be successful unless first caught by the anti-spamming filters). For instance:

```
* My private discussion list (PRIVATE-L)
* Subscription= By_Owner
* Ack= Yes
* Confidential= Service
* Validate= No
* Reply-to= List,Respect
* Review= Owners
* Send= Private
* Errors-To= Owner
* Owner= joe@example.com
* Notebook= Yes,E:\LISTS\PRIVATE-L,Weekly,Public
```



is a low-security private discussion list where subscriptions requests are passed on to the list owner(s) for review, only subscribers may post, and only subscribers may view the list archives. Here again, for more security you might want to set "Validate= Yes, Confirm", and of course you can have replies go to the original poster rather than to the list with "Reply-To= Sender, Respect" (with the same caveats as noted above in Section 7.13.1 [Public Discussion Lists](#)).

### 7.13.3 Edited Lists

An edited list is one that requires a human editor to approve messages sent to the list. Some list software and most USENET newsgroups refer to this as "moderation", but to avoid confusion between two types of moderated LISTSERV lists, the present example will be referred to as an "edited" list.

Examples of edited lists range from refereed electronic journals to lists where the list owner simply wishes to exercise control over which postings are allowed to go to the list.

To set up a basic edited list, simply add

```
* Send= Editor
* Editor= someuser@somehost.com
```

to the basic list header. Note that the primary Editor= specification (that is, the first editor defined by an Editor= keyword for the list) must be a human person who will be able to act on postings sent to him or her for approval. You may not use an access-level specification (such as "Owner") when defining the primary editor for a list.

Please note that L-Soft recommends setting "Send= Editor, Confirm" so as to add a level of security against malicious users forging mail from an "Editor=" address to get around your moderation settings, or against badly-configured "vacation" programs that simply reflect the message back to the list in a manner that makes it appear that the mail is coming from the editor's address. The "Confirm" option causes LISTSERV to request an "OK" confirmation from an editor when it receives mail claiming to be from that editor.

You can define multiple editors, but only the first editor will receive postings for approval. Anyone defined as an editor may post directly to the list without further intervention. Multiple editors can be defined on separate Editor= lines or can be grouped several on a line, for example,

```
* Editor= someuser@somehost.com,anotheruser@anotherhost.com
* Editor= yetanotheruser@his.host.com
```

To approve postings with the above configuration, the editor simply forwards (or "resends", or "bounces"--the terminology is unclear between various mail programs) the posting back to the list address after making any desired changes to the content. This should be done with a mail program that supports "Resent-" fields. If "Resent-" fields are not found by LISTSERV in the headers of the approved posting, then the posting will appear as coming from the editor's address rather than from the original poster. If your mail program does not support "Resent-" fields, you should use the "Send= Editor, Hold" option and approve messages with the "OK" mechanism described below.

If you do not need to physically edit the content of your users' posts (for instance, to remove anything considered "off-topic" or to remove included mail headers and so forth), you can code

```
* Send= Editor, Hold
```

The "Hold" parameter causes LISTSERV to send you a copy of the posting along with a "command confirmation request". To approve the posting, you simply reply to the confirmation request with "ok".

For security purposes, you can code

```
* Send= Editor, Confirm
```

which will cause LISTSERV to request a command confirmation ("ok") from the editor sending the approved posting back to the list. This makes it impossible for an outside user to "spooF" mail from an Editor address.

Naturally, you can also code

```
* Send= Editor, Hold, Confirm
```

Finally, please note that the NOPOST subscriber option will take precedence over Editor=, if set for someone defined as an editor. This means that if you have "Default-Options= NOPOST" for your list and you add an editor as a subscriber, you will have to manually reset the editor to POST (with "SET listname POST FOR userid@host") before things will work properly. You will know that this is necessary if your editor can successfully approve postings but is then told that he or she cannot post to the list.

### 7.13.4 Moderated Lists



**Note:** The Moderator= keyword is disabled in LISTSERV Lite.

A moderated list is similar to an edited list, but for LISTSERV's purposes it refers to a list that uses the Moderator= list header keyword to "load-share" posting approvals among several editors. It is set up similarly to an edited list, as follows:

```
* Send= Editor, Confirm
* Editor= someuser@somehost.com
* Moderator= someuser@somehost.com, anotheruser@anotherhost.com
* Moderator= yetanotheruser@his.host.com
```

This list will "load-share" the approval process between the three moderators, who will each receive one-third of the postings for approval. Note that a primary editor should still be defined.

If it is desired to have one editor handle more than a single share of the approvals, you simply define the editor more than once in Moderator=. For instance,

```
* Send= Editor, Confirm
* Editor= someuser@somehost.com
* Moderator= someuser@somehost.com, anotheruser@anotherhost.com
* Moderator= someuser@somehost.com, yetanotheruser@his.host.com
```

would cause every other posting to be forwarded to someuser@somehost.com for approval.

If the parameter "All" is coded at the beginning of the list of moderators, LISTSERV will send copies of all postings to all moderators, any of whom may approve the message. An example of this would be

```
* Moderator= All, kent@net.police.net, joe@bar.edu
```

Please note that something like

```
* Moderator= kent@net.police.net,All,joe@bar.edu,alex@reges.com
```

is not valid. "All" must appear at the beginning of the list of moderators.

Assuming "Send= Editor, Hold", once a message is approved by one of the moderators, any other moderator attempting to approve the same message will be told that the message cannot be found and has probably expired (since the cookie for that message will be gone).

If the message body is edited in any way before it is approved (i.e., by forwarding an edited copy back to the list), and more than one moderator is involved, duplicates are possible. Thus it is important that the moderators of any list set up this way pay close attention to whether or not the posting has already been approved by another moderator. Note carefully that this means if the "All" parameter is used in "Moderator=" with "Send= Editor" (that is, without the "Hold" parameter), again a separate synchronization method will have to be used to prevent duplicates, as two moderators are unlikely to make exactly the same edits to the message. Even if LISTSERV were able to identify the two submissions as being the same message, it would not know which to choose over the other.

The "Hold" and "Confirm" options for "Send=" can also be used with these examples, if desired. L-Soft recommends that "Confirm" be used by default.



**Note:** The NOPOST subscriber option will take precedence over both Editor= and Moderator=, if set for someone so defined. This means that if you have "Default-Options= NOPOST" for your list and you add an editor or a moderator as a subscriber, you will have to manually reset the editor to POST (with "SET listname POST FOR userid@host") before things will work properly. You will know that this is necessary if your editor or moderator can successfully approve postings but is then told that he or she cannot post to the list.



**Note for moderation "OK" requests and MIME attachment display:** In versions previous up to LISTSERV 1.8e, an OK confirmation request for a message coming to a moderated list displayed the message to be approved in its "raw" format; that is, there was no attempt made to display/decode MIME attachments that might be present in the message to be approved. LISTSERV now addresses the problem by including a copy of the first text/plain part (if one exists in the message) for the purpose of quick screening. The following restrictions apply:

1.) This is only done for MIME messages (even simple single-part ones, but they must have MIME headers).

2.) The text part in question is sent pretty much 'as is', that is, as an extra text/plain part in the message, with all the options and encoding and what not supplied in the original message. The reason is quite simply that it would be a lot of work and, in some extreme cases (incompatible code page, etc.), completely impossible, to embed it into the first text/plain part with the LISTSERV message. The drawback is that some mail agents might conceivably only show the first part until you take some kind of clicking action.

It is important to understand that only the first text/plain part is extracted in this fashion. The goal was to make it easier to approve or reject simple text messages, not to build a factory around a simple problem. The ENTIRE message is available at an extra click.

Where security is a concern, it is important to review the ENTIRE original message and not just the plain text part. There could be an obscene GIF or another text part or a text/html part not matching the contents of the text/plain part or whatever. This is why, again, you are given the ENTIRE original message.

List owners using certain email clients (specifically Pine, which handles attachments in a secondary viewing area) may find the new format difficult to use. If preferred, the pre-1.8e behavior may be reverted to by specifying "NOMIME" in the Send= list header keyword; for instance,

```
* Send= Editor, Hold, NoMIME
```

### 7.13.5 Semi-Moderated Lists

"Semi-moderation" was developed some years ago after a great debate on whether or not an "urgent" message should be allowed to be posted to an edited list without having to go through the approval process. Although this option is still available, it can be misused by anyone who knows about it, and is therefore not generally recommended for use. However, should this feature be deemed necessary, it is activated by setting

```
* Send= Editor, Semi-Moderated
```

Then, any subscriber needing to send an "urgent" message to the list simply types "Urgent:" in the subject line of their mail, followed by the subject of the message. Messages that do not have the "Urgent:" subject are forwarded to the list editor for approval as usual.

In order to minimize the chance of spam slipping through without editorial approval, messages with an "Urgent:" subject originating from non-subscribers will be rejected.

### 7.13.6 Self-Moderated Lists

So-called "self-moderated" lists were invented in 1993 or 1994 when the current epidemic of spamming was beginning to get cranked up and before the "spam filter" was developed by L-Soft. With the spam filter in operation, self-moderation is not as much of an issue anymore, but some lists still run this way.

Self-moderation takes advantage of the ability to make an access-level a secondary list editor, and is implemented as follows:

```
* Send= Editor, Confirm
* Editor= someone@someplace.com, (listname)
```

(The "Hold" and "Confirm" parameters for "Send=" may naturally be used if required. L-Soft recommends that "Confirm" be used by default.)

Usually, one of the list owners is the primary editor (here "someone@someplace.com") and the specification of (*listname*) makes all of the subscribers of the *listname* list

editors, and thus eligible to send messages directly to the list without editor intervention. Postings from non-subscribers (e.g., spammers) are deflected to the primary owner for his or her disposition.

There is one caveat to this kind of list. If a user subscribes to the list, and later his mail address changes (for instance, the hostname changes slightly but mail sent to the old address is automatically forwarded to the new address), any postings from him to the list from the new address will be forwarded to the editor because the new address is not subscribed to the list. Thus there is a certain amount of list-owner overhead on this kind of list in keeping track of users whose addresses have changed and modifying the subscriber list to reflect those changes. The "CHANGE" command can be of help in this regard.

### 7.13.7 Private Edited/Moderated Lists

This type of edited or moderated list allows subscribed users to post with editor or moderator intervention, but rejects postings received from non-subscribers with a note to the poster stating that they are not allowed to post.

Using the same header you would create for an private discussion list (see Section 7.13.2 [Private Discussion Lists](#)), simply add the following line to the header:

```
* Default-Options= REVIEW
```

You should also add `Editor=` and (optionally) `Moderator=` keyword settings to the list. At least one editor must be defined to handle the message approval chores, otherwise the first listed list owner will receive the messages for approval.

The following rules apply:

- For brand-new lists or existing lists which have no subscribers, all subscribers added to the list after this option is set will be set to REVIEW, and nothing further needs to be done.
- For existing lists with existing subscribers, you will need to set the existing subscribers to the REVIEW option manually, that is, with the command

```
QUIET SET listname REVIEW FOR *@*
```
- New subscribers who sign up or are added after you add the `Default-Options=` keyword setting will automatically be set to the REVIEW option.
- Finally, the list editor will also be set to REVIEW if he is subscribed to the list under this scenario. This can be important if the list editor wants to approve even his own postings (for instance, to help avoid someone spoofing mail to the list from his address). If the list editor does not require this "suspenders and belt" level of security, he can simply set himself to NOREVIEW.

### 7.13.8 Auto-Responders

*Since LISTSERV Lite does not support list-level mail templates, this functionality is effectively not available in LISTSERV Lite.*

An "auto-responder" is a type of list that simply responds with a set message whenever it receives mail from someone. This kind of list can be useful for things like service

messages or upgrade availability, or even to simply send back a standardized message to a user who has sent mail to a "support" address.

A simple auto-responder header might look like this:

```
* Auto-responder for service messages
* Owner= someone@someplace.com
* Send= Public      Notebook= No      Subscription= Closed
```

In other words, it can be very simple, since you probably don't want notebook archives for this kind of auto-responder, you don't want people to subscribe to the list as it isn't really a mailing list, and so forth. To make the auto-response message for this list, you'd then create a `listname.MAILTPL` file (see Section 10 [Interpreting and Managing Log Files](#) for details) that includes a `POSTACK1` template, like the following:

```
>>> POSTACK1 Service Message for &MYNAMES
&MYNAMES will be down Sunday from 0200 EST until 0500 EST for backups
and upgrades. For more information contact LSTMAINT@&MYHOST.
```

This particular template would inform the user that `LISTSERV` would be down (`&MYNAMES` translates to `LISTSERV@NODE` where `NODE` is the value of `NODE=` in the system configuration file) and to send questions to `LSTMAINT@` the local host. In order to change the service message, it would be necessary only to change the `POSTACK1` template.

### 7.13.9 Announce-Only Lists

An "announce-only" list would be used to distribute a newsletter or other timely information where responses to the list are neither expected nor desired. A typical announce-only list header might look like this:

```
* The FOO Product Announcement List
* Owner= foo@myhost.com
* Owner= Quiet:
* Owner= anotheruser@myhost.com
* Owner= yetanotheruser@myhost.com
* Editor= foo@myhost.com
* Editor= anotheruser@myhost.com
* Editor= yetanotheruser@myhost.com
* Notebook= No
* Errors-To= Owner
* Subscription= Open,Confirm
* Validate= No
* Review= Owners
* Send= Editor,Confirm
* Reply-To= foo@myhost.com,Ignore
* Sender= None
```

This list is set up so that generally any response to postings will go back to `foo@myhost.com`, which might be a special account set up specifically to handle such things, or a mail alias pointing to another account. The newsletter can be posted by `foo`, or `anotheruser`, or `yetanotheruser`, all of whom are editors, but the likelihood is that it would be posted from the `foo` userid so that the `From:` line would read "From: `foo@myhost.com`".



L-Soft strongly recommends that all announce-only lists use the "Send=Editor,Confirm" or "Send=Editor,Hold,Confirm" setting. The ",Confirm" parameter tells LISTSERV to require a confirmation for any posting sent by a user defined as an Editor=. This is important for two reasons:

- **Security** – This setting tells LISTSERV to request confirmation from the Editor for all postings it receives that purport to be from that Editor. This prevents hackers from forging mail under an Editor's address, because any forgeries will require that the Editor in question approve them before they go to the list.
- **Loop Protection** – Certain broken mailers can and will bounce mail back to your list in a "reflected" manner, that is, the bounce will appear to be a legitimate posting from the Editor to the list instead of looking like an error. This is different from a forgery attempt because (it is assumed) the mailer on the other end is not doing this with malicious intent. Requiring the editor confirmation will stop these potential loop-generating messages from getting through to the list.

To stop a posting from going to the list under this scenario, simply don't OK it and delete the confirmation request message.

### 7.13.10 Restricted Subscription Lists with Automatically-Generated Questionnaire

*Since LISTSERV Lite does not support list-level mail templates, this functionality is effectively not available in LISTSERV Lite.*

Sometimes it is desired to send out a little questionnaire before approving a subscription to a list with a very narrowly-defined topic or to lists created for members of specific organizations. By setting "Subscription= By\_Owner", you can of course force all potential subscriptions to require list owner approval. In the "old days", if you wanted more information before you approved the subscription request, you had to manually send a questionnaire out to the user and wait for him or her to return it to you.

By setting "Subscription= By\_Owner" and adding two simple template forms to your listname.MAILTPL (as explained in Section 9 [Creating and Editing Mail and Web Templates](#)), you can now have LISTSERV send your questionnaire out automatically, as soon as the subscription request is received.

The first template form you need to add to `listname.MAILTPL` is called SUB\_OWNER, and in this case it would typically look like this:

```
>>> SUB_OWNER &LISTNAME: &WHOM requested to join
.TO &WHOM
A copy of the &LISTNAME membership questionnaire has been sent to you.
Please read it carefully and follow the instructions to complete it and
return it to the list owners.
```

The `.TO &WHOM` directive is required so that the message is sent to the subscriber rather than to the list owner. If you want the non-quiet list owners to receive a copy of this message (which is admittedly unlikely), you can simply add `CC: &OWNERS` to the end of the `.TO` line, for example,

```
.TO &WHOM CC: &OWNERS
```

Or, if you want to cc: a specific user such as `joe@unix1.example.com`, use

```
.TO &WHOM CC: joe@unix1.example.com
```



**Note:** You cannot format the `SUB_OWNER` template; it all comes out as one long paragraph without formatting no matter what you do, because it is a "linear" template. But you should modify it from the default to let people know that they will receive a questionnaire to be filled out and returned.

The second template form you need to add to `listname.MAILTPL` is called `ADDREQ1` and it can be as simple or as detailed as you want. All of the available template formatting commands can be used in `ADDREQ1`. For instance:

```
>>> ADDREQ1 &LISTNAME Membership Survey
.RE OWNERS
.TO &WHOM
.CE &LISTNAME Membership Survey
```

NOTE: Please make sure when you send this back that it goes to the address `&LISTNAME-Request@&MYHOST`. Thanks.

This is a standard questionnaire required for all prospective subscribers to `&LISTNAME`. Blah blah blah...

In this case, you want the message to go to the subscriber, with a Reply-To: header pointing back to the (non-quiet) list owners. The first line indicating the return address is added for those users with mail clients that don't recognize Reply-To: headers.

You can also put a pre-formatted `ADD` job into the questionnaire to simplify your job when the questionnaire comes back. For instance,

```
.fo off
```

-----  
For List Owner's Use Only -- Be sure to include with your Reply

```
-----
// JOB
ADD &LISTNAME &WHOM &USERNAME
// EOJ
-----
```

```
.fo on
```

For more detailed information on mail templates, see Section 9 [Creating and Editing Mail and Web Templates](#).

### 7.13.11 Peered Lists

*This functionality is not available in LISTSERV Lite.*

Occasionally the need to split a very large list may arise. This was more common when LISTSERV ran only on BITNET, whereas the TCP/IP version of LISTSERV is not limited by BITNET constraints. However, because of the fact that subscribers may be scattered all over the world, in rare cases it can make sense to split (or "peer") a list and share the mail load among two or more LISTSERV servers. Peering also makes it possible to have list archives located in more than one place; for example, a list might be peered between a European host and a North American host, making it possible for subscribers on each continent to retrieve archives from the nearer host.



Although there is no problem about peering to another L-Soft LISTSERV list, linking to a non-L-Soft mailing list manager is not supported and can and will cause serious problems (including mailing loops) for which L-Soft international, Inc. could not be held responsible.

### 7.13.11.1 Linking Two or More LISTSERV Mailing Lists

Please observe the following points:

- All lists should have a Peers= keyword setting that includes all of the other peers in the group as its arguments. For example, consider a peer group containing ListA, ListB, and ListC. ListA must have "Peers= ListB@its.host.com,ListC@its.host.com", whereas ListB must have "Peers= ListA@its.host.com,ListC@its.host.com" and finally ListC must have "Peers= ListA@its.host.com,ListB@its.host.com".

For lists running on LISTSERV for VM, setting the Peers= keyword makes it possible to EXPLODE them for better network efficiency. (Because peering is not widely used today, it is unlikely that the EXPLODE command will be ported to other platforms.)

- All lists must have the same list-level password, set with the PW= list header keyword. If this point is ignored, messages approved on one peer will not be accepted by the other peer and an error message will be generated, i.e.,

```
The approval request code received together with your posting for
the MYLIST-L list is incorrect. For a peered list, this may be a
normal condition. The approval protocol is not guaranteed to work
among peer chains with pre-1.8b servers, and will also fail if the
peers have a different password. For a non-peered list, the only
likely explanation is a failure in the mail system or a recent change
in mail system version or configuration. At any rate, please
resubmit your message and go through the approval procedure a second
time, and contact the LISTSERV administrator if the problem
persists.
```

```
----- Rejected message (73 lines) -----
```

This means that you must explicitly set the PW= list header keyword for each peer and not use the password LISTSERV generates automatically at list creation time. (This is the only case in which it is important and indeed required to manually set PW= for a list.)

- Each peer must be subscribed to at least one other peer, and the "real name" field for the subscription MUST be set to "Peer Distribution List".

### 7.13.11.2 Moving Users From One (Peer) Server to Another

You should be aware of the fact that a MOVE operation is not just an ADD to the new server and a DELete to the current one. This would effectively transfer the person from the old server to the new one but his distribution options would be lost in the process. Besides, you should make sure that the user does not lose any mail in the process. The proper course of action to be taken when people are moved from one list to the other is the following:

- Send mail to the list telling people that a new peer server is being linked to the list, and that some subscribers will be moved to it.

- If the prerequisites for using the `MOVE` command are met, you should use either individual `MOVE` commands (in the case that there are very few users to move) or a `batch-MOVE` command with associated `DDname` (see the LISTJOB MEMO guide for more information on commands-jobs) to move the users. You may want to use the `QUIET` option to suppress notification if there are a lot of users to move.



**Warning:** The `MOVE` command should not be used to move peer list servers. See the `MOVE` command description for more details.

If you cannot use the `MOVE` command, you should try one of the following two methods:

- For each user to be moved, issue the following commands in the following order:
  - Query `listname FOR userid@host` (old server), write down the options.
  - `QUIET ADD listname userid@host full_name`
  - `QUIET SET listname options FOR userid@host`
  - Wait until you get confirmation for the two previous commands
  - `QUIET DELeTe listname userid@host` (old server)
- If there are a lot of users to move, the following method is preferred:
  - `GET listname` (old server)
  - `GET listname` (new server)
  - If you are using VM XEDIT, then receive both files and use the XEDIT "PUT" and "GET" commands to move users from one list to the other. You must preserve the contents of columns 81-100 across the move.
  - If you are using another text editor, then make sure that the editor you are using does not "imbed" control codes such as line breaks, tabs or word-wrapping characters into the text when you edit it. Use the cut and paste controls to copy lines in their entirety. You must preserve the contents of columns 81-100 across the move. Imbedded control codes and/or word wrap will generate errors when the list is stored back on the server.
  - Store the two lists back on their respective servers.

### 7.13.11.3 Special Commands For Peered Lists Only

**ADDHere** `listname userid@host <full_name> <PW=list_password>`

The `ADDDHERE` command is strictly identical to `ADD`, with the exception that the placement of the user is not checked against the list of peer servers; in other words, the specified user is added to the local list without any further verification. (By comparison, the `ADD` command causes `LISTSERV` to check automatically to see if there is no better-suited peer list for the specified user.)

**EXPLODE** `listname <F=fformat> [VM only]`

The `EXPLODE` command provides a means whereby a list can be automatically analyzed by `LISTSERV` to optimize the placement of its recipients over the various peer servers hosting the list. It requires a "Peers=" keyword to be defined in the list header (see the [List Keyword Reference](#) document). Non-BITNET userids will be exploded according to the network address of the corresponding gateway (as per the `SERVICE`

NAMES file), or ignored if the gateway could not be identified. LISTSERV will create a commands-job file containing the necessary `MOVE` command to transfer all the users which were found to be (possibly) mis-allocated to the peer server which is nearest to them. This file will then be sent to you so that you can review it before sending it back to the server for execution.

```
MOVE listname userid@host <TO> newhost <PW=list_password>
  DD=ddname listid@newhost [VM only]
```

The `MOVE` command allows list owners to easily move users from one peer server to another. It will move the complete user entry from the source server to the destination one, including full name as it appears in the specified list and all list distribution options. The `MOVE` operation will be done in such a way that no mail can possibly be lost by the target while the `MOVE` operation is in progress (duplicate mail might be received for a short duration, however). Notification will be sent to the target user unless the `QUIET` option was used.

If the source and destination list names are identical, only the destination node ('newhost') needs be specified. Otherwise, the full network address ('listid@newhost') must be specified.

The `MOVE` command requires both source and destination lists to have the same password. Since each server will have to send a password to the other to validate the (special) `ADD/DELETE` commands it is sending to the other, it has potentially a way to trap the password specified by the server, thus thwarting any attempt at inventing a protocol to allow use of this command on lists which have a different password. Besides, no `MOVE` operation will be accepted on lists which do not have a password at all, because for technical reasons it would allow unauthorized users to easily add someone to a list (since there would be no password validation).

The `MOVE` command is the proper way to effect a move operation. You should not use any other command/set of commands unless you cannot use `MOVE`. THE `MOVE` COMMAND SHOULD NOT BE USED TO MOVE DISTRIBUTION LISTS!!! Since a `MOVE` is basically an `ADD + DELETE`, with the latter being done only `AFTER` the `ADD` is completed, moving a distribution list address with the `MOVE` command can cause a duplicate link to be defined for a short period of time. This could result in a transient mailing loop, which could become permanent if the size of the looping mailfiles is less than the size of the inter-servers "DELETE" command jobfile, and the RSCS priority of the latter has been altered.

### 7.13.12 Super-Lists and Sub-Lists

*This functionality is not available in LISTSERV Lite.*

It is possible to define a "super-list" (as in opposite of sub-list), that is, a "container" list that includes all the subscribers in a predefined set of sub-lists. This can be done recursively to any depth. Only the LISTSERV maintainer can create a super-list, for security reasons. Concretely, the "Sub-lists=" keyword is protected from owner tampering in the same fashion as "Notebook=". The value is a comma separated list of all the sub-lists, which must all be on the same (local) machine. For instance:

```
* Sub-lists= MYLIST-L,MYOTHERLIST-L
```

The default value for this keyword is null, that is, to have no sublists. In addition, the super-list and all of its sublists must reside on the same LISTSERV server.

The only difference between a normal list and a super-list is what happens when you post to it. With the super-list, the membership of all the sub-lists is added (recursively) and duplicates are suppressed. Other than that, the super-list is a normal list with its own archives, access control, etc. You can even subscribe to it, and this is actually an important aspect of the operation of super-lists. If you are subscribed to the super-list itself, the subscription options used to deliver super-messages to you are taken from your subscription to the super-list, just like with any other list. All combinations are allowed, and in particular NOMAIL is allowed, meaning you don't want to get messages posted to the super-list. When you are subscribed to multiple sub-lists, on the other hand, things work differently:

- NOMAIL subscriptions are ignored. You will get the super-message if you have an active (not NOMAIL) subscription to at least one sub-list. The idea is that the super-message must be equivalent to posting to all the sub-lists, without the duplicates. Since all it takes to get a message posted to all the sub-lists is a single non-NOMAIL subscription, this is how the super-list works. The only way not to get the super-messages is to subscribe to the super-list directly and set yourself to NOMAIL.
- The DIGEST and INDEX options are ignored and internally converted to MAIL. The first reason is that, since in most cases the user will be on multiple sub-lists (otherwise you don't need a super-list in the first place), the only safe method to set subscription options for super-messages is by subscribing to the super-list so that there is no ambiguity. The second reason is that, in most cases, super-lists will be used for out of band administrative messages rather than for large volume discussions, so it is actually preferable to have the message sent directly. The third reason is that the super-list and sub-lists may not necessarily offer the same options (DIGEST and INDEX). In particular it is expected that many super-lists will not have archives. If you want a DIGEST or INDEX for the super-messages, you must subscribe to the super-list directly.

Topics, if defined, are evaluated on a per-list basis. That is, for every sub-list (and for the super-list), LISTSERV determines whether the topic of the message is one that you want to see. If not, it acts as if you were not subscribed to this particular list. Roughly speaking, this works very well if all the sub-lists have the same set of topics (or a well-defined set of common topics), and doesn't work well at all if every list has its own set of topics.

Postings to a super-list are always archived in the super-list's notebooks (if enabled), and never in the notebooks of the sub-lists. This is because by its nature a posting to the super-list is not equivalent to cross-posting a message to all of the sub-lists. Rather, LISTSERV recurses into the sub-lists and generates an "on the fly" listing of all of the users on the super-list and the sub-lists (this is how it avoids duplicates, among other things) and then treats this "on the fly" listing as if it were the subscriber list of the super-list itself. You will note that a super-list posting is always identified as coming from the super-list, regardless of whether a given user is subscribed to the super-list or to one or more of the sub-lists.



**Note:** A `REVIEW` command sent for the super-list will not recurse into the sub-lists pointed to by the super-list. If you have a super-list called `SUPER` and you send a `REVIEW SUPER` command, LISTSERV will respond with only the people who are subscribed directly to `SUPER`. The only way to find out what users are covered by the super-list is to send `REVIEW` commands for the super-list and all of its sub-lists.

The REPRO subscriber option is recursive and will be honored for users who are subscribed to the sub-lists.

Access to the super-list's notebook archives is not automatically recursive. If you want sub-list subscribers to be able to access the archives of the super-list (but don't want the sub-list subscribers to have to subscribe to the super-list), then you must configure the Notebook= keyword for the super-list so that it contains references to each of the sublists. For example, say we have a super-list called SUPER and two sub-lists called SUB-A and SUB-B. We want the subscribers of both SUB-A and SUB-B to be able to read the archives of SUPER (since postings to SUPER won't be archived in SUB-A or SUB-B), but we don't want people who aren't subscribed to any of the three lists to be able to access the archives. So we set

```
* Notebook= Yes,C:\LISTS\SUPER,Monthly,Private,(SUB-A),(SUB-B)
```

and anyone subscribed to the SUPER list or to the SUB-A or SUB-B lists can access the SUPER archives.

If you have many sub-lists, you can specify multiple Notebook= lines, for example,

```
* Notebook= Yes,C:\LISTS\SUPER,Monthly,Private,(SUB-A),(SUB-B)
* Notebook= (SUB-C),(SUB-D),(SUB-E),(SUB-F)
```

LISTSERV will read these two (or more) Notebook= lines and concatenate the values.

### 7.13.13 Cloning Lists

Some sites may have a need for many lists that are essentially identical. For instance, a series of class section lists for a university department may have the same owner, allow the same class of users to subscribe, and so forth. LISERSV makes it possible to maintain large collections of lists by "including" keywords from an external file.

For instance, consider a mathematics course with ten sections. Each section should have its own list (for instance, called M101-001, M101-002, and so forth), but the lists will otherwise be identical. The LISERSV maintainer simply creates a text file (in this case called M101 KEYWORDS) containing the keyword definitions that will be shared by the lists, as follows:

```
PUT M101 KEYWORDS PW=createpw
* Owner= mathwhiz@someuni.edu (Professor J. Random User)
* Owner= Quiet:
* Owner= gradasst@someuni.edu (Joe Doakes, Graduate Assistant)
* Notebook= Yes,/home/listserv/archives/m101,Monthly,Private
* Auto-Delete= No
* Errors-To= gradasst@someuni.edu
* Subscription= Closed
* Notify= Yes          Confidential= Yes          Validate= Yes,Confirm,NoPW
* Reply-to= List,Ignore  Review= Owners          Send= Private
* Default-Options= Repro
```

Next, the LISERSV maintainer stores this file in the usual way, by first making a filelist or catalog entry for it (as outlined in Section 8 [File and Notebook Archives](#)) and then storing it with a PUT operation. Generally the GET and PUT FACs for this file should

specify that the list owner(s) should be able to retrieve and store it. The file must be stored in LISTSERV's A directory (the same directory that contains the \*.LIST files).



**Note:** It is also possible to create this file directly in LISTSERV's A directory with a text editor; if you do so, make sure that you do not include the `PUT` command shown above. You should still make the filelist or catalog entry for the file so that the list owners can retrieve and store it.

Next, the LISTSERV maintainer creates and stores a skeleton list header for each of the section lists. The first section list (M101-001) is illustrated below:

```
PUT M101-001 LIST PW=createpw
* Math 101 Section 001 Mailing List
* .IK M101
```

The `.IK` command tells LISTSERV that whenever it uses this list, it should read the keyword definitions from the file `M101 KEYWORDS` (note carefully that the syntax is `".IK M101"`, not `".IK M101 KEYWORDS"`). Now, whenever the professor in charge of the class wants to make a change to all of the M101 lists (for instance, he has a new graduate assistant), he simply `GETs` the file `M101 KEYWORDS`, makes the changes, and `PUTs` the file back, instead of having to `GET` separate headers for each list and make the changes to all of them individually.



**Notes:** On some servers it may be necessary to stop and restart LISTSERV (or do a `GET+PUT` of all of the list headers involved) to make changes to the `KEYWORDS` file appear. This is because LISTSERV may have the `KEYWORDS` file and/or the list headers that use it cached at the time you modify it.

In order to see the complete list header, send a `REVIEW listname` command. The response to a `GET` will be only the skeleton header with the `.IK` command. If `GET` did not work this way, you would not be able to change or remove the `.IK` command line once you set it.

The sample `KEYWORDS` file above includes a `Notebook= keyword`. This will cause the notelogs for all of the lists that use this `KEYWORDS` file to be written in the same directory, per the example, `/home/listserv/archives/m101`. This means that in that directory you would have notelogs for the M101-001 list, the M101-002 list, and so forth (depending of course on what lists use the example `M101 KEYWORDS` file). If this behavior is not desired, simply don't put a `Notebook= keyword` in the `KEYWORDS` file, and define it in the list header for the cloned list instead, either before or after the `.IK` directive.

For the web archive interface, note carefully that if you do use the same directory for all of the cloned lists' notelogs, you will still have to make separate web archive directories for each list under your `WWW_ARCHIVE_DIR` directory if you intend to serve the archives via the web interface. In other words, the web interface doesn't care where you keep a list's notelogs as long as it has a directory specified under `WWW_ARCHIVE_DIR` for it to write the list's web archive indexes into. So while all of your notelogs may go into `/home/listserv/archives/m101`, regardless of the



name of the cloned list, you still need to make (for example) `/usr/local/etc/httpd/htfiles/archives/m101-001` and so forth in order to serve the notelogs on the web.

## 7.14 Merging existing LISTSERV lists

It is sometimes desirable (or necessary) to merge two or more existing lists. There are a couple of different ways to do this.

### 7.14.1 Merging List A into List B; List A User Options Not Preserved

This is perhaps the simplest merge operation and requires only that you get the list of subscribers from list A and add them to list B, probably with a bulk operation as explained in Section 7.17 [Bulk Operations \(ADD and DELETE\)](#). User options are not preserved across the move and the users from list A will be subscribed to list B with whatever default options are set for list B.

### 7.14.2 Merging List A into List B; List A User Options Preserved

In this case you need to GET both lists A and B, header and all (so you do not use the `HEADER` switch in this case). LISTSERV will return copies of the entire list files to you including all of the subscribers along with an encoded option string for each subscriber. Usually this will look something like this:

```
* My test list
* (remaining header lines removed for clarity)
*
xxxxx@APK.NET Pxxxx Axxxxx
2AAARAA4HAAA
xxxxxxxxxxx@AOL.COM Rxxxxx Axxxx
2AAARAA2bAAA
xxxxxxx@LSOFT.COM Nxxxxxx Bxxxxxxx
2AAARAA2bAAA
xxxxxxxx@CS.ROSE-HULMAN.EDU Mxxx Dxxxxxx
2AAARAA3nAAA
```

Depending on how your mail client handles long lines, the subscriber lines will be either:

1. Kept as one single long line in which the option string starts in column 81 of the line.  
– or –
2. Split into two separate lines, the subscriber address and real name on one line followed by the option string on the next line.

If case 1, then you should have no problem with the following operations. If case 2, then you must either:

- Reformat the lines so that the option strings start in column 81 rather than being on separate lines; or
- If your mail program supports MIME, re-order the file as a MIME/TEXT attachment by adding `F=MIME/TEXT` to your `GET` command (e.g., `GET MYLIST-L F=MIME/TEXT`). This should preserve the long lines without inserting end-of-line characters.

In any case a correctly-formatted subscriber line looks like this:

```
nxxxxx@LSOFT.COM Nxxxxx Bxxxxxx                2AAARAA2bAAA
```

Next, assuming that the subscriber lines are correctly formatted, cut and paste list B into a new mail message addressed to LISTSERV. Make sure that your mail client has all formatting options turned off; for instance, make sure line wrap and any automatic "rich text" or HTML mail formatting is turned off. If you do not do this there is no guarantee that the list file will reach LISTSERV properly formatted.

At the bottom of this new message, you can cut and paste the subscribers from list A. Note that you don't want the header of list A, just the subscriber lines. Make sure that there is no blank line between the subscribers you pasted from list B and the subscribers you have just pasted from list A.

Finally, you can now `PUT` your new merged copy of list B.<sup>1</sup>

### 7.14.3 Merging List A and List B into List C

In this case (where you may be starting a completely new list and want to merge two old lists into it), follow the directions above depending on whether or not you want to preserve user options across the merge or not. The only difference is that you will be combining the subscribers from two lists into another list instead of combining subscribers from one list into a second list. In this case you do need to be careful not to add duplicate addresses, as LISTSERV will not catch them when you `PUT` the new list file. In fact it is probably more sensible to set appropriate defaults to the new list and store the header by itself, then add the users with a bulk operation (not preserving their old options) so that LISTSERV can catch any duplicates you might add.

## 7.15 Migrating Lists From One Site to Another

In migrating lists to LISTSERV, there are three typical possibilities:

1. You are migrating lists from an existing LISTSERV site (e.g., moving from VM to unix).
2. You are migrating lists from a non-LISTSERV site to LISTSERV.
3. You are creating a LISTSERV list from a Sendmail alias or other database of email addresses.

### 7.15.1 Migrating Lists From One LISTSERV Site to Another LISTSERV Site

Naturally, this is the simplest migration, but it still requires a few important steps. The preferred method (and the one that generally works the best) is to `GET` the list from the old server, make any changes necessary to the header (e.g., location of Notebook

---

1. There is an alternate method of doing this kind of merge which requires access to the machine LISTSERV is running on. You can simply use the 'listview' program to output the list files in text format, and redirect them into editable files (with 'listview -a listname > listname.txt', for instance). Then you can use a text editor to combine the two lists and either mail the resulting merged list to LISTSERV or store the merged file as listname.list and restart LISTSERV to reformat the file (as explained above in Section 7.1). However, note carefully that the latter method (save and restart LISTSERV) is not recommended, nor is it the supported method for storing lists, as it bypasses a number of checks LISTSERV does when you use `PUT`.



archives) and PUT the resulting list file on the new server. This method (assuming no corruption or reformatting of the list file by intervening mail systems) is preferred because it involves LISTSERV's internal syntax checking and other error-handling functions, LISTSERV knows exactly where to put the files, and the migration isn't restricted by possible architecture-specific problems.

The drawback to the preferred method is that you have to migrate one list at a time, which may not be acceptable if you need to migrate many lists in a short period of time. In general, you can simply FTP your list files from the old server to the new server, but note the following:

- You can migrate only from VM to non-VM, or from non-VM to non-VM. You cannot migrate using the FTP method from non-VM back to VM (unless you are prepared to reconstruct your list files and so forth from scratch). Naturally a GET and PUT will work if you need to move from non-VM to VM.
- If migrating from VM to non-VM, then be sure to FTP the list files, archives, and so forth in ASCII mode. If you use binary mode, the files will be unreadable on your new system.
- If migrating from non-VM to non-VM, you can FTP the list files in binary mode and any other files in ASCII mode. Please note that if moving the list files by binary FTP does not work, you will have to migrate the list using the preferred method outlined above. You could also create a new list header on the new server and add the users with an ADD IMPORT job as detailed in the next section.
- Once you have FTP'd the files to the new server, decide where you want things to go. The list file itself should go into LISTSERV's A directory (typically `~listserv/home` on unix systems, `LISTSERV\MAIN` on Windows systems, and `LISTSERV_ROOT:[MAIN]` on VMS systems). You may want to make a separate directory on your new server for archives, then subdirectories of that directory for each list (see Section 5.8 [Setting Up Archive and Notebook Directories for Use with LISTSERV](#)). If so, make the appropriate directories and move the archive files there. (This is particularly important if you intend to use the ISP options such as the file quota subsystem.) If you are copying non-notebook archives, you should read Section 8 [File and Notebook Archives](#) in order to set up a filelist or catalog file for these files.
- Next, you should restart LISTSERV. This is particularly important when moving lists from VM to other platforms, as LISTSERV will need to reformat the file into the binary format used on non-VM machines. If this is successful, you will see two messages in the console log:

```
6 May 1996 12:50:14 Invalid record format for list XXXXX-L.  
6 May 1996 12:50:14 -> List reformatted successfully.
```

If this is not successful, you will need to open the list file in a text editor and look for anything that might have caused a problem. Note that list header lines have a limit of 100 characters in length.

- Before releasing the list to the general public, be sure to GET the list header and make any changes that need to be made. Typically, changes will need to be made to

the location parameters of the `Notebook=` and/or `Digest=` keywords, particularly if you are moving from one platform to another.



**Note:** The first digest sent from the new site will say "First ever".

### 7.15.2 Migrating Lists from Non-LISTSERV Sites

Non-LISTSERV list files (notably from Majordomo and ListProc, but from other MLM software as well) are not directly compatible with LISTSERV. While it is probably possible to write a script or batch file for the purpose of converting one format to the other, it is outside the scope of this manual to describe this process.

Majordomo users will note that LISTSERV does not require two separate lists for those who want individual messages and those who want digested summaries. LISTSERV handles digesting internally for those who have set the personal option `DIGEST` for the list. Thus those sites migrating to LISTSERV from Majordomo will probably want to merge the digested and non-digested subscribers into one single list and let all subscribers know that they can set themselves to `DIGEST` mode with the `SET listname DIGEST` command. (It would also be possible to send commands to LISTSERV to set all of the old digest subscribers to `DIGEST` before releasing the list to the public.)

Under most conditions, the method recommended by L-Soft for migrating a non-LISTSERV list into LISTSERV format is the following:

- Create a list header for your list as noted above and store it on the LISTSERV server. If you plan to set "Validate=" to any value but "No", set it to "No" until the following steps are completed.
- Create a LISTSERV command language job as follows:

```
QUIET ADD listname DD=X IMPORT
//X DD *
internet-address1
internet-address2
/*
```

where "*listname*" is the name of the new list, and "*internet-address1*", "*internet-address2*" and other users are the internet addresses from the original list that you want to add to the new list. Optionally, you can add the user's "real name" field, for example,

```
QUIET ADD listname DD=X IMPORT
//X DD *
internet-address1 full_name
internet-address2 full_name
/*
```

You should remove any lines from the original list that do not actually identify subscriber addresses. If you are converting to LISTSERV from ListProc, note that LISTSERV will not convert ListProc user options to their LISTSERV equivalents; you must take a line like

```
user1@somehost.com POSTPONE NEWLIST NO user's name
```

and reduce it at least to

```
user1@somehost.com user's name
```

Otherwise, the ListProc options will become part of the `full_name` field.

- Send the command language job to LISTSERV.
- You will receive in return a confirmation that the job executed and whether or not it was successful:

```
> QUIET ADD XXXXX-L DD=X IMPORT
ADD: no error, 2 recipients added, no entry changed, none
forwarded.
```

List archive notebooks from non-LISTSERV sites can be copied into a file archive area for the list and registered in the `listname` FILELIST (VM) or `listname.CATALOG` (non-VM), but it is not recommended that non-LISTSERV notebooks be renamed with LISTSERV naming conventions, as this may cause problems with LISTSERV's database functions. For instance, if you have ListProc or Majordomo notebook archives that were kept monthly, L-Soft does not recommend renaming them with the `listname.loggyymm` format.

For information about how to convert non-LISTSERV archives to LISTSERV format, please see Section 8.10.3 [Migrating Old Notebook Archives \(Non-LISTSERV to LISTSERV\)](#).

**Alternate method of creating the list**<sup>1</sup>: You can send the list header and subscriber list to LISTSERV in the body of an e-mail (attachments will be ignored, the header and subscriber list MUST be plain text in the body of the mail message). Only one list can be created per e-mail, and the body of the mail must look like this:

```
PUT listname LIST PW=createpw
* Long title of list
* (more list header lines, must begin with asterisks in column 1)
userid1@example.com His Name
userid2@example.net Her Name
```

In the above syntax example, "listname" is the name of the list, and "createpw" is the CREATEPW value from your site configuration file. The text of all lines must begin in column 1. All header lines must begin with an asterisk. There must not be any blank lines anywhere in the text (they would be considered as end-of-file markers).

Subscribers added in this fashion will inherit any Default-Options and Default-Topics from the list header.

---

1. Another alternate (but unsupported) method of creating the list. Note that you can also create the list, header and subscriber list together, using a text editor. Simply start adding subscribers right after the last header line, save the file with the extension ".list", and restart LISTSERV as noted above in 7.15.1 to reformat the list. Do not, however, attempt to edit the list with a text editor once it has been created--use only LISTSERV commands sent via mail, the VM console, or the LCMD utility to make changes. And note carefully that subscribers added in this fashion (i.e., without a pre-existing user options string) DO NOT inherit any default options set in the header, and would require that any required options be set manually (that is, with the command QUIET SET listname options FOR \*@\*). You can avoid this problem by not using this method.

### 7.15.3 Migrating Lists from Sendmail Alias Files, Databases, Etc.

In general, you will follow the same procedure outlined in 7.15.2 to migrate from these types of lists. You may wish to write an executable script of some sort to pull the addresses and names (if you have them) from your database and surround them with the appropriate CJLI commands, particularly if your database is made from a web site and you need to run a periodic job to add users to your lists.

### 7.16 Changing the Name of an Existing List

Changing the name of an existing list on the same server as opposed to migrating a list from another server is somewhat different. Here is a checklist of the basic steps involved in renaming an existing list. For the purpose of this example we will assume that the list is named `MYLIST-L` and we want to rename it to `JOESLIST-L`.



**Note:** Operations that call for using OS-level commands are not performed by issuing commands to LISTSERV, but rather by opening a console session and typing the commands at your system's command prompt.

1. Stop LISTSERV.
2. Find the `mylist-1.list` file. LIST files are kept on LISTSERV's A disk (VM) or in its A directory (non-VM). The A directory for non-VM servers is normally located at `~listserv/home` for unix servers, `LISTSERV_ROOT:[MAIN]` for VMS servers, and `LISTSERV\MAIN` for Windows servers.
3. Copy (using your OS's command for copying files) `mylist-1.list` to `joeslist-1.list`. (Under unix you must name the file in lower case. Copying the list file will preserve all subscribers and all subscriber options.)
4. If necessary, create the directory for `joeslist-1`'s archives. If you had `mylist-1`'s archives in `~listserv/lists/mylist-1`; for instance, you should create the directory `~listserv/lists/joeslist-1`. Once this directory is created, you can copy the `mylist-1` archive notebooks over to it, and then rename any of the `mylist-1.*` files to `joeslist-1.*`. (You will want to copy the current notebook over again later, to make sure you get all of the postings up to the time of the switch.)



**Note:** It is not necessary (and probably not desirable in any case) to copy the `DBNAMES`, `DBINDEX`, `DBRINDEX`, or `-RAC` files as they will be rebuilt automatically by LISTSERV. Also, you don't need to copy the `DIGEST` or `SUBJECTS` files as we're going to take care of them later.

5. Again, if necessary, you should also copy over any files referenced by the list's catalog or filelist and make a new catalog or filelist for `joeslist-1`. You will also need to make an entry in `site.catalog` (non-VM) or `listserv.filelist` (VM) for the new `joeslist-l` catalog or filelist.
6. If the list was available through the web archive interface, make a `joeslist-1` directory for the web archive indexes (see Section 5 [Configuring Your LISTSERV Site](#) for details).
7. Restart LISTSERV.
8. Issue a `GET JOESLIST-L (HEADER NOLOCK` command to get the header. Make any changes you feel necessary; for instance, in the list's description or in the comments which may or may not contain the old list's name. You will also need to make

changes to any keyword that contains a directory reference; for instance, the `Notebook=` and `Digest=` keywords, so that they point to the right place. `PUT` the list header back on the server. (This `PUT` will cause `LISTSERV` to build web archive indexes for the list.)

9. Issue a `HOLD JOESLIST-L` command to keep the list from processing any postings from earlybird users.

At this point you are finished copying the old list to the new list. Now you need to do some housekeeping before notifying the users of the change.

10. Issue a `QUIET SET MYLIST-L NODIGEST NOINDEX FOR *@*` command to `LISTSERV`. This will force `LISTSERV` to send out the accumulated `MYLIST-L` digest and index issues to all users who had those options set.
11. Issue a `HOLD MYLIST-L` command to `LISTSERV`.
12. Copy the final `MYLIST-L` notebook archive file over to the `JOESLIST-L` directory so that you have all of the postings up to the time you issued the `HOLD`.
13. Get the header of the `MYLIST-L` list. You can now add a "New-List=" keyword to the header to let people know that the name of the list has been changed. This requires that you remove all other keywords from the header except "Owner=" and "Confidential=". You can set

```
* New-List= JOESLIST-L@LISTSERV.MYHOST.COM
* Confidential= Yes
```

in the list header so that a) the list no longer appears in the global List of Lists and in the CataList and b) so that all mail and inquiries sent to the old list address will be forwarded on to the new one. When you've made the changes to the header, `PUT` it back on the server.

14. Issue a `FREE JOESLIST-L` command to `LISTSERV`. (You should not need to issue a `FREE MYLIST-L` command.)

Congratulations, you've finished renaming the list. At this point you should probably announce the change and let people know where to find the archives, etc.

## 7.17 Bulk Operations (ADD and DELETE)

It is possible to use "bulk" operations to "front-load" or otherwise simplify the job of adding and/or deleting users from lists. This will typically be used on very large announce-type lists but the functionality is naturally available for all lists.

### 7.17.1 Bulk ADD Operations

To front-load or just to add a large number of users to an existing list, you construct a `LISTSERV JOB` framework as follows and then send it to `LISTSERV`. The `QUIET` and `IMPORT` command words are optional; omit the square brackets if you use them. The "full name" field is optional as long as you use the `IMPORT` option; otherwise you must either specify "\*" (for an anonymous subscription) or a full name consisting of at least two separate words.

```
[QUIET] ADD listname DD=ddname [IMPORT] PW=yourpassword
//ddname DD *
userid1@host1.com [full name]
```

```
userid2@host2.com [full name]
...
useridn@hostn.com [full name]
/*
```

The `IMPORT` option implies a `QUIET ADD` (in other words you do not need to specify `QUIET` if you use `IMPORT`) and otherwise vastly speeds up the `ADD` process by loosening syntax checking and omitting success messages. If you do not use the `IMPORT` option and do not specify `QUIET`, the users you bulk add will receive the normal `SIGNUP` message and/or `WELCOME` file as usual.

It is also possible to do bulk operations through the Web Administration Interface; see Section 11 [Using the Web Administration Interface](#) for details.

### 7.17.2 Bulk DELETE Operations

If you have a large number of users to delete at one time, you can use a bulk delete syntax that is similar to the bulk `ADD` documented above. However please note that there is no "`IMPORT`"-type option for this feature, and as usual for the `DELETE` command you specify only the user's address in the data `DD`.

There is, however, a `BRIEF` option that can be specified, which is useful when you don't want a long list of "userid@host has been deleted from list xxxx" messages, one for each user deleted. Use of the `BRIEF` option tells `LISTSERV` to return only a count of the users that were deleted.

Once again you construct a `LISTSERV JOB` framework as follows and then send it to `LISTSERV`:

```
[QUIET] DELeTe listname DD=ddname PW=yourpassword
//ddname DD *
userid1@host1.com
userid2@host2.com
...
useridn@hostn.com
/*
```

You will probably want to use the `QUIET` modifier when doing a bulk delete, in order to suppress the notification message to the users being deleted.

It is also possible to do bulk operations through the Web Administration Interface; see Section 11 [Using the Web Administration Interface](#) for details. However, note that very large bulk `ADD` and `DELETE` jobs should be sent via email in preference to using the Web Interface.

### 7.18 Content Filtering

*This feature is not available in `LISTSERV Lite`.*

This feature is intended primarily to filter out-of-office messages and the like. It is not intended as a profanity filter. Attempts to configure it to filter profanity will most likely prove to be futile in the long run and are not recommended by L-Soft.

The `CONTENT_FILTER` mail template form, if present, contains filtering rules, one rule per line, empty lines ignored. Each rule has the following format:

```
[prefix:] pattern
```



The prefix, if present, can be a mail header tag (e.g. "Subject:"); "Header:" to check the whole header; or "Text:" to search the message text. The latter is the default if no prefix is supplied, it is provided in case the pattern contains a colon in the first word. If there are multiple mail header tags with the specified name (e.g. "Received:"), each such tag is searched and it is enough for one of them to match the pattern. If the requested tag is not present in the header, there is (surprise!) no match. A text search will search every line of the first text/plain part in the message. If there is no text/plain part, there is no match. Again, this is designed to filter read receipts, loops, chain letters, spam, you name it. There was no attempt on the developers' part to make this a profanity filter, and future versions will not be "enhanced" to make futile attempts at (for instance) decoding Word documents to look for obscene words.

Regular comparisons such as those described above are not case sensitive. Patterns are standard LISTSERV patterns, that is, the asterisk is the wildcard character. If there is no asterisk in the pattern, it is replaced with "\*"pattern\*" much like the SCAN command.



**Documented Restriction:** You cannot match literal asterisk characters in a string as there is no way to escape them. Any asterisk in a pattern will always be evaluated as a wildcard.

The content filter also supports "exact match" comparisons, which are triggered by a double colon. For instance:

```
Subject::
```

There are two significant differences between exact and regular match:

- You must supply your own wildcard characters in an exact match (if you want to use wildcards, that is). A regular match will insert leading and trailing wildcards if none are found. Thus, an exact match is the only way to make a comparison without wildcards.
- You can make an exact match for the empty string. Empty regular matches are ignored since they map to a wildcard comparison for \*\*, which would be always true. This also makes it possible to apply an exact match to a message that does not contain a specified header. For instance, if you want all messages to contain a (mythical) KABOOM: RFC822 header, with an exact match you can tell LISTSERV to perform one of the content-filtering actions if the the header is not present. This is not possible with a regular match.



**Note:** You cannot differentiate a header with an empty KABOOM field from a header with no KABOOM field.

One of the most handy uses for the exact match syntax is to be able to write a rule to reject messages with blank subject lines. For instance:

```
Subject::
```

```
Action: REJECT Please resubmit your message with a non-blank subject.
```

Every rule can, optionally, be followed by an action rule. This has the following format:

```
Action: ALLOW
```

```
Action: REJECT reason
```

```
Action: DISCARD comment
```

```
Action: MODERATE
```



(The available actions are the same for both regular and exact comparisons.) For instance,

```
>>> CONTENT_FILTER
Subject: Out of office
Action: REJECT OOO messages are not allowed on this list.
Subject: Auto-Generated:
Action: REJECT
Text: Click here to be removed
Action: REJECT Buzz off, spammer.
Subject::
Action: REJECT Please resubmit with a non-blank subject.
Subject: copyright
Action: MODERATE
To: friend@public.com
Action: DISCARD This guy is a spammer
```

The default is "Action: REJECT" with no specified reason. REJECT means that the message is rejected. MODERATE means that the message is to be forwarded to the list editor to be manually approved or rejected. DISCARD means that the message is to be dropped on the floor without further processing; any text following DISCARD is echoed to the LISTSERV console (and is thus logged).

ALLOW means that the message is allowed and all remaining rules are ignored. This could be used in moderated lists to allow the list moderator to bypass certain filters, for instance:

```
>>> CONTENT_FILTER
Subject: Out of office
Action: REJECT OOO messages are not allowed on this list.
From: JOE@EXAMPLE.COM
Action: ALLOW
Text: Click here to be removed
Action: REJECT Buzz off, spammer.
```

In the example above, messages with Subject: lines containing "Out of office" are rejected. Messages containing the text "Click here to be removed" are also rejected UNLESS they come from joe@example.com.

The text of the rejection is fetched from the BAD\_CONTENT mail template form, with the reason supplied as a variable called &COMMENT. The rejection message looks like this:

```
Date: Tue, 4 Dec 2001 22:03:42 -0500
From: "L-Soft list server at LISTSERV.EXAMPLE.COM (1.8e)"
<LISTSERV@LISTSERV.EXAMPLE.COM>
Subject: Rejected posting to TEST@LISTSERV.EXAMPLE.COM
To: Joe User <joe@EXAMPLE.COM>
```

```
Your posting to the TEST list has been rejected by the content filter.
OOO messages are not allowed on this list.
```

followed by the text of the posting including all mail headers. (In this case the body of the message contained the text "out of office" and the rule above was applied.)

A default site-wide CONTENT\_FILTER template form may be defined in `$$SITE$.MAILTPL` for use by lists whose owners do not prefer to provide their own custom versions in their `listname.MAILTPL` files.

## 7.19 DomainKeys Message Signing

*This feature is not available in LISTSERV Lite.*

DomainKeys message signing is available to sites running LISTSERV Classic or LISTSERV Classic HPO. Current LISTSERV maintenance is also required. For more information on how to configure LISTSERV for DomainKeys support, please refer to Section 5.12 [LISTSERV DomainKeys Support](#).

Assuming that it is available for your use, DomainKeys support for lists is enabled by default. This means that all list postings and administrative messages related to a list will be signed to assert that they actually originated from your LISTSERV server.

If for some reason you wish to disable DomainKeys message signing for a given list, you can do so by adding

```
* Misc-Options: NO_DKIM_SIGNATURE
```

to your list header. Or, if you prefer to disable it server-wide by default, you can add NO\_DKIM\_SIGNATURE to the DEFAULT\_MISC\_OPTIONS site configuration variable setting.

Incoming DomainKeys or DKIM signatures submitted to a mailing list will be removed unless "Misc-Options= KEEP\_DKIM\_SIGNATURE" is set in the list configuration. This is necessary because these signatures almost never match after the message has been processed. The worst thing that could possibly happen to your deliverability is a DomainKeys signature that does not match and causes the message to be flagged as suspicious.

The KEEP\_DKIM\_SIGNATURE option is experimental and not meant for general use. As DomainKeys is specified today, signatures DO NOT survive posting to mailing lists (LISTSERV or otherwise), so LISTSERV removes them by default to avoid triggering alerts for subscribers on systems that have implemented the client side of DomainKeys. The DKIM specification may be more robust in this respect, but even DKIM signatures will probably not survive when posted through a mailing list. Use the KEEP\_DKIM\_SIGNATURE option at your own risk.

## Section 8 File and Notebook Archives



**Documented Restriction:** The hierarchical listname.catalog system documented in 8.4, below, is not available under LISTSERV Lite. You may store files on a Lite server for people to retrieve, but the files must be registered in the site.catalog file and must reside in the same directory with the \*.list files so that LISTSERV can find them.

There are three file server systems currently in use by various versions of LISTSERV:

- The VM (mainframe) version of LISTSERV continues to support the "traditional" file server system. While it is very powerful, this file server system dates back to 1986 and suffers from a few annoying limitations. In addition, it is written in a non portable language. This will be replaced eventually with the "new" file server system, currently under development.
- The non-VM versions of LISTSERV 1.8d enhanced further the new file server system introduced in non-VM 1.8c, which included most of the functionality of the "traditional" file system. Notably, GIVE and file "packages" became available. Most end user commands continue to work as before. However, there is no guarantee that the internal data files manipulated by the file server functions will remain as before. Note that SITE.CATALOG files from versions 1.8a through 1.8c are still supported and will not need to be changed in order to work with 1.8d and later.
- The non-VM versions of LISTSERV 1.8a and 1.8b supported a "temporary" file server system, to provide an interim solution while the new system was being developed. This temporary system supports only a subset of the functions of the traditional system. This system is no longer supported by L-Soft as it has been superseded by the new non-VM file server referenced above.

In general, the three systems are compatible, with the understanding that the temporary system does not include all the possible options. However, the mechanism for registering files (defining them to the file server system) is different.

Since the first and third systems will eventually be replaced by the second system, rather than providing an exhaustive section detailing all filelist aspects from the management side, we have provided only a basic overview of the two systems currently in the field with 15.0, with pointers to where further information may be obtained.

### 8.1 The File Archive

The file archive consists of all files other than notebook logs that have been stored on the LISTSERV host for your list.

Users can find out what files are available for a specific list by sending the command INDEX listname to the appropriate LISTSERV host.

### 8.2 Starting a file archive for your list

#### On VM Systems ONLY

With the traditional system (running on the VM servers), the LISTSERV maintainer creates files called "xxxx FILELIST", which contain definitions for all the files belonging to a particular archive.

These FILELIST files must be created by the LISTSERV maintainer at the site before they can be edited by the list owner.<sup>1</sup>

### On Workstation and PC Systems

The LISTSERV maintainer stores "root-level" file definitions in a file called SITE.CATALOG, which should be placed in the same directory with the SYSTEM.CATALOG file.<sup>2</sup> The LISTSERV maintainer can also define "sub-catalogs" which in turn can define further files. You should be aware of the differences between VM and workstation file server functions as many people are using and will continue to use the VM file server with different conventions, and may give you incorrect advice. Non-VM sites should skip section 8.3, and use the information below in section 8.4 to maintain their file archives.

## 8.3 Filelist Maintenance (VM Systems Only)

*If you are running LISTSERV under unix, Windows, or VMS, please skip this section as it does not pertain in any way to your implementation of LISTSERV.*

Maintaining the filelist for your archive is not difficult. It requires only that you have a working knowledge of VM XEDIT (or your local system's editor) and understand how to send files via e-mail.

### 8.3.1 Creating a Filelist (VM Systems Only)

Please see FSV GUIDE (available at <ftp://ftp.lsoft.com/documents/fsv.guide>) for details.

### 8.3.2 Adding FAC Codes (VM Systems Only)

Please see FSV GUIDE (available at <ftp://ftp.lsoft.com/documents/fsv.guide>) for details.

### 8.3.3 Retrieving the Filelist (VM Systems Only)

To retrieve your filelist in an editable format, send the command

```
GET listname FILELIST PW=XXXXXXXXX (CTL
```

to the LISTSERV host where the filelist is stored. The (CTL switch causes LISTSERV to lock the filelist until you store it again or explicitly unlock it with an UNLOCK *listname* FILELIST command. (If you don't want to lock the filelist, use (CTL NOLOCK instead.) If your mail account is not located on the same host as LISTSERV, you will need to provide your personal password (same as your password for getting and putting your lists).

---

1. If you are interested in the mechanics of starting a VM-type filelist, the best reference is "Setting Up the LISTSERV File Server--A Beginner's Guide" by Ben Chi ([bec@albany.edu](mailto:bec@albany.edu)). This publication is available from LISTSERV@LISTSERV.NET as FSV GUIDE, or at <ftp://ftp.lsoft.com/documents/fsv.guide>.

2. Under unix, all files accessed by LISTSERV must be named in lower case (i.e., 'ls' must show site.catalog, not SITE.CATALOG or Site.Catalog). Internally LISTSERV does not care about case since it translates everything to lower case for the purpose of accessing the unix file system, and you may use upper or mixed case within the catalog file itself.

A filelist retrieved with the (CTL option does not look like the filelist you get with an INDEX command. A sample (CTL option filelist appears below.

Figure 8-1 Sample Filelist Retrieved with (CTL Option

```
* Files associated with MYLIST and available to subscribers:
*
*          rec          last - change
* filename filetype  GET PUT -fm lrecl nrecs  date    time  Remarks
* -----
MYLIST  POLICY      ALL OWN V   79   45 94/03/16 12:04:23  Mission Statement
MYLIST  BOOKLIST     ALL OWN V   79  177 94/04/19 16:24:57  Books of interest
MYLIST  QUARTER     ALL OWN V   73  113 95/03/11 08:57:04  Quarterly posting

* Listowner's files (not public)
MYLIST  FAREWELL     OWN OWN V    78    9 95/03/11 08:53:41  Goodbye memo
MYLIST  WELCOME      OWN OWN V    73  105 95/03/11 09:14:38  Hello memo
```



**Notes:** The filelist does not include the comment lines you would normally see at the top of an INDEX filelist; nor does it include any notebook archives. LISTSERV creates these lines dynamically at the time the INDEX command is received from a user. If the filelist you have retrieved has any of this kind of material in it, either a) you have not retrieved the filelist correctly, or b) you or someone else has stored the filelist previously with this material included. If you did a GET with (CTL, you should be able to remove these extraneous lines by simply deleting them.

If you do an INDEX of your archive and it has (for instance) two sets of comment lines or duplicate notebook archive listings, then you should GET the filelist with (CTL and edit out the offending lines. While the extra lines will not affect the operation of the file server, they are a source of potential confusion for your users.

### 8.3.4 Adding File Descriptors to the Filelist (VM Systems Only)

Adding a file to a filelist is not exactly accurate terminology, although it is a widely-used phrase. Adding files to file archives is a two-step process: First, add a file descriptor to the appropriate filelist and store the filelist on the server. Second, store the file itself on the server.

To add a file descriptor, start a line with a space and then type in your file's name, access codes, five dots (periods), and then a short description, each separated by a space. For example:

```
MYLIST FAQ ALL OWN . . . . . Frequently-Asked Questions for MYLIST
```



**Note:** The line must begin with a space. Also, you must place five dots separated by spaces between the PUT file access code (here it is OWN) and the short description. These dots are place holders for the record format (recfm), logical record length (lrecl), number of records (nrecs), and the date and time of the last update. If these dots are not present, LISTSERV will return an error message when you try to store the filelist.

The line you have just added does not look like the other lines in the filelist. Ignore the "pretty" formatting. LISTSERV will reformat the information for you. After adding the line, your filelist should look like this:

Figure 8-2 Adding a File Descriptor to the Filelist

```

Files associated with MYLIST and available to subscribers:
*
*          rec          last - change
* filename filetype  GET PUT -fm lrecl nrecs  date      time      Remarks
* -----
MYLIST  POLICY      ALL OWN V   79   45 94/03/16 12:04:23 Mission Statement
MYLIST  BOOKLIST     ALL OWN V   79  177 94/04/19 16:24:57 Books of interest
MYLIST  QUARTER     ALL OWN V   73  113 95/03/11 08:57:04 Quarterly posting
MYLIST  FAQ ALL OWN . . . . . Frequently-Asked Questions for MYLIST

* Listowner's files (not public)
MYLIST  FAREWELL     OWN OWN V   78    9 95/03/11 08:53:41 Goodbye memo
MYLIST  WELCOME      OWN OWN V   73   105 95/03/11 09:14:38 Hello memo

```

You can add comment lines to the filelist by placing an asterisk in the left-most column instead of a space. Comment lines can act as indexes, descriptions, or pointers to other resources.

Once you are finished adding file descriptors, save the filelist to disk.

### 8.3.5 File Access Codes (FAC) for User Access (VM Systems Only)

FACs define which users have access to files in the file archive. The FAC for GET indicates who may retrieve the files, and the FAC for PUT indicates who may store the files on the server. (Note that some special FACs exist for "superusers" such as the LISTSERV maintainer(s) and the LISTSERV Master Coordinator, who may GET and PUT any file regardless of its GET/PUT permissions.)

The basic FAC codes that are always available for VM servers are:

- ALL – Universal access.
- PRV – Only members of the associated mailing list have access.
- OWN – Only the owners of the associated mailing list have access.

(The FAC codes PRV and OWN work only on the VM filelist system. They do not work on the non-VM catalog system. See Section 8.4 [The listname.CATALOG System on Non-VM Systems](#) if you are configuring the non-VM systems.)



**Note:** This assumes the name of the filelist is identical to the name of the associated mailing list – for instance, MYLIST@FOO.BAR.EDU would have a MYLIST LIST file and a MYLIST FILELIST file. Ask your LISTSERV maintainer for assistance if this is not the case or if you need special FACs added for special user access to files.)

### 8.3.6 Deleting File Descriptors from the Filelist (VM Systems Only)



**Warning:** Before you delete file descriptors from the filelist, you should delete the files themselves from LISTSERV's archive disk. See Section 8.6 [Deleting Files from the Host Machine](#) for instructions. If this step is not followed, LISTSERV may not be able to find the file you want to delete after you edit the filelist and store it.

### 8.3.7 Storing the Filelist (VM Systems Only)

1. Create a mail message to LISTSERV at the appropriate host. (Sending a filelist to LISTSERV@LISTSERV.NET will not work. The filelist must be sent to the host it resides on.)
2. Include the filelist file as plain text in the body of the mail message. Do not attach it with MIME or another encoding scheme, as LISTSERV does not translate encoded messages.
3. Make sure that your mail client does not automatically add a signature file to the bottom of your mail. If it does, your signature file will be treated as part of the filelist and will be stored along with it.
4. At the top of the filelist, add a single line as follows:

```
PUT filename FILELIST PW=XXXXXXXX
```

where XXXXXXXX is your personal password for LISTSERV on that host. Note that this is similar to the PUT command used when storing the list file.

5. Send the filelist to LISTSERV.

Once LISTSERV acknowledges the receipt and storage of the filelist, you can send the files that correspond to the file descriptors in your filelist. See Section 8.5 [Storing Files on the Host Machine](#) for instructions.

## 8.4 The listname.CATALOG System on Non-VM Systems



**Documented Restriction:** The hierarchical listname.catalog system documented below is not available under LISTSERV Lite. You may store files on a Lite server for people to retrieve, but the files must be registered in the site.catalog file and must reside in the same directory with the \*.list files so that LISTSERV can find them.

LISTSERV version 1.8c and later uses a file archive registration system similar to (but differing in important respects from) the old VM FILELIST system. This system is available on the VMS, unix, and Windows ports only. VM sites will continue to use the old FILELIST system indefinitely as it still offers more functionality than the new system.

Files to be made generally available to users (e.g., not specific to any one list on your server) should still be registered in the site.catalog file as before.

Prior to 1.8c, entries in site.catalog were written like this:

```
MY.FILE          my.file./home/lists/xyz          ALL JOE@XYZ.COM
```

In 1.8c a new "native" format for these entries was introduced, and the new format is used in all of the examples below. The old format remains supported for compatibility.

However, note that you **MUST** use the old format if any of the directories in the path contains a period.





**Documented Restriction:** All files manipulated by LISTSERV must be accessible through LISTSERV's OS-independent file access methods. This means that files whose name contains spaces or control characters (or, under unix, upper case characters) cannot be accessed. Similarly, files whose name does not contain a period cannot be manipulated by LISTSERV. There is no limit on the length of the file name, only on its contents. Note that these "system filenames" are not visible to the end users, who refer to the files by the names assigned in the catalog.

### 8.4.1 Adding Files to the SITE.CATALOG

This is the most basic way to add files to LISTSERV's file archive system so they can be made available to users via the `GET` command.

To register a new file to the server on workstation systems, the LISTSERV maintainer adds a line to the `SITE.CATALOG` file. If `SITE.CATALOG` does not already exist (it is not shipped with the installation kits), simply open a new ASCII text file named `site.catalog` in the same directory as `system.catalog` and add entries to it as shown below. (Do not just add entries to `system.catalog` as this file will always be overwritten during a software update.)

Here is what a typical `SITE.CATALOG` entry looks like under Windows NT:

```
MY.FILE      C:\FILES\XYZ\MY.FILE  XXX YYY
```

And the same entry under Unix would look like this:

```
MY.FILE      /files/xyz/my.file  XXX YYY
```



**Note:** Under Unix, LISTSERV does not observe case-sensitivity internally. Therefore you cannot define two different files with the same non-case-sensitive filename. In other words, LISTSERV will not differentiate between `MY.FILE` and `my.file`, or even `My.File`. But note carefully that the physical files you store must be named in lower-case; in other words, the output of an `'ls'` command must show `my.file`, not `MY.FILE` or `My.File`. LISTSERV will handle this issue automatically when you `PUT` the files, but be forewarned if you store the files on the server via `ftp` or the Unix file system.)

Finally, here is an OpenVMS example:

```
MY.FILE      XYZ:[FILES]MY.FILE  XXX YYY
```

The first item, `MY.FILE`, is the name by which the file is known to LISTSERV. That is, the users will use `GET MY.FILE` to order a copy of that file. The name should contain only one period.

The second item, for instance `C:\FILES\XYZ\MY.FILE`, is the name LISTSERV will use for the actual disk file, in native OS format. Note that the directory must be created before you register the file. For security reasons, LISTSERV will not create the directory (or set the protections) for you. (LISTSERV will normally need full access to these files.)

The third and fourth items are "File Access Codes" (FACs). The first is for read accesses, and the second for writing. The following file access codes are available for non-VM servers (for VM FAC codes, see Section 8.3.5 [File Access Codes \(FAC\) for User Access \(VM Systems Only\)](#)):

- `ALL` – Universal access.
- `CTL` – Only the LISTSERV maintainers have access.

- PRIVATE (xxx) – Only members of the xxx list have access.
- OWNER (xxx) – Only the owners of the xxx list have access.
- SERVICE (xxx) – Only users in the service area of the xxx list have access.
- NOTEBOOK (xxx) – Same access as the archives of the xxx list.
- user@host – The user in question is granted access.

Except for ALL, which must occur on its own, multiple file access code entries can be specified, separated by a comma with no intervening space. For instance:

```
MY.FILE C:\FILES\XYZ\MY.FILE JOE@XYZ.EDU, JACK@XYZ.EDU, PRIVATE(XYZ-L) CTL
```

defines a file that Joe, Jack and the subscribers of the XYZ-L list can order via the GET command, but that only the LISTSERV administrator can update.



**Important:** These "file access codes" apply to LISTSERV commands (GET, PUT, INDEX) only, and not to the workstation or PC's file security system. It is your responsibility to protect the actual disk file by setting the file protections for the directory in which they are created.

### 8.4.2 Delegating File Management Authority

The sub-catalog enhancement allows the LISTSERV administrator to delegate file management authority in a controlled and secure manner. Multiple list owners can be given the authority to maintain their own sub-catalog in a predefined directory. With the LISTSERV-ISP add on (under development), a quota can be imposed on the directory in question.

The procedure works as follows:

1. The LISTSERV administrator creates the sub-catalog and identifies the directory where the files will be stored, and the person(s) who will be in charge of managing it ("catalog owners").
2. The catalog owners use the GET and PUT commands to update their catalog and register new files in their directory. Each file has the usual GET and PUT file access codes, allowing the catalog owners to further delegate the management of individual files to third parties ("file owners").
3. The file owners manage the files in question using the GET and PUT commands. Authorized users can retrieve the files using the GET command.



**Note:** This functionality is available in the VM version, using a different syntax. See Section 8.3 [Filelist Maintenance \(VM Systems Only\)](#) for information on managing the VM file archive system.

If you are migrating from VM to one of the non-VM versions of LISTSERV, please note that it is not necessary to create a subcatalog file for WELCOME, FAREWELL and MAILTPL files. If a subcatalog for these files is not created, they do not appear in the output of an INDEX command. However, there are two ways to force them to appear:

- As the result of an INDEX command without qualifier: simply define the file in SITE.CATALOG.
- As the result of an INDEX listname command: simply define the file in the listname.CATALOG.

### 8.4.3 Creating a Sub-Catalog

To create a sub-catalog, the LISTSERV administrator edits the file called SITE.CATALOG (or site.catalog under unix) in LISTSERV's main directory (the directory where SYSTEM.CATALOG/system.catalog is located). A sub-catalog is defined as follows:

```
MY.CATALOG      /home/lists/xyz/my.catalog  ALL  JOE@XYZ.COM
(1)              (2)                    (3)      (4) (5)
```



**Notes:** The name must end in '.CATALOG', but otherwise it can be anything. In particular, there does not need to be a list by that name.

The directory specification indicated for the catalog file (e.g., /home/lists/xyz) is where ALL the files defined in the sub-catalog will be stored. DO NOT USE LISTSERV'S MAIN (A) DIRECTORY FOR THIS PURPOSE! The catalog owner will be given FULL ACCESS to all the files in this directory, so make sure to create a new, empty directory. If the sub-catalog is being set up for a list owner, it may be a good idea to put the list archives and the sub-catalog in the same directory.

A file name must be provided for the sub-catalog file itself. This name, however, does not need to match.

This file access code controls the authority to INDEX the sub-catalog. This will also be the default GET access code for all the files registered in the sub-catalog.

This file access code defines the catalog owner(s) and default file owner(s) for all the files in the sub-catalog.

There is no need to reboot LISTSERV after updating the SITE.CATALOG file. Also, bear in mind that you are responsible for the OS-level security of the directory you create for the catalog. The file access codes in SITE.CATALOG only affect operations that go through LISTSERV; it is your responsibility to make sure that other users of the computer are given the appropriate access level to any directory you create for LISTSERV's purposes.

### 8.4.4 Updating the Sub-Catalog

Once the sub-catalog is created, the catalog owner(s) can register new files using the following procedure (in this example, it will be assumed that the sub-catalog is called MY.CATALOG):

1. Send a GET MY.CATALOG command to LISTSERV (or, if the catalog is brand new, start from an empty file).
2. Register new file(s) in the catalog (see below).
3. Use the PUT MY.CATALOG PW=XXXXX command to store the updated catalog.

Alternatively, if the catalog owner has an account on the LISTSERV host system and write access to the directory associated with the sub-catalog, the file can be edited

directly. Note however that, in that case, the LISTSERV-ISP quota system will be inoperative as it has no control over disk accesses which do not go through LISTSERV itself.

The format of sub-catalogs is similar to that of SITE.CATALOG:

```
MY.FILE           my.file           ALL JOE@XYZ.COM
(1)              (2)              (3) (4)
```



**Notes:** (1) This defines the name of the file as seen by LISTSERV users. That is, the command to retrieve the file will be GET MY.FILE.

(2) This defines the name of the actual disk file where the contents of MY.FILE will be stored. Normally, you should specify the same as (1), or just an equal sign (LISTSERV will then substitute the name you provided for (1)). However, in some cases you may want to make a particular file available under multiple names. This can be done by registering multiple files (ie multiple values for (1)), and using the same (2) value every time.

(3) This file access code determines who can order the file through a GET command. See Section 8.4.1 [Adding Files to the SITE.CATALOG](#) for more information on FAC codes.

(4) This file access code determines who can update the file with the PUT command. See section 8.4.1 [Adding Files to the SITE.CATALOG](#) for more information on FAC codes.

(2) defaults to the value of (1), and (3) and (4) default to the GET and PUT access codes of the sub-catalog itself, respectively. So, in most cases a sub-catalog entry will be as simple as:

```
MY.FILE
```

Additionally, comment lines (starting with an asterisk) or blank lines can be interspersed with file definitions. These comments will be echoed when the sub-catalog is indexed (see below), in sequence with the file definitions. For instance, your catalog could read:

```
*
* Files for the XYZ sub-project
*
XYZ.AGENDA
XYZ.BUDGET
XYZ.PROPOSAL-1
XYZ.PROPOSAL-2
```

### 8.4.5 Indexing the Sub-Catalog

If MY.CATALOG is defined as:

```
MY.CATALOG      /home/lists/xyz/my.catalog      xxx JOE@XYZ.COM
```

then any user who matches the 'xxx' file access code is authorized to issue an INDEX MY command to get a formatted version of the catalog. For compatibility with older versions of LISTSERV, GET MY.FILELIST will produce the same results. If there is a mailing list called MY, a list of the archive files will be appended automatically.

## 8.5 Storing Files on the Host Machine



**Note:** LISTSERV does not currently recognize "attachments" created by many popular mail clients as files to be stored with the PUT command. Such files must be part of the body of the message that contains the PUT command. This means that binary files must be stored either in 7-bit format (uuencoded, etc.) or ftp'd to the server and placed in the appropriate directory by the LISTSERV maintainer or other privileged user.

If you store binary-format files on the server, you should be careful to note in the file catalog or filelist that users who want to GET the files will need to use an F= modifier (e.g., GET BINARY.FILE F=MIME/APPL) when ordering them by email.

To store a file on any LISTSERV host, first ensure that it has been registered with an entry in a filelist or the site catalog. Then mail the file to LISTSERV with a single line at the top of the document. Follow the directions below:

1. Edit your file and save it. Add a single line at the top of the file as follows (square brackets indicate optional parameters):

```
PUT filename extension [filelist|catalogname] PW=XXXXXXXX
```

(This line will not appear to people who GET the file from LISTSERV.) Replace XXXXXXXX with your personal password. If you specify the filelist or catalog name, do not put the square brackets around the name.

There are a couple of issues that need to be noted here:

- If the file you are going to store is registered in the sitewide catalog or filelist, do not specify the name of the catalog or filelist.
- If the file you are going to store is registered in a sub-catalog or filelist other than the sitewide one, you may have to specify the name of the sub-catalog or filelist in order to be able to store the file. This is because it is entirely possible that two lower-level filelists or catalogs may have files registered with the same name (for instance, README.TXT). If LISTSERV has two sub-catalogs registered (for instance, MYLIST CATALOG and HISLIST CATALOG) that both have a file called README.TXT registered, then a PUT README.TXT command will tell LISTSERV to try and store the file in the first catalog it comes to in the hierarchy. If MYLIST CATALOG is registered before HISLIST CATALOG in SITE CATALOG, LISTSERV will try to store the file as if it belonged to MYLIST (which we assume is what you want). However, if HISLIST CATALOG is registered before MYLIST CATALOG (and many sites

like to keep things in alphabetical order, so this is a most likely scenario), LISTSERV will try to store the file as if it belonged to HISLIST, and you will get an error stating that you aren't allowed to store the file.

- You MUST turn off your signature file (if one is enabled in your mail client) in order to successfully store files. If you do not, LISTSERV will store your signature file at the end of the file.
2. Be sure that the file has been registered with an entry in a filelist or the site catalog.
  3. Be sure that you have defined a "personal password" to LISTSERV with the `PW ADD` command before you `PUT` the new or edited file. If you have done this but can't remember the password, send a `PW RESET` command to LISTSERV, then a new `PW ADD` command.
  4. Send the mail message to LISTSERV.

## 8.6 Deleting Files from the Host Machine

To delete a registered file on any LISTSERV host:

1. Create a new mail message addressed to LISTSERV. Add a single line at the top of the message as follows:

```
PUT filename extension [filelist|catalogname] PW=XXXXXXXX
```

(Replace `XXXXXXXX` with your personal password.) The same issues noted in Section 8.5 [Storing Files on the Host Machine](#) regarding the filelist/catalog name are operative here.

You MUST turn off your signature file (if one is enabled in your mail client) in order to successfully delete files. If you do not, LISTSERV will store your signature file in place of the file you are trying to delete instead of deleting the file.

2. Be sure that you have defined a "personal password" to LISTSERV with the `PW ADD` command before you `PUT` the delete job. If you have done this but can't remember the password, send a `PW RESET` command to LISTSERV, then a new `PW ADD` command.
3. Send the mail message to LISTSERV.
4. LISTSERV will tell you that the file has been successfully deleted.
5. For VM Systems ONLY: `GET` the `listname FILELIST` for your list and delete the line for the file you've just deleted. `PUT` the `listname FILELIST` back on the server.
6. For Workstation and PC Systems ONLY: `GET` the `listname.CATALOG` for your list and delete the line for the file you've just deleted. `PUT` the `listname.CATALOG` back on the server. Note that this is not necessarily required since under non-VM, if the physical file does not exist, LISTSERV will not include it in the output of an `INDEX` command. This is primarily a housekeeping measure.

## 8.7 Automatic File Distribution (AFD) and File Update Information (FUI)

AFD and FUI have not yet been ported to the workstation and PC environments. However, this feature is supported on VM and will be supported in the near future on the other platforms.





**Note:** If you are running LISTSERV under unix, Windows, or VMS, please skip the rest of this section as it does not pertain in any way to your implementation of LISTSERV.

These two features are similar in their command syntax, but do different things. AFD provides a method whereby users may subscribe to specific files, which will be sent to them any time the files are updated. For instance, if you have a FAQ file that is updated monthly, a user could send an AFD subscription to that FAQ file and LISTSERV would send it to the user every time you updated and stored the FAQ.

FUI, on the other hand, is a method whereby a user subscribes to a file but receives only a notification that the file has been updated. The user can then GET the file when applicable.

AFD and FUI can be password-protected to protect users from network hackers who might forge mail from the user subscribing to large or frequently-updated files. If a password is not provided in an AFD ADD or FUI ADD command, LISTSERV warns the user that it would be a good idea to password protect the subscription.

## 8.8 File "Packages"

You can define a group of files as a "package" that can be retrieved by users with a single GET command. First, ensure that all the files in the package are defined in the appropriate filelist and stored on the server as detailed above.

Next, create a file descriptor in the appropriate filelist or catalog for a file called *filename* \$PACKAGE (or *filename*. \$PACKAGE for non-VM), where *filename* is the name you have chosen for the group of files. Be sure that the filetype or extension is \$PACKAGE, with a leading \$ sign, and store your filelist.

Now create the actual *filename* \$PACKAGE file. At the top of the file you can insert comment lines beginning with asterisks, for example:

```
* MYLIST $PACKAGE
* Packing list for MYLIST PACKAGE
*
* You can make other comments here, such as
* the contact email address.
*
* filename filetype filelist
*=====
```

Following these comment lines, you insert lines for each of the files contained in the package. There are two ways to format entries in your \$PACKAGE file:

- A "compatibility" mode that works on all platforms, and which is identical to the original method used on VM (and which VM servers still must use). In the compatibility mode the basic format for the entries is

```
filename filetype filelist <optional_comments>
```

for example,

```
MYLIST    $PACKAGE MYLIST The packing list
INTEREST FILE      MYLIST Interest groups
```



```
NETIQUET FILE      MYLIST How to behave
ANOTHER FILE      MYLIST No comment
```

- In the second (new) mode for non-VM servers only, the entries are formatted like this:

```
filename.extension <optional_comments>
```

for example,

```
MYLIST.$PACKAGE The packing list
INTEREST.FILE Interest groups
NETIQUET.FILE How to behave
ANOTHER.FILE No comment
```



**Note:** Anything that is not the name of a file in the package must be commented out with an asterisk in the left-most column of the line. It is possible to create a package file without any comment lines at all, but this is not preferable in practice. Often users will get the package file itself just to see what is in it. You should include a reference to the package file itself so that the user will get a copy of the "packing list" to check against the files he receives from LISTSERV.

The final step is to send the package file to LISTSERV like any other file.

Now users can do one of two things:

1. They may get the entire package of files sent to them by sending LISTSERV the command GET filename PACKAGE (without the \$ sign); or
2. They may request that LISTSERV send only the package file itself by sending LISTSERV the command GET filename \$PACKAGE (with the \$ sign).

Packages may be subscribed to with the AFD and FUJ commands (VM only).

## 8.9 Finding More Information on File Archives

Other guides that refer to File Archive setup and maintenance are found at:

<http://www.lsoft.com/resources/manuals.asp>.

## 8.10 Notebook Archives

Notebook archives are files in which postings to the list are stored (assuming that notebooks are enabled for the particular list). In general, they are managed automatically by LISTSERV, with certain functions left to the list owner(s). For instance, there is no need to register notebook archives in the *listname.FILELIST* or *listname.CATALOG*; this is taken care of automatically.

### 8.10.1 Setting Up Notebook Archives for a List

Setting up notebook archives requires only a few steps:

1. Make sure that you have disk space for the notebook archives and that the directory in which they will reside has been created with appropriate security privileges. LIST-

SERV needs read and write access to any directory it uses for notebooks. Note that, for security reasons, LISTSERV will not create the directory if it does not exist.

2. Add the Notebook= keyword to the list header with appropriate settings. (If you are not the LISTSERV maintainer, you will have to ask the LISTSERV maintainer to do this for you.)
3. Store the list header back on the server.

For instance, let's assume you have a list called MYLIST running on a unix server and you wish to store its archives in a directory called `/usr/listserv/home/mylist-archive`. Notebooks are to be kept on a monthly basis and are to be available to anyone. Your Notebook= keyword would look like this:

```
* Notebook= Yes,/usr/listserv/home/mylist-archive,Monthly,Public
```



**Note:** Only the LISTSERV maintainer may change the location of Notebook archives (or change Notebook= No to Notebook= Yes). Anyone else attempting to PUT the list header after changing these values will result in the following message being sent in response:

```
The following problems have been detected in the list header:
```

```
* Notebook= ...
```

```
Error: The first two parameters of the "Notebook=" keyword may only be updated by the LISTSERV administrator.
```

```
Please refer to the list keyword documentation (available via the "INFO KEYWORDS" command) for more information about keyword syntax.
```

```
PUT operation rejected, old list remains unchanged.
```



**Note:** This output will appear either if an attempt is made to change "Notebook= No" to "Notebook= Yes", or if an attempt is made to change the location where notebook archives are stored on the server, by anyone who is not a LISTSERV maintainer.

Similar restrictions also apply to the Digest= keyword. See the [List Keyword Reference](#) document for details.

### 8.10.2 Migrating Old Notebook Archives to a New Site (LISTSERV to LISTSERV)

If migrating old notebook archives from one LISTSERV site to another, you can simply ftp (in TEXT mode) the notebooks from the old host to the new host, put them in the directory reference in the Notebook= keyword settings, and LISTSERV will immediately recognize their presence. You can also migrate the notebooks with GET and PUT.

### 8.10.3 Migrating Old Notebook Archives (Non-LISTSERV to LISTSERV)

LISTSERV notebooks follow a modified VM MailBook format, which is as follows:

A line of 73 "=" signs (ASCII 0x3D)

RFC822 headers, starting with the Date: header<sup>1</sup>

Blank line (actually part of RFC822 headers)

Message body

For instance:

```
=====  
Date:      Fri, 6 Mar 1998 17:05:01 -0500  
Sender:    Test list <TEST@XXXXXX.NET>  
From:      Nathan Brindle <nathan@XXXXXX.NET>  
Mime-Version: 1.0  
Content-Type: text/plain; charset="us-ascii"
```

This is a test.

```
=====  
Date:      Thu, 12 Mar 1998 13:23:07 -0500  
Sender:    Test list <TEST@XXXXXX.NET>  
From:      Nathan Brindle <nathan@XXXXXX.NET>  
Subject:    Test
```

This is another test

```
=====  
Date:      Thu, 12 Mar 1998 13:24:58 -0500  
Sender:    Test list <TEST@XXXXXX.NET>  
From:      Nathan Brindle <nathan@XXXXXX.NET>  
Subject:    Test 3  
Mime-Version: 1.0  
Content-Type: text/plain; charset="us-ascii"
```

Yet another test.

```
=====  
The last message in the archive is not followed by a separator line (in other words, the  
separator line is found at the beginning of each message, not at the end of each  
message).
```

If you can reformat your non-LISTSERV archives this way then you can rename them using standard LISTSERV filenames:

- For monthly archives: *listname.logyyymm*
- For weekly archives: *listname.logyyymmw*
- For yearly archives: *listname.logyy*

---

1. Note that by default, LISTSERV requires that the RFC822 Date: header be the first header in the message. If some other header or headers come before Date:, LISTSERV will not properly delimit the messages in the notebook.

You can work around this restriction for archives migrated from non-LISTSERV systems by using the list header keyword "Notebook-Header= Full"; however you must still remove unix mailbox separator lines like the one shown at the end of this section or LISTSERV will not be able to correctly index the notebook.

It should also be noted that lists with "Notebook-Header= Full" consume more disk space for very little gain, which is why the default is "Notebook-Header= Short", and why we suggest reformatting such migrated archives to the LISTSERV "short" header format.

(where *yy* = 2 digit year, *mm* = 2 digit month, *w* = letter A-F denoting the week of the month), place them in the directory pointed to by the Notebook= keyword for the list, and LISTSERV will index them and make them available via the web archive interface and so on.<sup>1</sup> In order for the web archive interface to notice new notebook files you must either GET and PUT the list header or restart LISTSERV.

If a list owner is planning to store archive files via the PUT method, the LISTSERV maintainer must first make dummy files with the same filenames in the list's notebook directory so that LISTSERV will not say that the file does not exist and reject the PUT operation. However please note that you should not make entries for the notebooks in *listname.catalog* (if one exists). LISTSERV makes its list of notebooks "on the fly" every time an INDEX command is issued for the list.

If your old archives have lines at the beginning of each message like this:

```
From userid@host.com Thu Feb 2 15:27:02 1995
```

you should delete them; this is the message separator used by sendmail. LISTSERV does not use it and it may in fact cause problems with indexing if left in.

### 8.10.4 Deleting Old Notebook Archives

The LISTSERV maintainer may delete old notebook archives that are no longer needed in one of two ways:

- Use standard file system commands from the console prompt to delete the files. On VM, use CMS ERASE; on Unix, rm; on VMS, DEL; on Windows systems, DEL or ERASE.
- Send a PUT command by itself (in essence, you are storing a zero-length file) via mail to LISTSERV. For instance:

```
PUT MYLIST LOG9607 PW=mypersonalpw
```

by itself would delete the file MYLIST LOG9607.

Two important issues:

- This command MUST be issued by email. It cannot be issued via the "Execute a LISTSERV command" facility of the web management interface.
- You MUST turn off your signature file (if one is enabled in your mail client) in order to successfully delete files. If you do not, then LISTSERV will store your signature file in place of the file you are trying to delete instead of deleting the file.

---

1. If your old archives are from Majordomo or ListProc you might want to look at an unsupported Perl script found at <ftp://ftp.lsoft.com/CONTRIB/notebook-conv.pl> which converts sendmail-style notebooks to LISTSERV format. This is really ListProc-specific but it would probably work with Majordomo archives. In any case this user-contributed script is not supported in any way by L-Soft, so please direct any questions about it to its authors.

### 8.10.5 Indexing existing notebook archives

LISTSERV creates the notebook archive index "on the fly" as required. If there is an existing *listname*.FILELIST or *listname*.CATALOG, it appends the index of notebook archives to the end of the index of other files. Otherwise, the index of notebooks is generated and sent by itself. The user simply issues the command `INDEX listname` to receive the index of available files and notebooks.



## Section 9 Creating and Editing Mail and Web Templates

Significant changes were made both to the mail template processor and to the default mail templates themselves for the version 14.3 release and following. In the main, these changes allow an unprecedented level of control over what were previously hard-coded, non-optional internal messages sent by LISTSERV in response to various commands.

It is now possible to define site-wide defaults for all template forms without having to edit the DEFAULT MAILTPL file, which is not upgrade-safe. Any customizations made at the site level to the default templates now go into the site-level SITE MAILTPL file, which will not be disturbed during an upgrade. Defining site-wide defaults in the old `$$SITE$ MAILTPL` file, which in any case did not work for all templates and was never intended for this use to begin with, is now deprecated, and such usage is no longer supported in LISTSERV 15.

During startup, LISTSERV will migrate existing templates from `WWW_ARCHIVE MAILTPL` to `SITE MAILTPL`, unless one of the following conditions arises:

- A conflicting template (same name, different contents) is already in `SITE MAILTPL`
- The copy of the template in `WWW_ARCHIVE` matches the first template in the system search order.

If there are no conflicts, `WWW_ARCHIVE MAILTPL` is then renamed to `WWW_ARCHIVE OLDTPL`. If a conflict is detected, LISTSERV will not attempt to determine which version of the template is "correct", but rather will log something like the following:

```
7 Oct 2004 10:57:05 Migrating templates from WWW_ARCHIVE to SITE...
- $TEST_TEMPLATE: conflicting version found in SITE
7 Oct 2004 10:57:05 Conflicts detected, finish migration manually
```

It is then incumbent on the site maintainer to harmonize the differing versions of any templates reported to be in conflict.

### 9.1 Using LISTSERV Templates

Templates are used to generate some of the mail LISTSERV sends to users in response to commands it receives. Among these are the "You are now subscribed . . ." message, the message sent to users when LISTSERV cannot find a subscription for them in a specified list, and others. Note that certain administrative mail (for instance, the response to the `STATS` and `RELEASE` commands) is hard-coded into LISTSERV and cannot be changed.

Other templates are used to generate the HTML code used by the web archive and administration interfaces.

A word about nomenclature: When we talk about "templates" we are talking about "files that contain one or more template forms", in other words, files like `DEFAULT MAILTPL` or `DEFAULT WWWTPL`. A "template form" is an individual section of a template which begins with a title line (three ">" symbols followed by a space, the name of the template form, and (optionally) a short description of the template, which for some template forms is also used as the subject of the mail LISTSERV constructs with the template form), followed by one or more lines of copy and/or imbedded commands, and ends at the next



title line or the end of the file, whichever is reached first. A template may contain one or more template forms.

## 9.2 Getting Copies of the Default Template Files

LISTSERV stores its default mail template information in a file called DEFAULT MAILTPL, which can be requested by list owners and LISERSERV maintainers from LISERSERV with the `GET` command, just like any other file. The LISERSERV maintainer will find this file in LISERSERV's "A" directory (usually `~listserv/home/default.mailtpl` on unix, `LISERSERV\MAIN\DEFAULT.MAILTPL` on Windows systems, and `LISERSERV_ROOT:[MAIN]DEFAULT.MAILTPL` under VMS).



**Note:** DEFAULT MAILTPL contains the (obsolete) static web interface template forms.

LISTSERV stores its default dynamic web interface template forms in a file called DEFAULT WWWTPL, which can be retrieved in a manner identical to that for DEFAULT MAILTPL.



**Note:** It is considered unwise (and it is not supported) to modify the contents of DEFAULT MAILTPL or DEFAULT WWWTPL themselves, as these files will be overwritten by upgrades. It is possible to make sitewide changes that will not be overwritten without disturbing either of these files.

All template forms may be customized using the built-in tools found in the web administration interface described in Section 11 [Using the Web Administration Interface](#). Edited template forms are placed in site-level or list-level template files that will not be overwritten by software upgrades.

## 9.3 Types of Templates

There are three different types of templates – Mail Templates, Message Templates, and Message Fragments.

### 9.3.1 Mail Templates

A mail template is a complete mail message. All commands are available, substitutions that make sense in the context of the specific message are available, the message may be formatted, and while other templates may be imbedded with the `.IM` command, the message is in and of itself ready for LISERSERV to send.

### 9.3.2 Message Templates

The next level down is a message template, i.e. a template that by default does not send any mail but supplies text that will ultimately be shown to the user - not always by e-mail, it could be over the web interface for instance.

Typically, message templates that are sent by mail will be "bundled" into a single message. However, a given message template can be forced to send an individual unbundled mail by using the new `.SM` command.

In a message template, you have the following restrictions:

- The commands `.TO/.CC/.RE/.CS` are not allowed unless you use `.SM`.

- Normally there will be limited access to formatting unless stated otherwise in the template. By "limited" is meant that you will, at a minimum, have the capability to break the text into separate paragraphs and to use .FO ON/OFF, although leading, trailing and duplicate blank lines will be removed.
- Unless stated otherwise, .QQ will suppress the message. Note that this does not mean that other related messages are also suppressed, especially if they have already been sent. .QQ only suppresses the template you are currently reading.

### 9.3.3 Message Fragments

The lowest level is a message fragment. It could be a list of months in French, or a common sequence of words that is put in a template not just to make other templates more readable, but to improve consistency (see &MSGREF). This fragment is used in another messages and is not itself a message, hence the following restrictions:

- .SM is not supported in message fragments (and thus other commands related to sending mail, for instance, .TO/.CC, etc. are not supported, either). If .SM is specified in a message fragment, an error similar to the following is raised in the web interface:

```
>>> Mail template "MSG_SUBSCRIBE_FRAGMENT_OPTCHANGED1" does not support .SM
```

and no email is actually sent. In the example case, .SM would have to be specified in the message template MSG\_SUBSCRIBE\_SUCCESS in order for it to succeed.

- .SJ is ignored in message fragments The .SJ command should be issued in the template that sends the actual message, not in the fragment.
- Formatting (.FO ON/OFF) is not allowed. Everything will be internally merged into a single line (this is similar to what was previously called a "linear" template).
- .QQ is not supported in a message fragment.

## 9.4 Naming Conventions for Message Templates and Fragments

The following naming convention for message templates and message fragments has been introduced:

MSG\_command\_action\_name

- 'command' is the name of the command. For instance, the name of a template that controls list postings starts with "MSG\_POSTING\_".
- 'action' is some kind of general action or category of action or the like. Every template with the same command and action SHOULD have the same calling conventions, in terms of what is allowed or not allowed, how the results are used, what global variables are available. Of course, each individual template may also have one or more variables that are specific to it. For instance, when a virus is detected, you will have the name of the virus, which is not available when "Sizelim=" is exceeded. If 'action' is something\_FRAGMENT, the message is a fragment.
- 'name' is a unique identifier.

## 9.5 Mail Template Format and Embedded Formatting Commands

### 9.5.1 Mail Template Format

Each individual template form starts with a form name and subject line, such as:

```
>>> EXAMPLE1 This is the subject line
```

The following instructions are reminders for the form name and subject line:

- The template form starts with the line containing the form name and subject, and ends with the next line starting with '>>>', or at the end of the file.
- The subject line may contain substitutions (such as "&LISTNAME: &WHOM requested to join").
- Ensure that there is a blank space (ASCII 0x20) between '>>>' and the name of the form, or LISTSERV will not recognize the form.
- The names of the template forms must be typed in UPPER CASE.

A template form contains text and, optionally, formatting/editing commands, which start with a period in column 1. All other lines are treated as normal text: sequences starting with an & sign are substituted, then lines are joined together to form a paragraph, which is finally formatted like with any non-WYSIWYG text processor. You can suspend formatting with `.FO OFF` and resume it with `.FO ON`; when formatting is suspended, LISTSERV no longer joins lines to form a paragraph, but simply writes one line of text to the message for each line read from the template form. This makes it possible to include tables or a text-mode logo, but can create seriously imbalanced text if substitutions are used. For instance, a typical &WHOM substitution can range from a dozen characters to 60 or more, even though it only takes up 5 characters on your screen when you enter it.

### 9.5.2 Common Variable Substitutions

The following substitutions are always available:

- &DATE – Long-style date (04 Jan 1998)
- &TIME – hh:mm:ss
- &WEEKDAY – Three-letter day of the week, in English
- &MYNAMES – The substitution you will use most of the time when you need to refer to LISTSERV. For Internet-only or BITNET-only servers, this will display LISTSERV's only e-mail address. For servers with both Internet and BITNET connectivity, it will say "LISTSERV@hostname (or LISTSERV@nodeid.BITNET)".
- &MYSELF – LISTSERV's address, in the form LISTSERV@XYZ.EDU or, if no Internet hostname is available, LISTSERV@XYZVM1.BITNET.
- &MYNODE – LISTSERV's BITNET nodeid, without the '.BITNET', or its Internet hostname if no NJE address is available.
- &MYHOST – LISTSERV's Internet hostname or, if none is available, its NJE address (with '.BITNET').

- `&MBX(addr)` – Looks up the specified address in LISTSERV's signup file and displays "name <addr>" if a name is available, or just the original address otherwise. This is typically used to give the name of the command originator or target, along with his e-mail address: `&MBX(&WHOM)` or `&MBX(&INVOKER)`. Please note however that `&WHOM` and `&INVOKER` are not always available in every template. The "addr" parameter is always required; `&MBX` by itself is syntactically invalid.
- `&RELEASE` – LISTSERV's release number (e.g., "15.0").
- `&OSTYPE` – The operating system under which LISTSERV is running.
- `&OSNAME` – The full operating system name including the version number, e.g., "VM/ESA 1.2.3", "Windows NT 4.0", "Linux 2.0.27", "SunOS 5.4", etc.
- `&HARDWARE` – The type of machine LISTSERV is running on, e.g., "Pentium (512M)".

The following substitutions are also available for templates related to mailing lists:

- `&LISTNAME` – Either the short or long name of the list based on the value of "List-Address=" and/or its system default. By default the long ("List-ID=") name is used if present.
- `&TITLE` – Title of the list, or empty string.
- `&KWD(kwd)` – Value of the specified keyword for the list. You do not need to specify the name of the list - it is implicit. You need not put quotes around the keyword names either, although quotes will be accepted if present. Optionally, you can specify a second numeric argument to extract just one of the terms of a list header keyword; for instance, if the list header contains "Notebook=Yes,L1,Monthly,Private", `&KWD(NOTEBOOK,4)` has the value "Private". A third argument, also optional, specifies the default value for the keyword in case it was not initialized. It is meant to be used for conditional formatting in the default templates and list owners should not worry about it.
- `&LITE` – Has the value 1 when running the LISTSERV Lite product, and 0 otherwise. This variable can be used to write generic templates that account for the differences between the two products.
- `&LITEFE` – Has the value 1 when running the Free Edition of LISTSERV Lite. Similar to but distinct from `&LITE`.
- `&ISODATE` – Returns today's date in ISO format, i.e., yyyy-mm-dd.
- `&DAYSEQ(n)` – Used to create FAQ templates with rotating topics. May also be used to create bottom banners with rotating text (e.g., for lists with multiple commercial sponsors who get "ad space" in the banner on a rotating basis).

In addition, many template forms have their own specific substitutions, meaningful only in their specific context. For instance, a message informing a user that he was added to a mailing list may have an `&INVOKER` substitution for the address of the person who issued the ADD command. This is not meaningful for a template form intended to inform a user that he must confirm his subscription to a list within 10 days, so it is not generally available. If you attempt to use a substitution which is not available, the template

processor writes an error message to the mail message it is generating, but sends it anyway, in the hope that the recipient will be able to figure out the meaning of the message in spite of the error.



**Important:** If you need to include a sentence with an ampersand character, you will have to double it to bypass the substitution process, as in "XYZ &&co."

The mail template processor also supports HTML-like variable closure, in addition to the traditional LISTSERV closure (both methods are supported concurrently; there is no need to select one over the other). For example:

- **Traditional** – For more information, please send mail to &EMAIL or call &PHONE.
- **HTML** – For more information, please send mail to &EMAIL; or call &PHONE;.

Previously, HTML writers who used HTML closure conventions would not get the expected results. This change makes it easier for webmasters to get the desired results the first time.

### 9.5.3 Template Commands

Any line starting with a period in column 1 is processed as a formatting command. Note that neither substitutions nor formatting commands are case sensitive. The table below lists the formatting commands that list owners may need to use.

*Table 9-1 Formatting Commands for List Owners*

Command	Description
. *	Comment: anything on this line is simply ignored. This is useful for recording changes to template files when there are multiple owners. Just add a comment line with the date and your initials every time you make a change, for the benefit of the other owners.
.CC OFF	Removes all "cc:" message recipients. You can also add message recipients by specifying a series of e-mail addresses after the .CC statement, as in .CC JOE@XYZ.EDU. PC mail users should note that in this context "cc:" is a RFC822 term that stands for "carbon copy". RFC822 messages may have "cc:" recipients in addition to their "primary" recipients. There is no real technical difference between the two, the "cc:" indicator just denotes a message that is being sent for your information. Some administrative messages sent to list owners are copied to the user for their information, and vice-versa; this behavior can be disabled by adding a .CC OFF statement to the template.
.CE text	Centers the text you specify (just the text you typed on the same line as the .CE command). This can be useful to highlight the syntax of a command.
.FO OFF	Turns off formatting: one template line = one line in the final message.
.FO ON	You can resume formatting with .FO ON or .FO RAGGed. (.FO RAGGed requires LISTSERV 1.8e-2002a or later, that is, build date of 31 October 2002 or later.)

Command	Description
<code>.FO RAGGed</code>	Changes right-justified text formatting to left justified text formatting. You can resume right-justified formatting with <code>.FO ON</code> . ( <code>.FO RAGGed</code> requires LISTSERV 1.8e-2002a or later, that is, build date of 31 October 2002 or later.)
<code>.QQ</code>	Cancels the message. LISTSERV stops reading the template form and does not send anything. This is useful if you want to completely remove a particular message; note however that this can be confusing with certain commands, as LISTSERV may say "Notification is being sent to the list owners" when in fact nothing will be sent because of the <code>.QQ</code> command in the template form.
<code>.QU</code>	Ends processing of the current template as if you had reached the end, but without cancelling the message. The main purpose is to avoid multi-level nested <code>.BB/.EB</code> conditional blocks (see below) that are hard to keep track of.
<code>.RE OWNERS</code>	Adds a 'Reply-To:' field pointing to the list owners in the header of the generated message. Use this command when you think users are likely to want to reply with a question. You can also use <code>.RE POSTMASTER</code> to direct replies to the LISTSERV administrator, if this is more appropriate.
<code>.TO</code>	Replaces the default recipients of a message with the value specified. For instance, if you use the <code>ADDREQ1</code> template form to send new subscribers a questionnaire, application form or similar material, you will need to add a <code>'.TO &amp;WHOM'</code> instruction to your modified template form, as by default the user will not receive a copy.

A number of more advanced commands are available to list owners with more sophisticated needs and some programming experience. If you encounter one of these commands in a template, you will probably want to leave it alone.

*Table 9-2 Advanced Formatting Commands for List Owners*

Command	Description
<code>.ASIS text</code>	Tells LISTSERV to leave the text immediately following the <code>.ASIS</code> directive alone, that is, don't convert "<" and ">" characters into HTML <code>&amp;lt;</code> and <code>&amp;gt;</code> when creating pages. This is specifically for use in HTML templates where it is important not to convert parts of a URL reference. For instance, <p><code>.ASIS Click &lt;a href="http://some.host.com/some-doc.html"&gt;here&lt;/a&gt;.</code></p> As with the <code>.CE</code> directive, the text you intend to affect with the <code>.ASIS</code> directive must not wrap. The <code>.ASIS</code> directive will only work on text it finds on the same physical line into which it is coded.
<code>.BB cond</code>	Begin conditional block. <p>Related commands are <code>.EB</code>, <code>.ELSE</code>, and <code>.QU</code>. See Section 9.5.4 <a href="#">Conditional Processing</a> for usage.</p> See also <code>.QUIF</code> .

Command	Description
.CS text	<p>Define a (non standard) character set for the template in question, i.e.,</p> <p>.CS ISO-8559-7</p> <p>This setting is ignored if the template does not actually contain special characters (for instance, if the template is written in 7-bit ASCII). Otherwise the appropriate headers are created for the message in question when it is sent out.</p>
.DD ddname	<p>Copies the contents of the specified DD into the message. This is meaningful only if a DD has been set up by LISTSERV for this purpose. As a rule of thumb, you should either leave these statements unchanged or remove them.</p>
.EB	<p>End conditional block (see .BB).</p>
.ELSE	<p>Conditional ELSE directive (see .BB).</p>
.IM name	<p>Imbeds (inserts) another template form at this point in the message. This is used to avoid duplicating large pieces of text which are mostly identical, such as the templates for "you have been added to list X by Y" and "your subscription to list X has been accepted".</p> <p>As noted below, LISTSERV will not pick up an "imbedded" template form from \$SITE\$.MAILTPL. If you wish to include an "imbedded" template form (e.g., \$SIGNUP) in \$SITE\$.MAILTPL, you must also include the template form that calls it with the .IM command.</p>
.QU	<p>Stop (i.e., QUIT) processing of the current template as if you had reached the end, but without cancelling the message. The main purpose is to avoid multi-level nested .BB/.EB conditional blocks that are hard to keep track of.</p>
.QUIF	<p>(QUIT IF) Similar to .QU, stop processing the current template without cancelling the message, based on the result of an inline conditional comparison. The full syntax is</p> <pre>.QUIF var operator value</pre> <p>For instance,</p> <pre>.QUIF &amp;RC = 0</pre> <p>This single statement is strictly equivalent to the block</p> <pre>.BB &amp;RC = 0 .QU .EB</pre>
.SE var text	<p>Defines or redefines a substitution variable. This is convenient for storing temporary (text) expression results which need to be used several times. Even standard variables such as &amp;LISTNAME can be redefined - at your own risk. You must enclose the text expression in single quotes if you want leading or trailing blanks.</p>



Command	Description
.SJ	<p>Set the subject line for the message. .SJ does two things:</p> <ol style="list-style-type: none"> <li>1. It overwrites the heading originally obtained from the '&gt;&gt;&gt;' header line starting the template. A mail template uses this heading as the "Subject:" field. A message template, on the other hand, does not use this heading for anything, but it is still changed internally. See .SM below.</li> <li>2. It leaves an explicit indication to the template invoker that it is desired to change the subject line. For a mail template, this has already happened. For a message template, on the other hand, the invoker is strongly urged to take reasonable steps to make it happen. "Reasonable" is defined by the invoker, not by the customer.</li> </ol> <p>.SJ is not available for message fragments. The subject line should be set in the template that creates the actual message.</p> <p>One important caveat when not using .SM: Most templates are going to be for command replies. A command reply may be followed by another reply to the same command (eg multi-user ADD), or to another command. All these replies are bundled into the same e-mail message sent in answer to the command(s), again unless you use .SM. And this message can only have one subject by definition.</p>
.SM	<p>Send a copy of the message via email, if echoed to the web interface.</p> <p>.SM is not available in message fragments and is only available in message templates which do not contain a comment to the effect that .SM is not supported for that particular message template.</p>
.TY text	<p>Types one line of text on the LISTSERV console log. This can be useful to the LISTSERV maintainer for debugging, and also to record information in the console log.</p>

### 9.5.4 Conditional Processing

LISTSERV mail templates can be programmed with an if/then/else logic that evaluates available data and performs the appropriate task depending on the outcome of the evaluation.

A conditional block begins with the command .BB (begin block) and ends with .EB (end block). In the simplest case, the boolean expression following .BB is evaluated and, if false, all the text between the .BB and .EB delimiters is skipped. For instance, from \$SIGNUP:

```
.bb &DEFOPT ^= ''
```

Following instructions from the list owner, your subscription options have been set to "&DEFOPT" rather than the usual LISTSERV defaults. For more information about subscription options, send a "QUERY &LISTNAME" command to &MYNAMES.

```
.eb
```

.ELSE may be specified in cases where an alternate text is required, as in this more complicated example (also from \$SIGNUP):

```
.bb ((&x ^= POSTMASTER) and (&x ^= OWNER)) and (&x ^= OWNERS)
Please note that it is presently possible for
.bb &x = PUBLIC
anybody
.else
other people
.eb
to determine that you are signed up to the list through the use of the
"REVIEW" command, which returns the e-mail address and name of all the
subscribers. If you do not want your name to be visible, just issue a
"SET &LISTNAME CONCEAL" command.

.eb
```

Additionally, it is possible to simply exit a conditional at an arbitrary point by using a .QU (QUit) command. For instance, the MSG\_POSTING\_REJECT\_BAD\_ATTACHMENT message template is used both to reject unwanted attachment types as well as viruses. In order to avoid processing the entire template (which contains other text that is not germane to a virus rejection), .QU is used to simply exit processing and send the message immediately.

```
.bb &VIRUS = 1
.* A virus was detected in the message. Note that this is only possible
.* for messages sent to a list, LISTSERV does not execute attachments so
.* messages sent to LISTSERV are not checked for viruses.
Your posting to the &LISTNAME list has been rejected because it contains
.bb &VIRUS_NAME ^= ''
the '&VIRUS_NAME;'
.else
a
.eb
.bb &VIRUS_FILENAME ^= ''
virus in attachment '&VIRUS_FILENAME;'
.else
virus.
.eb
You are strongly advised to check your computer for viruses as soon
as possible!
.qu
.eb
.* text for rejecting attachments follows
```



**Notes:** Conditional blocks may nest to an arbitrary depth.

The expression evaluator is recursive but not very sophisticated; the restriction you are most likely to encounter is that all sub-expressions have to be enclosed in parentheses if you are using boolean operators. That is, ".BB &X = 3" is valid but ".BB &X = 3 and &Y = 4" is not.

String literals do not require quoting unless they contain blanks, but quotes are accepted if supplied.

Comparison operators are = <> ^= IN and NOT IN (the last two look for a word in a blank-separated list of options, such as a keyword value). These operators are not case-sensitive; == and ^= are available when case must be respected. Boolean operators are AND and OR.

A command line in a conditional block must be contained on one physical line and may not wrap, so be careful when sending MAILTPL files back to LISTSERV that you do not accidentally wrap long .BB lines.

The operators =\* and ^=\* are available to perform wildcard matches in conditional blocks. For instance JOHN\_DOE@UNIX.EXAMPLE.COM =\* J\*DOE@\*EXAMPLE.COM is a true statement. The wildcard specification is on the right-hand side whereas the actual text (or variable) you are evaluating is on the left.

### 9.5.5 The .QUIF Command

.QUIF (QUIt IF) allows the invoker to stop processing the template if a certain condition is met, but without having to define a full-blown conditional block. For instance, the .IM command used to imbed one template into another returns a success code in the template variable &RC. If imbedding succeeds, &RC is set to 0, and something like the following is possible:

```
>>> MSG_POSTING_REJECT_BAD_ATTACHMENT Received an attachment not allowed
by "Attachments="
.* Use legacy BAD_ATTACHMENT template if present
.im *BAD_ATTACHMENT
.quif &rc = 0
.* at this point template processing stops and the message is sent if
&RC = 0
.* otherwise processing would continue with any following text.
```

"quif &rc = 0" is strictly equivalent to the block

```
.bb &rc = 0
.qu
.eb
```

### 9.5.6 Using 8-Bit Characters in Templates

If you include 8-bit characters (e.g., accented or national language characters) in templates, LISTSERV will automatically encode the templates on-the-fly using MIME quoted-printable encoding. While there is no guarantee that every mail program will be able to properly display 8-bit characters, those mail programs that do understand MIME quoted-printable encoding should have no trouble doing so.

## 9.6 Editing List-Level Default Templates

*Please note that list-level mail templates are not available in LISTSERV Lite.*

It is strongly recommended that the web interface customization tools be used to customize the "look and feel" of list templates. However, some administrators may wish to use the old method. If so, then simply make a copy of DEFAULT.MAILTPL on your local machine and name it listname.MAILTPL. Keep the original DEFAULT.MAILTPL around in case you make a mistake and need to start over.

At this point, you could theoretically store the listname.MAILTPL back on the LISTSERV host. However, without making any changes that would be somewhat pointless. At the very least you should edit the INFO template form before storing the template. Note also that you need only store the sections of the template that you have changed. For instance, if you edit the INFO template form but leave the rest of the template untouched, you can delete the rest of the template and store the INFO template form alone as listname.MAILTPL. The benefit to this approach is that any administrative changes to the rest of the default template are automatically applicable to your list as soon as they are made, rather than requiring that you edit your mail template individually to reflect such changes. If using the manual-editing procedure, L-Soft recommends that this approach be followed as the default.

### 9.6.1 The INFO Template Form

The INFO template form is LISTSERV's response to the command INFO listname. By default, it contains the following:

*Figure 9-1 Default Content of the INFO Template form for DEFAULT.MAILTPL*

```
>>> INFO Information about the &LISTNAME list
There is no information file for the &LISTNAME list. Here is a copy of the
list "header", which usually contains a short description of the purpose of
the list, although its main purpose is to define various list configuration
options, also called "keywords". If you have any question about the
&LISTNAME list, write to the list owners at the generic address:

.ce &LISTNAME-Request@&MYHOST

.dd &LISTHDR
```



**Note:** the replaceable parameters &LISTNAME and &MYHOST. Don't change &MYHOST; LISTSERV replaces it with the correct value for the name of the host site. &LISTNAME automatically inserts the name of the list. It's probably best to use &LISTNAME to refer to the list throughout the document rather than to replace it with something like "MYLIST-L". This ensures that the template form will be consistent with the default and will be simpler to debug should a problem arise. Also, in the event the name of the list changes, it will be unnecessary to edit the template form (although it would have to be renamed to match the new name of the list, of course).

Should it be desirable to replace the default INFO template form with information about the list, it is probably best to remove the .dd &LISTHDR line. This line instructs LISTSERV to read in the header of the list and add it to the response in lieu of any other

data about the list. Many list owners add descriptive comment lines to their list headers, thus this default.

Here is a minimally-edited sample INFO template form for a list called MONKEYS.<sup>1</sup>

*Figure 9-2 Sample of an Edited INFO Template Form*

```
>>> INFO Information about the &LISTNAME list
&LISTNAME is an open, unmoderated discussion list featuring monkeys. Things
such as how to care for a pet monkey, monkey diseases, monkey lore, endan-
gered species of monkeys, and monkey psychology are likely to be discussed.
The list is NOT intended for discussion of Darwinism and/or theories of
evolution.

If you have any question about the &LISTNAME list, write to
the list owners at the generic address:

.ce &LISTNAME-Request@&MYHOST
```

### 9.6.2 DEFAULT MAILTPL Templates



**Note:** We no longer provide a complete list of template forms in this section as there are simply too many of them. Templates are listed by name and description in the customization section of the web interface.

The following are template forms that can be defined, but which are not present by default in DEFAULT.MAILTPL. If you want to define them for a particular list, the easiest way to do so is via the web interface.

- `POSTACK1` (optional) – When present, this message is sent in reply to any message posted to the list. This is very useful for creating "infobots", or just for returning a standard acknowledgement to contributors. The `&SUBJECT` variable contains the subject of the original message, and naturally the usual substitutions (`&LISTNAME`, `&DATE`, `&TIME`) are available.
- `TOP_BANNER`, `BOTTOM_BANNER` (optional) – When these template forms are present, their contents are automatically inserted at the top (respectively bottom) of each and every message posted to the list. Typically, the top banner would be used for a copyright or short legal warning which absolutely has to be seen by each and every reader. The bottom banner could contain instructions for signing off the list, a disclaimer, an acknowledgement of a sponsor's contribution, a "tip of the week", etc.



**Documented Restriction:** The use in banners of substitutions which do not yield a constant result (e.g., `&TIME`) will defeat the duplicate mail detection part of LISTSERV's loop-checking heuristics in any case where a subscriber is forwarding all mail back to the list. L-Soft advises that such substitutions never be used in a `TOP_BANNER` or `BOTTOM_BANNER`.

LISTSERV does not attempt to remove bottom banners from individual messages in digests, for instance, which have been included as quoted material in responses.

1. Thanks to Marty Hoag of NEW-LIST.

- `TOP_BANNER_HTML`, `BOTTOM_BANNER_HTML` (optional) – When these template forms are present, they will be used "as is" for HTML message parts. If absent, the regular banner is used for HTML, probably with less than 100% satisfaction.



**Documented Restriction:** The use in banners of substitutions which do not yield a constant result (e.g., `&TIME`) will defeat the duplicate mail detection part of `LISTSERV`'s loop-checking heuristics in any case where a subscriber is forwarding all mail back to the list. L-Soft advises that such substitutions never be used in a `TOP_BANNER_HTML` or `BOTTOM_BANNER_HTML`.

- `CONTENT_FILTER` (optional) – When present, provides `LISTSERV` with a ruleset for message content filtering that can be configured at the list level. See Section 7.18 [Content Filtering](#) for more information on how to use content filtering.

### 9.6.3 Tips for Using Templates

- Many list owners require prospective subscribers to fill in a little questionnaire before being added to the list, or to explicitly state that they have read the list charter and agree to follow all rules or be removed from the list. The most convenient method, for both list owner and subscriber, is to have the `SUBSCRIBE` command return a copy of the questionnaire (or list charter, etc.), and not forward the request to the owner. The user answers the questions and returns them directly to the list owner, who then adds the subscriber manually. Naturally, it is more convenient for the user if this information arrives in a separate message, with a 'Reply-To:' field pointing to the list owner's address. Thus, you should not use the `SUB_OWNER` template form for this purpose, because it is a linear template form and does not give you any control over the 'Reply-To:' field. The `SUB_OWNER` template form could be modified to read "A copy of the list charter is being sent to you, please read it carefully and follow the instructions to confirm your acceptance of our terms and conditions." The list charter would then be sent separately, through the `ADDREQ1` template form. You would use the `.RE OWNERS` command to instruct `LISTSERV` to point the 'Reply-To:' field to the list owners, and `.TO &WHOM` to change the destination from list owner to subscriber. If you want to receive a copy of the message, you can use `.TO &WHOM cc: xxx@yyy`.
- When writing template forms, it is a good idea to use substitutions (`&XXXX`) for information which may change in the future. In particular, it is not uncommon for lists to have to be moved from one host to another, and this will be a lot easier if the template forms use substitutions for the list address and list host. The `&LISTADDR` substitution translates the full address of the list (`XYZ-L@XYZ.COM`), whereas `&LISTNAME` is just the name (`XYZ-L`). For references to the server and host, use `&MYHOST` for the Internet hostname, `&MYSELF` for the server address (normally `LISTSERV@&MYHOST`), and `&OWNER` for the xxx-request mailbox address. These substitutions are "universal" and can be used in all template forms. For instance, if you decide to make a bottom banner with instructions for leaving the list, the text could read: "To leave the list, send a `SIGNOFF &LISTNAME` command to `&MYSELF` or, if you experience difficulties, write to `&OWNER`."

## 9.7 Using the DAYSEQ(n) Function

The DAYSEQ(n) function is quite powerful. This function allows the list owner to code template forms (such as the PROBE1 or BOTTOM\_BANNER messages) that change or "rotate" automatically.

The DAYSEQ(n) function is invoked in a .BB - .EB conditional block, and n corresponds to the number of days in the rotation period, i.e., to the number of variations that you want to make to the text of the message. &DAYSEQ(n) returns a number from 1 to n which increases by 1 every day, with no special regard for weekends. That is, if the rotation period is to last for a week, you code DAYSEQ(7). If the rotation period is 15 days, you code DAYSEQ(15). Two examples follow:

### 9.7.1 Rotating Bottom Banner

To create a rotating bottom banner, follow this example. A list has three commercial sponsors, each of whom are provided with an advertisement every three days. (Note that this doesn't take weekends into account; in this example, if company A is featured in the banner on Monday, it will be featured again on Thursday and then again on Sunday. However, in the following week it will be featured on Wednesday, Saturday, and Tuesday, so it will actually get rather good coverage.) Our BOTTOM\_BANNER template form would look like this:

```
>>> BOTTOM_BANNER
.BB &DAYSEQ(3) = 1
Today's copy of the &LISTNAME newsletter has been brought to you by Company A.
.EB
.BB &DAYSEQ(3) = 2
Today's copy of the &LISTNAME newsletter has been brought to you by Company B.
.EB
.BB &DAYSEQ(3) = 3
Today's copy of the &LISTNAME newsletter has been brought to you by Company C.
.EB
```

If a company needs to get a higher percentage of "air" time than another, you can simply assign it more than one of the possible n values of &DAYSEQ(n). For instance, if you have two companies but one should get twice as many days of "air" time, you might code something like this:

```
>>> BOTTOM_BANNER
.BB (&DAYSEQ(3) = 1) OR (&DAYSEQ(3) = 3)
Today's copy of the &LISTNAME newsletter has been brought to you by Company A.
.EB
.BB &DAYSEQ(3) = 2
Today's copy of the &LISTNAME newsletter has been brought to you by Company B.
.EB
```

This would cause Company A's message to appear on days 1 and 3 of the rotation period and Company B's message to appear on day 2 only.



### 9.7.2 Rotating FAQ via the PROBE1 Template and "Renewal= xx-Daily"

Subscription renewal can be coded with daily granularity (however, please note that it is and remains inadvisable to use renewal intervals of less than a week). If you further code subscription probing into the "Renewal=" keyword with the ",Probe" parameter, you open up the possibility of turning the standard PROBE1 template form into a periodic FAQ. Here's how:

We'll assume to start that you will code "Renewal= 15-Daily,Probe" in your list header. (You can experiment with other numbers, but since we have two messages and will be using &DAYSEQ(2), we need an odd renewal period.) We'll also assume that you want to send two versions of your FAQ each month; the first, a complete FAQ document, and the second, an abbreviated "reminder" version that just contains information about how to sign off, how to post to the list, and so forth. The basic algorithm is therefore:

When &DAYSEQ(2) = 1, send the full FAQ.

When &DAYSEQ(2) = 2, as it will 15 days later, send the abbreviated FAQ.

Your PROBE1 template form would thus look like this:

```
>>> PROBE1 Periodic FAQ posting for &LISTNAME
&WEEKDAY, &DATE &TIME
.BB &DAYSEQ(2) = 1
```

This is the complete FAQ for &LISTNAME. Please read it and keep a copy for future reference. A FAQ document for &LISTNAME is distributed every 15 days, the full FAQ alternating with a shorter "reminder" FAQ.

```
<body of the full FAQ document>
.EB
.BB &DAYSEQ(2) = 2
```

This is the abbreviated FAQ for &LISTNAME. Please read it and keep a copy for future reference. A FAQ document for &LISTNAME is distributed every 15 days, the full FAQ alternating with a shorter "reminder" FAQ.

```
<body of the abbreviated FAQ document>
.EB
```

### 9.7.3 Calculating the Value for DAYSEQ()

When you first start using a rotating banner with the &DAYSEQ variable, the &DAYSEQ(n) = 1 period begins based on the number of days elapsed since a baseline. On VM (and in REXX generally) you can calculate today's value easily with:

```
/* */
say Date('B') + 1
```

If you do not have access to a REXX interpreter, Date('B') is described as "the number of complete days (that is, not including the current day) since and including the base date, 1 Jan 0001, in the format 'dddddd' (no leading zeros or blanks)."<sup>1</sup> It also is equal to the

---

1. Cowlshaw, Michael: The REXX Language: A Practical Approach to Programming, 2nd ed., p.92. Englewood Cliffs, NJ: Prentiss-Hall, Inc., 1990.

C language expression  $\text{time}(0)/86400 + 719162$  or, for OpenVMS users, to the Smithsonian base date plus 678575.

For example, for Friday 22 Oct 2004, the value of `Date('B') + 1` is 731876. This value increases by one every day at midnight.

### 9.8 Storing the <listname>.MAILTPL File on the Host Machine

The procedure differs slightly on VM systems, but the following will work for unix, VMS and Windows systems:

1. Get a copy of DEFAULT.MAILTPL and edit it.
2. Be sure that you have defined a "personal password" to LISTSERV with the PW ADD command before you PUT the template file. If you have done this but can't remember the password, send a PW RESET command to LISTSERV, then a new PW ADD command.
3. Send the file to LISTSERV with a `PUT listname MAILTPL PW=XXXXXXXX` command at the top of the file, just as if you were storing the list itself. Replace XXXXXXXX with your personal password.

The variation for VM systems is that the LISTSERV maintainer will have to create a fileid for the file before it can be PUT on the server and seen by LISTSERV.



**Note:** The LISTSERV maintainer can create and edit these files in place with any standard text editor. Changes made to template files in this way are available to LISTSERV immediately after they are saved.

### 9.9 DIGEST-H and INDEX-H Template Files

Two other template files that are available pertain to the automatic "digestification" feature. You may create and store files called listname DIGEST-H and listname INDEX-H. These files define custom digest headers and custom index headers, respectively. The DIGEST-H and INDEX-H files are plain text files, like the WELCOME and FAREWELL files, and the instructions for storing them on the server are identical. Note that, as with the WELCOME and FAREWELL files, you cannot use the template formatting commands and replaceable parameters discussed above.

*Figure 9-3 Typical Contents of a DIGEST-H or INDEX-H File*

```
The MYLIST list is sponsored by ABig Corporation.

See http://www.abig.com for information on ABig Corporation's products.
```

The contents of DIGEST-H and INDEX-H are appended to the digest or index, respectively, immediately following the list of topics.

*Figure 9-4 Sample DIGEST Output for a List with a DIGEST-H File*

```
Date: Tue, 11 Jun 2001 11:52:41 -0500
From: Automatic digest processor <LISTSERV@MYHOST.COM>
Reply-To: My test list <MYLIST@MYHOST.COM>
To: Recipients of MYLIST digests <MYLIST@MYHOST.COM>
Subject: MYLIST Digest - 10 Jun 2001 to 11 Jun 2001

There is one message totalling 10 lines in this issue.

Topics in this issue:

    1. Testing 125...3 sir!

The MYLIST list is sponsored by ABig Corporation.

See http://www.abig.com for information on ABig Corporation's products.
```



**Notes:** The INDEX-H output would be similar to the above figure, following the list of postings.

You can't add a digest or index "footer" because according to the standard, anything after the end of the digest text is supposed to be discarded.

## 9.10 WWW Interface Templates and Template Forms

The following describes the available template files and their respective template forms for the WWW archive and administration interface. L-Soft does not advise modifying these templates unless you know exactly what you are doing. If you modify the templates it is strongly recommended that you keep copies of the originals in a safe location for fall-back. In practice, this means that you should NEVER edit the default template files themselves.

### 9.10.1 Web Forms (Static) Contained in DEFAULT MAILTPL



**Note:** We no longer provide a complete list of template forms in this section as there are simply too many of them. Templates are listed by name and description in the customization section of the web interface.

### 9.10.2 The WWW\_ARCHIVE.MAILTPL File

In LISTSERV 14.3, this file became obsolete. LISTSERV will migrate existing templates from WWW\_ARCHIVE MAILTPL to SITE MAILTPL, unless one of the following conditions arises:

- A conflicting template (same name, different contents) is already in SITE MAILTPL; or
- The copy of the template in WWW\_ARCHIVE matches the first template in the system search order.

If there are no conflicts, WWW\_ARCHIVE MAILTPL is then renamed to WWW\_ARCHIVE OLDTPL. If a conflict is detected, LISTSERV will not attempt to

determine which version of the template is "correct", but rather will log something like the following:

```
7 Oct 2004 10:57:05 Migrating templates from WWW_ARCHIVE to SITE...
- $TEST_TEMPLATE: conflicting version found in SITE
7 Oct 2004 10:57:05 Conflicts detected, finish migration manually
```

It is then incumbent on the site maintainer to harmonize the differing versions of any templates reported to be in conflict.

### 9.10.3 The DEFAULT.WWWPTL File (Dynamic Templates)

The DEFAULT WWWPTL file contains the default templates for the parts of the WWW archive interface that are not defined in DEFAULT MAILTPL. Unless you have specific issues that need to be resolved (such as a national language preference or a need to point certain links to non-standard locations), it is strongly recommended NOT to edit this file. One reason for this is that DEFAULT WWWPTL will be overwritten by a software update. The safest approach to customizing the look and feel of your site is to use the customization features built into the web interface, which save your customizations in a different place.

When customizing these templates, there are two fundamental differences between them and the templates in DEFAULT MAILTPL:

1. Any substitution variable that you use (for instance, &LISTNAME) must be escaped with a "+" symbol between the ampersand and the name of the variable, thus: &+LISTNAME. (Note that, as with the regular mail template forms, not all substitution variables are available in every HTML template form.)
2. Any dot-formatting command you use (for instance, .CC , .BB , etc.) must have a "+" symbol rather than the dot, thus: +CC +BB



**Note:** We no longer provide a complete list of template forms in this section as there are simply too many of them. Templates are listed by name and description in the customization section of the web interface.

### 9.10.4 The SITE.WWWTPL File

The preferred method of editing site.wwwtpl is to use the web interface's built-in customization features to edit templates. L-Soft no longer recommends editing site.wwwtpl by hand.

However, if desired, you can override the default.wwwtpl file by providing a customized site.wwwtpl file in LISTSERV's A directory.

The site.wwwtpl file is the file in which edited web templates are placed. This prevents your site-wide definitions being overwritten in an upgrade (i.e., when default.wwwtpl will normally be overwritten). If a given template is found in site.wwwtpl, that version of the template takes precedence over the one found in default.wwwtpl. This means that you don't have to duplicate every template form in default.wwwtpl, just the ones you don't want overwritten by an upgrade.

Note that for list-level templates, site.wwwtpl will itself be overridden by definitions in any listname.wwwtpl files you have installed.

### 9.10.5 National Language Template Files (idiom.mailtpl)

National language templates can be written and used with LISTSERV (L-Soft does not provide them). The use of such templates is optional and governed by two settings:

- **Site-wide** – The `DEFAULT_LANGUAGE=` site configuration variable allows you to set the site-wide national language template for use by all lists on the server. By default this variable is unset and `DEFAULT MAILTPL` (or the corresponding customizations in `SITE MAILTPL`) is used.
- **List-level** – The `Language=` list header keyword can be used to specify a national language template to be used for a particular list, for instance a Spanish-language list on an otherwise English-language server.

To create a national language template, you simply copy `DEFAULT MAILTPL` to *idiom MAILTPL*, where *idiom* is the name of the language, and translate it as desired. You also use *idiom* to specify the value for `DEFAULT_LANGUAGE=` and/or `Language=`. For a given language you can specify anything you want for *idiom*; in other words LISTSERV does not care if you call the file `FRANCAIS MAILTPL` or `FRENCH MAILTPL`, but if you do call it `FRENCH MAILTPL` you must specify `FRENCH` as the *idiom*, and likewise, if you call it `FRANCAIS MAILTPL` you must specify `FRANCAIS` as the *idiom*. LISTSERV has no information about what a given language is called, it simply looks for a `MAILTPL` file for the *idiom* data supplied.

If you are going to translate the web interface template forms into *idiom* as well, you will need to copy `DEFAULT WWWTPL` to *idiom MAILTPL* and again, translate as desired. This requires that you add a special template form to `WWW_ARCHIVE MAILTPL`, so that the 'wa' CGI script will know what to look for, as follows:

```
>>> LANGUAGE
idiom
```

where *idiom* is, of course, the same value we've been talking about above. For instance for a `FRANCAIS` *idiom* you'd use

```
>>> LANGUAGE
FRANCAIS
```

See also Section 9.3.1 [Mail Templates](#) regarding the use of 8-bit characters in template forms.

### 9.10.6 Template Precedence

For template forms found in `DEFAULT MAILTPL`, the following precedence is used when LISTSERV searches for a given template form:

```
listname MAILTPL
idiom MAILTPL
WWW_ARCHIVE MAILTPL
DEFAULT MAILTPL
```

That is to say, if LISTSERV needs a copy of the `ADD1` mail template form, it will look first in the *listname.mailtpl* file for the list in question. If no such file exists, or if `ADD1` is not present in *listname.mailtpl*, LISTSERV will look in *idiom.MAILTPL* (if `Language=` or `DEFAULT_LANGUAGE=` is set to *idiom*). Again, if the `ADD1` form is not present in

idiom.mailtpl, or if idiom.mailtpl does not exist, LISTSERV will then look in default.mailtpl (www\_archive.mailtpl is skipped because ADD1 is not a web template form) and pull out the default ADD1 template form.

For template forms found in DEFAULT WWWTPL the precedence is:

```
listname WWWTPL
idiom WWWTPL
SITE WWWTPL
DEFAULT WWWTPL
```

The same sequence of events applies as for the MAILTPL files, except that SITE WWWTPL is never skipped (all template forms in the WWWTPL files are web forms).

## 9.11 Serving Up Custom Web Pages for your List

*This feature is not available in LISTSERV Lite.*

Originally in order to serve up custom or special web pages for a list it was necessary to construct those pages as HTML files and place them either into the /archives directory or link them from somewhere else. This was sometimes impossible for list owners who had no administrative access to the server's web directories or who had no other place from which to serve web pages.

It is possible to add ad-hoc web page templates by creating new (non-standard) template forms and enabling their display by setting a special variable value, SHOWTPL\_ALLOWED, in the template form. For instance, one could set up a page with special administrative information, the list charter, netiquette information, or the like, and serve it and maintain it directly from the LISTSERV web template interface without need for any other access to the server.

### 9.11.1 A Practical Example: ADMIN\_POST

The author used to serve up an administrative posting via FTP back in the days when his lists lived on a server that had FTP access to the archive notebooks. When FTP access to the server was cut off due to security concerns, he had to find another way to serve the information via the web. Here is how it was done:

(The following example assumes that you have the 1.8e web administration interface installed. New template forms cannot be created this way in previous versions.)

First, log into the web administration interface. Choose the list for which you will be making a new page, and then click the **[Templates]** button to enter the mail and web template editing area. Since the template you will be creating is a web template, click the **[Switch to WWW templates]** button to change modes.

Next, type the name of the new template form into the box provided, and click **[Create]**. The Edit List Template screen opens with the command response

```
The ADMIN_POST form has been successfully stored in the TEST
template library.
```

In the **Description** box, type a description of the template, for example, "Administrative information page".

In the large text box provided for the template text, first type the following line:

```
+SE SHOWTPL_ALLOWED 1
```

This line tells LISTSERV that it is allowed to serve the page on the web. If the line is not found, the template will not be available.

Following this line you can start adding your HTML. However note carefully that you cannot override the default headers and footers that have already been defined by other template forms in the library. You can start with a <title></title> block but it will be followed by the pre-defined header and then by your HTML.

After adding your HTML, click **[Update]**, and the template form will be stored. The URL for the page you are defining in this example will be `http://your_server_hostname/path_to_wa?SHOWTPL=ADMIN_POST&L=listname` (the parameters for 'wa' are case-sensitive and must be sent in upper case). For instance, the author's version of the ADMIN\_POST template form can be viewed at [http://peach.ease.lsoft.com/scripts/wa.exe?SHOWTPL=ADMIN\\_POST&L=VISBAS-L](http://peach.ease.lsoft.com/scripts/wa.exe?SHOWTPL=ADMIN_POST&L=VISBAS-L).

## 9.12 Modifying the Output of LISTSERV's HELP Command (non-VM)

LISTSERV's HELP command output is designed primarily to show the basic syntax of certain commonly used commands, and point users to other documents that explain how to use LISTSERV. Some sites may wish to amplify the message, and this can be done by creating a file called 'localcmd.helpfile' in the same directory where LISTSERV keeps permvars.file, default.mailtpl, and so forth.



**Note:** L-Soft does not recommend removal or alteration of the LSVHELP.FILE that ships with the software (and which contains the default text for the HELP command response) as LISTSERV expects it to be present and it will be overwritten in an upgrade.

LOCALCMD HELPFILE was originally intended to allow non-VM sites to document local commands constructed by using list exits (see the [Advanced Topics Guide for LISTSERV](#) for details regarding list exits and local commands). As such, when LISTSERV sees a LOCALCMD HELPFILE, it appends the contents of LOCALCMD HELPFILE after a line that says "The following local commands are also available:". For instance, if you have a local command called /XYZ, you could have a LOCALCMD HELPFILE containing something like:

```
/XYZ      <options>                (One line description of /XYZ)
```

and the output of HELP would then be

```
> help

LISTSERV(R) version 15.0 - most commonly used commands
INFO      <topic|listname>      Order documentation (plain text files)
SUBscribe listname <full name>  Subscribe to a list
SIGNOFF   listname              Sign off from a list
SIGNOFF   * (NETWIDE)           - from all lists on all servers
Query     listname              Query your subscription options
Search    listname keyword...   Search list archives
SET       listname options      Update your subscription options
INDEX     <listname>            Order a list of LISTSERV files
GET       filename filetype     Order a file from LISTSERV
```



There are more commands; send an INFO REFCARD command for a comprehensive reference card, or just INFO for a list of available documentation files.

If you prefer, you can use LISTSERV through its web interface at <http://listserv.example.com/cgi-bin/wa.exe> (the full manuals can also be browsed online at this URL).

The following local commands are available at this installation:

```
/XYZ      <options>                (One line description of /XYZ)
```

This server is managed by:

```
LSTMAINT@LISTSERV.EXAMPLE.COM
```

For most sites, however, locally-added commands probably won't be available. If you use the LOCALCMD HELPFILE functionality at all, it will likely be to enhance the output in order to make it more understandable for users. So you probably would not want to see the line "The following local commands are available at this installation:" followed by text that doesn't document commands. You can turn off this line by simply adding

```
.NH
```

as the first line of LOCALCMD HELPFILE. Note that `.NH` is the only formatting command available in LOCALCMD HELPFILE; it is otherwise a flat ASCII file that outputs exactly as you type it.

Let's say that you are having a problem where the LISTSERV postmasters are fielding a lot of questions that really ought to be sent to list owners (get me off this list, my address changed, etc.) and a little investigation indicates that these people are getting the postmaster address from the HELP command. It's not reasonable to remove the postmaster's address from the output since it should always be possible to find out who is running the server (in case loops develop, etc.), but you could create a LOCALCMD HELPFILE like the following to indicate to users where various kinds of questions should be sent:

```
.NH
```

For help with a specific list, please write to the list owner(s) at the generic list owner's address. This address takes the form

```
listname-REQUEST@LISTSERV.EXAMPLE.COM
```

For instance, if you are subscribed to a list called DOGLIST-L, to contact the list owners you would write to

```
DOGLIST-L-REQUEST@LISTSERV.EXAMPLE.COM
```

PLEASE DO NOT ACTUALLY WRITE TO DOGLIST-L-REQUEST. This is just an example, you must substitute the name of the mailing list in question. There is no DOGLIST-L mailing list on this server and mail sent to DOGLIST-L-REQUEST is discarded unread.

Please do not write to the server manager unless you do not get a response from the list owner. Thank you!

### 9.13 The \$SITE\$.MAILTPL File



**Note:** Because any customizations made at the site level to the default templates now go into the site-level SITE MAILTPL file, which will not be disturbed during an upgrade, defining site-wide defaults in the old \$SITE\$ MAILTPL file, which in any case did not work for all templates and was never intended for this purpose to begin with, is now obsolete. Most if not all templates that were formerly generated internally by LISTSERV can now be customized as mail templates, message templates, or message fragments, depending on the message.

Sites that are still using \$SITE\$ MAILTPL for this or any other purpose should migrate to the new system. L-Soft support will not field support inquiries concerning \$SITE\$ MAILTPL for LISTSERV 15 and later, but will request that you migrate the templates concerned into the new system.

## Section 10 Interpreting and Managing Log Files

### 10.1 Logs Kept by LISTSERV

LISTSERV keeps logs of all of its activity. On VM systems, this information is kept in the LISTSERV console log. On unix systems, it is kept in `$LSVROOT/listserv.log` by default. Note that unix systems create the log by redirection of standard output to a file; see the 'go' script if you are interested in this process.

On VMS and Windows systems, there may be several different logs depending on the configuration of the system. For instance, in addition to the LISTSERV log itself, there will be logs for the SMTP "workers" if this feature is enabled. On Windows systems, there will also be a log for the SMTP "listener" if it is in service. By default, logs under VMS are kept in `LISTSERV_ROOT: [LOG]` and logs under Windows are kept in `\LISTSERV\LOG`.

### 10.2 Managing the Logs

#### 10.2.1 Making Daily Logs

While LISTSERV for VM (via WAKEPARAM FILE) and Windows "turns" the log at midnight each day, automatically creating daily logs, LISTSERV for VMS and unix does not.

LISTSERV for VMS can "turn" the log via the revision control system if you simply stop and restart LISTSERV once a day (note that you must stop LISTSERV completely and restart it in a new process). This will create files like `LSV_machinename.LOG;1` and `LSV_machinename.LOG;2` in `LISTSERV_ROOT: [LOG]` (by default).

LISTSERV for unix creates a single file in the `$LSVROOT` directory called `listserv.log`. As this file can get quite long, it will probably be most productive to create a shell script called by a cron job to stop LISTSERV, rename the existing `listserv.log` and move it to another location (e.g., `$LSVROOT/oldlogs`), and then restart LISTSERV. L-Soft does not provide shell scripts for this purpose.

#### 10.2.2 Cleaning Your Log Files

On all systems, you will probably want to clean out your old logs on a regular basis. To do this, you will want to write a script that executes automatically (e.g., via WAKEPARAM on VM, as a cron job on unix, or as an AT job on Windows). A sample REXXette for use with Windows that keeps the last 5 days' worth of compressed logs and deletes anything older than that follows:

Figure 10-1 Sample of a CLEANLOG.REXX Script

```

/**/
logdir = 'E:\LISTSERV\LOG'
tempfile = logdir'\CLEANLOG.TMP'
keep = 5

/* First zip all the logs, except today's */
'DIR/B' logdir'\*.LOG >' tempfile
If rc ^= 0 Then Exit

today = Date('S')
Do forever
  line = Translate(Linein(tempfile))
  If line == '' Then Leave
  Parse var line '-date'
  If date = today Then Iterate
  Parse var line fn'.'
  Say 'ZIPping' line'...'
  'ZIP -j -m -q' logdir'\fn logdir'\line
End
Call Lineout tempfile

/* Now delete ZIP files older than 'keep' days */
'DIR/B/O-N' logdir'\*.ZIP >' tempfile
If rc ^= 0 Then Exit

n. = 0
Do forever
  line = Translate(Linein(tempfile))
  If line == '' Then Leave
  Parse upper var line pfx'-date'
  n.pfx = n.pfx + 1
  If n.pfx <= keep Then Iterate
  Say 'Deleting' line'...'
  'DEL' logdir'\line
End
Call Lineout tempfile
'DEL' tempfile

```



**Note:** While it is of course possible to simply delete the log file on a daily basis, for the purpose of debugging potential problems this is not recommended.

### 10.3 Interpreting the LISTSERV log

This file, LISTSERV's main logging file, has different names under different ports of the product.

- VM: LISTSERV logs events to the LISTSERV user's console log.
- OpenVMS: LISTSERV\_ROOT:[LOG]LSV\_machine.LOG;x
- Unix: ~\$LSVROOT/listserv.log
- Windows: LISTSERV\LOG\LISTSERV-yyyymmdd.LOG

Under VM, the console log can be "turned" by placing the appropriate command in the WAKEPARAM file. Under Windows NT and 95, LISTSERV automatically "turns" the log at midnight. For OpenVMS, if you reboot daily as explained in Section 10.2 [Managing the Logs](#), the logs are numbered using the revision tracking system, e.g., LSV\_PEAR.LOG;1, LSV\_PEAR.LOG;2, etc. (if you don't do the daily reboot, LISTSERV just keeps a single log file). For Windows, "yyyymmdd" is the year, month and day of the log, for example, LISTSERV-19980104.LOG is the log for 4 January 1998. Unix logs are simply the output of the program written to standard output and shell scripts (not provided by L-Soft) can be written to "turn" the logs daily via cron.

### 10.3.1 Expiring Cookies

```
25 May 1996 00:00:00 Expiring cookie from xxxxx@ICOM.CA:
> SIGNOFF TECHLINK
25 May 1996 00:00:00 Sent information mail to xxxxx@ICOM.CA
25 May 1996 00:00:00 Expiring cookie from xxxxxxxx@LIGHTSIDE.COM:
> SUBSCRIBE WINNT-L Sxxxx Bxxxxx
25 May 1996 00:00:00 Sent information mail to xxxxxxxx@LIGHTSIDE.COM
```

These entries refer to expiring "OK" confirmation cookies.

### 10.3.2 Releasing and Reallocating a Disk Slot

```
25 May 1996 00:00:00 Virtual disk slot E(E:\LISTSERV\ARCHIVES\NISTEP-L) released
.
25 May 1996 00:00:00 Directory E:\EASE\PEACH\LISTS\IATN accessed as virtual disk
slot E.
```

LISTSERV uses "disk slots" in rotation to minimize the overhead involved in opening a file, performing an operation, closing the file, and then possibly having to reopen the file immediately to perform another operation. A given "disk slot" stays open until it is needed for another file.

### 10.3.3 Reindexing a List

```
25 May 1996 00:00:01 Reindexing MYLIST-L LOG9605D...
```

LISTSERV rebuilds the index for the current notebook archive file of a given list immediately prior to sending the DIGEST and INDEX versions of the list.

### 10.3.4 Distributing a Digest

Here is a typical log sequence for the distribution of a daily digest:

```
25 May 1996 00:00:35 Virtual disk slot E(E:\LISTSERV\ARCHIVES\HONYAKU) released.
25 May 1996 00:00:35 Directory E:\LISTSERV\ARCHIVES\SPAM-L accessed as virtual
disk slot E.
25 May 1996 00:00:37 Reindexing SPAM-L LOG9605...
25 May 1996 00:00:37 SPAM-L digest is being distributed...
25 May 1996 00:00:37 Distributing mail ("SPAM-L") from owner-SPAM-
L@PEACH.EASE.LSOFT.COM...
25 May 1996 00:00:38 Mail forwarded to H-NET.MSU.EDU for 2 recipients.
25 May 1996 00:00:38 Mail forwarded to UAFSYSB.UARK.EDU for 2 recipients.
25 May 1996 00:00:38 Mail forwarded to LISTMAIL.SUNET.SE for 2 recipients.
```

The preceding 3 jobs were forwarded to LISTSERV servers on the DISTRIBUTE backbone. The following mail is posted to 45 users who are not served by the DISTRIBUTE backbone in a single BSMTP "envelope".

```

25 May 1996 00:00:38 Mail posted via SMTP to xxx@AMERICAN.EDU.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxxxx@AOL.COM.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxxxx@AOL.COM.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxx@BCVMS.BC.EDU.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxx@BMACADM.BITNET.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxx@CLEMSON.EDU.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxxxxxxxx@COMPUSERVE.COM.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxxxx@EMAIL.GC.CUNY.EDU.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxxxx@ERE.UMONTREAL.CA.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxxx@ETERNA.COM.AU.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxxx@GOL.COM.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxxxx@GPU.SRV.UALBERTA.CA.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxxxx@HAWAII.EDU.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxxx@IBM.NET.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxxxx@IDIR.NET.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxxxx@INET.UNI-C.DK.
25 May 1996 00:00:38 Mail posted via SMTP to xxx@KSUVM.KSU.EDU.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxx@LOC.GOV.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxx@LOONY-TOONS.TAMU.EDU.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxxx@LTRR.ARIZONA.EDU.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxxxx@MAILBOX.SYR.EDU.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxxxx@MO.NET.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxx@MUSICA.MCGILL.CA.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxxxx@NANDO.NET.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxxxx@NETCOM.COM.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxxxx@NYIQ.NET.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxx@PAMELA.INT.MED.UNIPI.IT.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxxx@PIONEER.STATE.ND.US.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxxx@PSC.LSA.UMICH.EDU.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxx@PSUVM.PSU.EDU.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxxx@PUCC.PRINCETON.EDU.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxxx@SCS.UNR.EDU.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxxx@SILVERPLATTER.COM.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxxx@SJVVM.STJOHNS.EDU.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxx@SPCVXA.SPC.EDU.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxx@TELERAMA.LM.COM.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxx@UA1VM.UA.EDU.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxxxx@UABDPO.DPO.UAB.EDU.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxxxx@UCONNVM.UCONN.EDU.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxxxx@UTARLV1.UTA.EDU.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxxxx@UVVM.UVIC.CA.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxxxx@VM1.HQADMIN.DOE.GOV.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxx@VM1.MCGILL.CA.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxxx@VM3090.EGE.EDU.TR.
25 May 1996 00:00:38 Mail posted via SMTP to xxxxxxx@WATSON.IBM.COM.

```

LISTSERV always summarizes and tells you what it's done:

```
25 May 1996 00:00:38 Done - 3 jobs (6 rcpts), 1 outbound file (45 rcpts).
```

### 10.3.5 Daily Error Monitoring Reports

```
25 May 1996 00:00:43 Generating daily nondelivery monitoring reports...
```

Here LISTSERV deletes a subscriber who has gone over the Auto-Delete= limits:

25 May 1996 00:00:44 -> Deleted xxxxxxxx@CARIARI.UCR.AC.CR from ACCESS-L.

LISTSERV now sends out the daily error monitoring reports to the appropriate people (defined in "Errors-To=" for each list).

25 May 1996 00:00:44 Sent information mail to xxxxxxxx@CARIARI.UCR.AC.CR  
 25 May 1996 00:00:45 Sent information mail to xxxxxxx@LINUS.DC.LSOFT.COM  
 25 May 1996 00:00:46 Sent information mail to xxxxxx@AF.PENTAGON.MIL  
 25 May 1996 00:00:46 Sent information mail to xxxxxx@ESA.MHS.COMPUSERVE.COM

As LISERVER continues to run and process errors for the various lists, it will update the listname.AUTODEL file whenever it receives an error that it understands:

25 May 1996 00:05:43 Automatic nondelivery report processing for WIN95-L:  
 25 May 1996 00:05:43 -> All errors temporary, no action taken.  
 25 May 1996 00:07:34 Automatic nondelivery report processing for EXCEL-G:  
 25 May 1996 00:07:34 -> 1 monitoring entry updated.

### 10.3.6 Processing Mail for Local Lists

25 May 1996 00:39:23 Processing file 8209289 from MAILER@PEACH.EASE.LSOFT.COM  
 25 May 1996 00:39:25 Processing mail from xxxxxx@PRIMENET.COM for ACCESS-L  
 25 May 1996 00:39:25 Virtual disk slot E(E:\EASE\PEACH\FTP\EXCEL-L) released.  
 25 May 1996 00:39:25 Directory E:\EASE\PEACH\FTP\ACCESS-L accessed as virtual disk slot E.  
 25 May 1996 00:39:26 Distributing mail ("ACCESS-L") from owner-access-1@PEACH.EASE.LSOFT.COM...  
 25 May 1996 00:39:26 Mail posted via SMTP to xxxxxxxxxxxxxxxx@NR-COMMS.RADIO.BBC.CO.UK.  
 25 May 1996 00:39:26 Mail posted via SMTP to xxxxxxxx@OHS.ORG.  
 25 May 1996 00:39:26 Mail posted via SMTP to xxxxxx@POBOX.COM.  
 25 May 1996 00:39:26 Done - 1 outbound file (3 rcpts).  
 25 May 1996 00:39:26 Distributing mail ("ACCESS-L") from owner-access-1@PEACH.EASE.LSOFT.COM...  
 25 May 1996 00:39:26 Mail forwarded to LISERVER@HEARN for 4 recipients.  
 25 May 1996 00:39:27 Mail forwarded to LISTMAIL.SUNET.SE for 11 recipients.  
 25 May 1996 00:39:27 Mail forwarded to LISERVER@ICINECA for 4 recipients.  
 25 May 1996 00:39:27 Mail forwarded to LISERVER.GMD.DE for 3 recipients.  
 25 May 1996 00:39:27 Mail forwarded to LISERVER@AEARN for 2 recipients.  
 25 May 1996 00:39:27 Processed 192 recipients...  
 25 May 1996 00:39:27 Done - 5 jobs (24 rcpts), 1 outbound file (192 rcpts).  
 25 May 1996 00:39:27 Distributing mail ("ACCESS-L") from owner-access1@PEACH.EASE.LSOFT.COM...  
 25 May 1996 00:39:27 Mail posted via SMTP to xxxxxx@ADC.COM.  
 25 May 1996 00:39:27 Mail posted via SMTP to xxxxxxxx@MSMEL.PRAXA.COM.AU.  
 25 May 1996 00:39:27 Mail posted via SMTP to xxxxxxxx@SPRYNET.COM.  
 25 May 1996 00:39:27 Done - 1 outbound file (3 rcpts).  
 25 May 1996 00:39:27 Message DISTRIBUTEd to 222 recipients.  
 25 May 1996 00:39:28 Sent information mail to xxxxxx@PRIMENET.COM

### 10.3.7 Administrative Mail (X-ADMMAIL)

25 May 1996 00:01:16 From MAILER@PEACH.EASE.LSOFT.COM: X-ADMMAIL OWNER-AFNS  
 25 May 1996 00:01:16 Processing file 8206153 from MAILER@PEACH.EASE.LSOFT.COM  
 6 Jun 2000 09:12:42 Automatic nondelivery report processing for AFNS:  
 6 Jun 2000 09:12:42 -> All errors temporary, no action taken.

This particular type of mail, when sent to the OWNER-listname address, is a delivery error being returned to the RFC 821 MAIL FROM: address. The last two lines of this log



excerpt entry simply indicate that auto-deletion is enabled for the list and that in this case the error received was not a permanent one.

```
26 Jun 2000 09:12:38 From MAILER@PEACH.EASE.LSOFT.COM: X-ADMMAIL TEST-REQUEST
JOE.USER@EXAMPLE.COM
```

```
26 Jun 2000 09:12:39 Sent information mail to JOE.USER@EXAMPLE.COM
```

When sent to the listname-REQUEST address, the mail is forwarded to all non-quiet list owners and the REQACK1 template form (see Section 9 [Creating and Editing Mail and Web Templates](#)) is sent back to the user who wrote to the listname-REQUEST address.

### 10.3.8 DISTRIBUTE Jobs from Remote Hosts

Just as our LISTSERV sends out DISTRIBUTE jobs to the backbone, it also receives them for distribution from remote backbone hosts.

```
25 May 1996 00:01:16 Processing file 8206155 from
MAILER@PEACH.EASE.LSOFT.COM
```

```
25 May 1996 00:01:16 From LISTSERV@PSUVM.PSU.EDU: X-B64 ID=PANL.MAILDIST
CLASS=A
```

```
25 May 1996 00:01:16 Rescheduled as: 8206156
```

```
25 May 1996 00:01:16 Processing file 8206156 from
LISTSERV@PEACH.EASE.LSOFT.COM
```

```
25 May 1996 00:01:16 From LISTSERV@PSUVM: DIST2 MAIL I=Y FROM=owner-pan-
1@BINGVMB.CC.BINGHAMTON.EDU FORW(CRC) HOST(626 626 (...))
```

```
25 May 1996 00:01:17 Distributing mail ("PAN-L") from owner-pan-
1@BINGVMB.CC.BINGHAMTON.EDU...
```

```
25 May 1996 00:01:17 Mail posted via SMTP to xxxxxxxx@ACS.RYERSON.CA.
```

```
25 May 1996 00:01:17 Mail posted via SMTP to xxxxxx@AMAIL.AMDAHL.COM.
```

```
25 May 1996 00:01:17 Mail posted via SMTP to xxxxxxxx@AOL.COM.
```

and so forth.

### 10.3.9 Requesting "OK" Confirmation for Commands

```
25 May 1996 00:01:17 Processing file 8206160 from MAILER@PEACH.EASE.LSOFT.COM
```

```
25 May 1996 00:01:17 From xxxxxxxx@AUSCONSULT.COM.AU: UNSUB EXCEL-L
```

```
25 May 1996 00:01:17 Requesting confirmation, cookie=3EB4F2
```

```
25 May 1996 00:01:17 Sent information mail to xxxxxxxx@AUSCONSULT.COM.AU
```

### 10.3.10 Subscription Summary Updates (SUPD Jobs)

X-SUPD jobs update the file used by "LIST SUMMARY". LISTSERV receives the file from another LISTSERV host and distributes it down the line:

```
25 May 1996 00:04:54 Processing file 8206401 from MAILER@PEACH.EASE.LSOFT.COM
```

```
25 May 1996 00:04:54 From LISTSERV@INTERNET.COM: X-B64 ID=X-SUPD.JOB ASCII
CLASS=J
```

```
25 May 1996 00:04:54 Rescheduled as: 8206402
```

```
25 May 1996 00:04:54 Processing file 8206402 from LISTSERV@PEACH.EASE.LSOFT.COM
```

```
25 May 1996 00:04:54 From LISTSERV@INTERNET.COM: DIST2 I=Y
FROM=LISTSERV@INTERNET.COM FORW(CRC) HOST(626 626 626 626 626 626 691 686 (...))
```

```
25 May 1996 00:04:54 Distributing file "X-SUPD JOB" from LISTSERV@INTERNET.COM...
```

```
25 May 1996 00:04:54 File forwarded to LIME.EASE.LSOFT.COM for 1 recipient.
```

```
25 May 1996 00:04:54 File forwarded to WIN95.DC.LSOFT.COM for 1 recipient.
```

```
25 May 1996 00:04:54 File forwarded to SPIDER.EASE.LSOFT.COM for 1 recipient.
```

```
25 May 1996 00:04:54 File forwarded to HOME.EASE.LSOFT.COM for 1 recipient.
```

```

25 May 1996 00:04:54 File forwarded to LISTSERV.GEORGETOWN.EDU for 1 recipient.
25 May 1996 00:04:54 File forwarded to CC1.KULEUVEN.AC.BE for 1 recipient.
25 May 1996 00:04:54 File forwarded to LISTSERV.USHMM.ORG for 1 recipient.
25 May 1996 00:04:54 File forwarded to LISTSERV.CLARK.NET for 1 recipient.
25 May 1996 00:04:54 File "X-SUPD JOB" distributed to
LISTSERV@PEACH.EASE.LSOFT.COM.
25 May 1996 00:04:54 Done - 8 jobs (8 rcpts), 1 outbound file (1 rcpt).
25 May 1996 00:04:54 Processing file 8206411 from LISTSERV@PEACH.EASE.LSOFT.COM
25 May 1996 00:04:54 From LISTSERV@INTERNET.COM: X-SUPD FWD=NO
DATE=1996052500:00:04 DATA=5 56 PHOTOPRO 465 PHOTOTECH 268 PHOTOAS (...)

```

### 10.3.11 Global List of Lists Updates (LUPD Jobs)

X-LUPD jobs update the list of lists. LISTSERV receives the file from another LISTSERV host and distributes it down the line:

```

25 May 1996 00:04:08 Processing file 8206361 from MAILER@PEACH.EASE.LSOFT.COM
25 May 1996 00:04:08 From LISTSERV@LISTSERV.UIC.EDU: X-B64 ID=X-LUPD.JOB ASCII
CLASS=J
25 May 1996 00:04:08 Rescheduled as: 8206362
25 May 1996 00:04:08 Processing file 8206362 from LISTSERV@PEACH.EASE.LSOFT.COM
25 May 1996 00:04:08 From LISTSERV@LISTSERV.UIC.EDU: DIST2 I=Y FROM=LISTSERV@LIS
TSERV.UIC.EDU FORW(CRC) HOST(626 626 626 626 626 691 (...))
25 May 1996 00:04:08 Distributing file "X-LUPD JOB" from
LISTSERV@LISTSERV.UIC.EDU...
25 May 1996 00:04:08 File forwarded to LIME.EASE.LSOFT.COM for 1 recipient.
25 May 1996 00:04:08 File forwarded to WIN95.DC.LSOFT.COM for 1 recipient.
25 May 1996 00:04:08 File forwarded to SPIDER.EASE.LSOFT.COM for 1 recipient.
25 May 1996 00:04:08 File forwarded to HOME.EASE.LSOFT.COM for 1 recipient.
25 May 1996 00:04:08 File forwarded to LISTSERV.GEORGETOWN.EDU for 1 recipient.
25 May 1996 00:04:08 File forwarded to CC1.KULEUVEN.AC.BE for 1 recipient.
25 May 1996 00:04:08 File forwarded to LISTSERV.USHMM.ORG for 1 recipient.
25 May 1996 00:04:08 File forwarded to LISTSERV.CLARK.NET for 1 recipient.

```

LISTSERV also sends itself a copy:

```

25 May 1996 00:04:08 File "X-LUPD JOB" distributed to
LISTSERV@PEACH.EASE.LSOFT.COM.
25 May 1996 00:04:08 Done - 8 jobs (8 rcpts), 1 outbound file (1 rcpt).
25 May 1996 00:04:08 Processing file 8206371 from LISTSERV@PEACH.EASE.LSOFT.COM
25 May 1996 00:04:08 From LISTSERV@LISTSERV.UIC.EDU: X-LUPD FWD=NO
DATE=1996052423:00:02 HDR=YES

```

The following entry tells LISTSERV to replace the information it has for NEWIO-L in its global list of lists:

```

> REP NEWIO-L NEWIO-L /Info about replacing ILLINET Online hardware and
software/2616677089
> CKS 3881006631
25 May 1996 00:04:10 GLOBLIST FILE has been successfully updated.

```

The entries for deleted lists also are updated (deleted from GLOBLIST FILE):

```

25 May 1996 00:24:11 Processing file 8207550 from
LISTSERV@PEACH.EASE.LSOFT.COM
25 May 1996 00:24:11 From LISTSERV@VM1.NODAK.EDU: X-LUPD FWD=NO
DATE=1996052423:00:04 HDR=YES
> DEL WLREHAB
> DEL WDAMAGE
> DEL POWER-L

```

```

> DEL QUEST
> DEL RS1-L
> DEL SANGEET
> DEL SPRINT-L
> DEL STAFFGOV
> DEL TAG-L
> DEL TELUGU
> DEL THEORY-A
> DEL THEORY-B
> DEL THEORY-C
> DEL THEORYNT
> DEL TOW
> DEL TWSGIS-L
> DEL UND-SEMI
> CKS 3794276653
25 May 1996 00:24:13 GLOBLIST FILE has been successfully updated.

```



**Note:** If the "CKS" checksum does not check out, the job is discarded without being processed.

### 10.3.12 Valid "OK" Confirmation Received

Here is a set of typical log entries for the receipt of a valid "OK" confirmation. Notice that LISTSERV accepts the OK and then issues itself the command that required confirmation. Other than the "OK", this behavior (at least in the log) is identical to how LISTSERV handles commands that do not require confirmation.

```

225 May 1996 00:04:58 Processing file 8206418 from
MAILER@PEACH.EASE.LSOFT.COM
25 May 1996 00:04:58 From xxxxxxxxxxx@SOL.KISS.DE: OK
25 May 1996 00:04:58 From xxxxxxxxxxx@SOL.KISS.DE: SIGNOFF AFWEEKLY
25 May 1996 00:04:58 To xxxxxxxxxxx@SOL.KISS.DE: You have been removed
from the AFWEEKLY list.
25 May 1996 00:04:58 Sending FAREWELL message to xxxxxxxxxxx@SOL.KISS.DE
25 May 1996 00:04:58 Sent information mail to xxxxxxxxxxx@SOL.KISS.DE
25 May 1996 00:04:58 Sent information mail to:
    > xxxxxxx@AFSYNC.HQ.AF.MIL xxxxxxx@AFSYNC.HQ.AF.MIL
    > xxxxxx@AFNEWS.PA.AF.MIL xxxxxxx@MASTER.PA.AF.MIL
25 May 1996 00:04:58 Sent information mail to xxxxxxxxxxx@SOL.KISS.DE

```

The "OK" may contain the return cookie:

```

25 May 1996 00:08:50 Processing file 8206772 from MAILER@PEACH.EASE.LSOFT.COM
25 May 1996 00:08:50 From xxxxxxxxxxx@GENIE.COM: OK 71365E
25 May 1996 00:08:50 From xxxxxxxxxxx@GENIE.COM: SUBSCRIBE AFNS Mxxxx Exxx
Kxxxxxer
25 May 1996 00:08:51 To xxxxxxxxxxx@GENIE.COM: You have been added to the AFNS
list.
25 May 1996 00:08:51 Sent information mail to xxxxxxxxxxx@GENIE.COM
25 May 1996 00:08:51 Sending WELCOME message to xxxxxxxxxxx@GENIE.COM
25 May 1996 00:08:51 Sent information mail to xxxxxxxxxxx@GENIE.COM
25 May 1996 00:08:51 Sent information mail to:
    > xxxxxxx@AFSYNC.HQ.AF.MIL xxxxxxx@AFSYNC.HQ.AF.MIL
    > xxxxxx@AFNEWS.PA.AF.MIL xxxxxxx@MASTER.PA.AF.MIL
25 May 1996 00:08:51 Sent information mail to xxxxxxxxxxx@GENIE.COM

```

### 10.3.13 Invalid "OK" Confirmation Received

"OK" confirmation codes relate to specific userids. For instance, if you try to execute a command as "someuser@someplace.com" and reply to the "OK" from "someuser@unix1.someplace.com", LISTSERV will not perform so-called "fuzzy matching" or do a DNS lookup to determine whether or not "unix1.someplace.com" maps to "someplace.com". Therefore, since the code and the userid don't match, LISTSERV will respond that the confirmation code does not match any pending job.

This message is also sent if the "OK" comes after the "cookie" expires, since of course there is no longer any pending job matching it.

```
25 May 1996 01:16:28 Processing file 8211043 from MAILER@PEACH.EASE.LSOFT.COM
25 May 1996 01:16:29 From xxxxxxx@MSN.COM: ok
25 May 1996 01:16:29 To xxxxxxx@MSN.COM: The confirmation code you gave (78B484)
does not correspond to any pending (...)
```

### 10.3.14 User Already Subscribed to a Given List

The user may be trying to change his "real name" field in the list. In any case, if the "real name" field matches the one in the SUBSCRIBE command, the following is logged:

```
25 May 1996 00:11:42 Processing file 8206935 from MAILER@PEACH.EASE.LSOFT.COM
25 May 1996 00:11:42 From xxxxxxxx@NS.NET: SUBSCRIBE IN-TOUCH Txxxxx Hxxxxxx
25 May 1996 00:11:42 To xxxxxxxx@NS.NET: You are already subscribed to the IN-
TOUCH list as "Txxxxx Hxxxxxx".
25 May 1996 00:11:42 Sent information mail to therrera@NS.NET
```

If the "real name" field is different, the following is logged:

```
25 May 1996 00:16:18 Processing file 8207278 from
MAILER@PEACH.EASE.LSOFT.COM
25 May 1996 00:16:18 From xxxxxxxx@EROLS.COM: SUBSCRIBE IN-TOUCH Rxxxxxx
Sxxx
25 May 1996 00:16:18 To xxxxxxxx@EROLS.COM: The name associated with
your xxxxxxxx@EROLS.COM subscription has been (...)
25 May 1996 00:16:18 Sent information mail to xxxxxxxx@EROLS.COM
```

### 10.3.15 Non-Command Text in Mailings to LISTSERV

```
25 May 1996 00:32:07 Processing file 8207703 from MAILER@PEACH.EASE.LSOFT.COM
25 May 1996 00:32:07 From MAILER@PEACH.EASE.LSOFT.COM: X-ADMMAIL OWNER-EXCEL-L
Mailer-Daemon@inf.com
25 May 1996 00:32:15 Processing file 8207705 from MAILER@PEACH.EASE.LSOFT.COM
25 May 1996 00:32:15 From xxxxxxxx@IX.NETCOM.COM: ok
25 May 1996 00:32:15 From xxxxxxxx@IX.NETCOM.COM: sub WINNT-L Txxxx D.Sxxxx
25 May 1996 00:32:15 To xxxxxxxx@IX.NETCOM.COM: You have been added to the
WINNT-L list.
25 May 1996 00:32:15 Sent information mail to xxxxxxxx@IX.NETCOM.COM
25 May 1996 00:32:15 From xxxxxxxx@IX.NETCOM.COM: Txxxx Sxxxx
25 May 1996 00:32:15 To xxxxxxxx@IX.NETCOM.COM: Unknown command - "TXXXX". Try
HELP.
25 May 1996 00:32:15 From xxxxxxxx@IX.NETCOM.COM: xxxxxxxx@ix.netcom.com
25 May 1996 00:32:15 To xxxxxxxx@IX.NETCOM.COM: Unknown command - "XXXXXXXX@IX
.NETCOM.COM". Try HELP.
25 May 1996 00:32:15 From xxxxxxxx@IX.NETCOM.COM: http://www.netcom.com/
~xxxxxxxxx/
25 May 1996 00:32:15 To xxxxxxxx@IX.NETCOM.COM: Unknown command - "HTTP:". Try
```



### 10.3.20 FIOC Cache Notifications

LISTSERV caches files that it uses for efficiency. Occasionally, you may see a warning that the FIO cache has reached a preset limit (FIOC\_WARNING in the site configuration file). See the FIOC\_CACHE, FIOC\_TRIM, and FIOC\_WARNING site configuration variables in the [Site Configuration Keyword Reference](#) document for more information. If you get a lot of these warnings, you may want to consider adjusting the cache values.

```
25 May 1996 01:24:06 Virtual disk slot E(E:\EASE\PEACH\FTP\VBDATA-L)
released.
```

```
25 May 1996 01:24:06 Directory E:\LISTSERV\ARCHIVES\SPAM-L accessed as
virtual disk slot E.
```

```
*** FIO file cache now totals 20659k. A list of cached files follows. ***
```

File size	Usage	Flags	File name
-----	-----	-----	-----
5071k	1	U	E:\LISTSERV\ARCHIVES\SPAM-L\SPAM-L.LOG9605
2k	1		E:\listserv\TMP\LISTSERV.CMSUT1
242k	1		E:\listserv\main\PERMVARS.FILE
4k	1		E:\listserv\main\VBDATA-L.DIGEST
378k	1		E:\EASE\PEACH\FTP\VBDATA-L\VBDATA-L.LOG9605D
121k	1		E:\listserv\main\POSTNOTE.LIST
282k	1		E:\listserv\main\FUTURESUPERSTOCK.LIST
115k	1		E:\listserv\main\AUTOTECHNET.LIST
6k	1		E:\LISTSERV\ARCHIVES\HONYAKU\HONYAKU.DIGEST
1k	3		E:\listserv\main\DIGESTS.FILE
11k	3		E:\listserv\TMP\TEMP.FILELIST

(Many more lines were deleted)

### 10.3.21 Web Archive/Administration Interface Logging

When LISTSERV receives a request from the 'wa' interface, it logs the activity as below. Note that LISTSERV receives an X-LOGIN command from 'wa' and issues a validation code if the password and the user's e-mail address match. Note also that LISTSERV logs the IP address of the machine making the request via 'wa'.

```
5 Aug 1997 10:51:08 From IUSR_XXX@PEACH.EASE.LSOFT.COM: X-LOGIN
xxxxxx@lsoft.com 206.241.13.58 PW=YYYYYYYYY
5 Aug 1997 10:51:08 To IUSR_XXX@PEACH.EASE.LSOFT.COM: ***OK***
6093C01B81CF68D42B
5 Aug 1997 10:51:09 From IUSR_XXX@PEACH.EASE.LSOFT.COM: X-LOGCK
6093C01B81CF68D42B AUTHINFO(206.241.13.58) WM: LIST OWNED
5 Aug 1997 10:51:21 From IUSR_XXX@PEACH.EASE.LSOFT.COM: X-LOGCK
6093C01B81CF68D42B AUTHINFO(206.241.13.58) OWNER(TEST) WM: GET TEST (HDR
NOL (...))
```

The following indicates a timeout after 60 seconds of inactivity:

```
4 Feb 1998 10:26:53 From IUSR_XXX@PEACH.EASE.LSOFT.COM: X-LOGCK
37BA2700C7AA3AE9EE AUTHINFO(208.141.38.1) TIMEKILL(60)
```

This indicates a web archive search and the response:

```
4 Feb 1998 10:26:53 From IUSR_XXX@PEACH.EASE.LSOFT.COM: X-LOGCK
3EA2D501187014BB04 AUTHINFO(204.149.110.125) NOTEBOOK(spam-1) DBS:
SEARC (...)
4 Feb 1998 10:26:56 Reindexing SPAM-L LOG9802A...
4 Feb 1998 10:27:08 To IUSR_PEACH@LSOFT.COM: -> 23 matches.
```

This indicates a subscription via the web interface:

```
4 Feb 1998 10:27:08 From IUSR_PEACH@LSOFT.COM: X-LOGCK
37BA2700C7AA3AE9EE AUTHINFO(208.141.38.1) WM: SUBSCRIBE WINNT-L Kevin Mc
(...)
4 Feb 1998 10:27:08 Requesting confirmation, cookie=1723C284
```

### 10.3.22 X-SPAM Jobs

From time to time a server receives X-SPAM jobs from other LISTSERV hosts. These jobs are "spam alerts" which tell the server that spam has been detected at another site and a user has been put under 48 hour quarantine.

```
14 Jun 1999 06:48:52 Processing file 41256617 from
MAILER@PEACH.EASE.LSOFT.COM
14 Jun 1999 06:48:52 From LISTSERV@PLUM.EASE.LSOFT.COM: X-B64 ID=X-
SPAM.JOB ASCII CLASS=J
14 Jun 1999 06:48:52 Rescheduled as: 41256618
14 Jun 1999 06:48:52 Processing file 41256618 from
LISTSERV@PEACH.EASE.LSOFT.COM
14 Jun 1999 06:48:52 From LISTSERV@PLUM.EASE.LSOFT.COM: DIST2 I=Y
FROM=LISTSERV@LISTSERV.AOL.COM FORW(CRC) HOST(626)
14 Jun 1999 06:48:52 Distributing file "X-SPAM JOB" from
LISTSERV@LISTSERV.AOL.COM...
14 Jun 1999 06:48:52 File "X-SPAM JOB" distributed to
LISTSERV@PEACH.EASE.LSOFT.COM.
14 Jun 1999 06:48:52 Done - 1 outbound file (1 rcpt).
14 Jun 1999 06:48:52 Processing file 41256619 from
LISTSERV@PEACH.EASE.LSOFT.COM
14 Jun 1999 06:48:52 From LISTSERV@LISTSERV.AOL.COM: X-SPAM
exxxx_XXXXXXXXXX@YAHOO.COM 4214CE9E
14 Jun 1999 06:48:52 -> Registered.
```

At the end of this, the address `exxxx_XXXXXXXXXX@YAHOO.COM` has been registered as a spammer and will be quarantined for 48 hours.

### 10.3.23 X-TBREG Jobs

X-TBREG jobs are sent out by all LISTSERV Lite Free Edition servers and any other servers that are set to TABLELESS runmode (see "RUNMODE" in the [Site Configuration Keyword Reference](#) document). These jobs register your server with a central L-Soft server, and are important for two reasons: first, so that your server and lists can show up in the L-Soft-maintained Catalist service; and second, so that your server can participate correctly in the LISTSERV distributed server model without needing to update LISTSERV's networking tables on a regular basis.



Non-Free Edition servers set to NETWORKED or STANDALONE runmode do not generate X-TBREG jobs, although they may receive them from remote hosts to be cached for DISTRIBUTE purposes.

For more information about server registration, see Section 5.6 [Server Registration](#).



**Note:** LISTSERV Lite Free Edition servers cannot change their runmode and therefore will always self-register this way.

### 10.3.24 Responses to LVMON@VM.SE.LSOFT.COM

If you are running in Networked or Tableless mode you may see these from time to time:

```
13 Mar 2000 16:45:13 From LVMON@VM.SE.LSOFT.COM: RELEASE
13 Mar 2000 16:45:13 To   LVMON@VM.SE.LSOFT.COM: LISTSERV(R) High
Performance for Windows NT version 1.8d, managed by:
13 Mar 2000 16:45:13 From LVMON@VM.SE.LSOFT.COM: SHOW
13 Mar 2000 16:45:13 From LVMON@VM.SE.LSOFT.COM: SHOW WWW_ARCHIVE_URL
13 Mar 2000 16:45:13 To   LVMON@VM.SE.LSOFT.COM: WWW_ARCHIVE_URL = http://
/peach.ease.lsoft.com/scripts/wa.exe
13 Mar 2000 16:45:13 From LVMON@VM.SE.LSOFT.COM: SHOW CTR 200003
13 Mar 2000 16:45:13 From LVMON@VM.SE.LSOFT.COM: SHOW CTR 200002
13 Mar 2000 16:45:13 Sent information mail to LVMON@VM.SE.LSOFT.COM
```

VM.SE.LSOFT.COM is a central L-Soft server that collects publicly-available statistics and other information for the CataList service and for L-Soft's use in developing usage metrics. All of the commands sent by LVMON are documented and can be issued by any user. See also 5.7 of this manual for more information on inter-server information sharing.

### 10.3.25 MIME Parser Messages

In 1.8e, LISTSERV's MIME parser was completely rewritten to address a number of issues, not the least of which were to add uuencoded binary filtering support and anti-virus support. When a MIME message is received by LISTSERV, information about the makeup of the message is logged; for example, here is a message with an RTF file attached which has been sent to a list:

```
9 Nov 2001 10:26:52 Processing file 0118 from MAILER@LISTSERV.EXAMPLE.COM
Part 1 [1-243]: MULTIPART/MIXED; parent=0; boundary="-----_480
189806==_"
Part 2 [1-6]: TEXT/PLAIN; parent=1; boundary=""
Part 3 [7-235]: APPLICATION/RTF; parent=1; boundary=""
Parts: 3 skipped: 0
```

And here is a similar message posted to LISTSERV itself with a command in the plain text part and an RTF attachment:

```
9 Nov 2001 10:31:44 Processing file 0121 from MAILER@LISTSERV.EXAMPLE.COM
Part 1 [1-235]: MULTIPART/MIXED; parent=0; boundary="-----_480
480784==_"
Part 2 [1-5]: TEXT/PLAIN; parent=1; boundary=""
Part 3 [6-234]: APPLICATION/RTF; parent=1; boundary=""
Parts: 3 skipped: 0
Rewriting part 2
9 Nov 2001 10:31:44 -> Decoding MIME message parts...
9 Nov 2001 10:31:44 Rescheduled as: 0122
```

```

9 Nov 2001 10:31:44 Processing file 0122 from LISTSERV@LISTSERV.EXAMPLE.COM
Part 1 [1-1]: TEXT/PLAIN; parent=0; boundary=""
Parts: 1 skipped: 0
9 Nov 2001 10:31:44 From ncb@EXAMPLE.COM: show lic
9 Nov 2001 10:31:44 Sent information mail to ncb@EXAMPLE.COM

```

In this latter message, LISTSERV sees the three MIME parts of the message and skips them all, passing the message to the MIME parser. LISTSERV then rewrites the second MIME part (the TEXT/PLAIN attachment that actually contains the command(s) to be executed) and places a new job containing only the TEXT/PLAIN attachment in the spool, which is then executed by the command processor. The APPLICATION/RTF attachment is simply discarded.

Finally, here is an example of a subscription request received from a site running an InterScan VirusWall gateway, which wraps the original message in a special MIME wrapper. The actual "raw" message as received by LISTSERV looks like this:

```

X-MimeOLE: Produced By Microsoft Exchange V6.0.5762.3
content-class: urn:content-classes:message
MIME-Version: 1.0
Content-Type: multipart/mixed;
                boundary="-----InterScan_NT_MIME_Boundary"
Subject:
Date: Mon, 26 Nov 2001 14:51:18 -0800
Message-ID: <66C896FC5FA3EB4E9BC85569962F378403BCC430@example.com>
X-MS-Has-Attach:
X-MS-TNEF-Correlator:
Thread-Index: AcF2zNx1xlGA3wa1RES/vvt+eFWNGw==
From: "Joe User" <joe@example.com>
To: <LISTSERV@LISTSERV.EXAMPLE.COM>
Return-Path: joe@example.com
X-OriginalArrivalTime: 26 Nov 2001 22:51:19.0067 (UTC)
FILETIME=[DCAAC6B0:01C176CC]

This is a multi-part message in MIME format.
-----InterScan_NT_MIME_Boundary
Content-Type: multipart/alternative;
                boundary="----=_NextPart_001_01C176CC.DC8A9762"
-----=_NextPart_001_01C176CC.DC8A9762
Content-Type: text/plain;
                charset="us-ascii"
Content-Transfer-Encoding: quoted-printable
SUBSCRIBE TEST Joe User

-----=_NextPart_001_01C176CC.DC8A9762
Content-Type: text/html;
                charset="us-ascii"
Content-Transfer-Encoding: quoted-printable
SUBSCRIBE TEST Joe = User
=00
-----=_NextPart_001_01C176CC.DC8A9762--
-----InterScan_NT_MIME_Boundary--

```

Versions of LISTSERV previous to 1.8e were unable to parse any text out of this type of message, but from version 1.8e forward the MIME parser is able to find the plain text attachment and execute it:

```
27 Nov 2001 10:13:01 Processing file 0083 from MAILER@LISTSERV.EXAMPLE.COM
Part 1 [1-24]: MULTIPART/MIXED; parent=0; boundary="-----
InterScan_NT_MIME_Boundary"
Part 2 [3-0]: MULTIPART/ALTERNATIVE; parent=1; boundary="-----
_NextPart_001_01C176CC.DC8A9762"
Part 3 [7-14]: TEXT/PLAIN; parent=2; boundary=""
Part 4 [15-21]: TEXT/HTML; parent=2; boundary=""
Parts: 4 skipped: 0
Rewriting part 3
27 Nov 2001 10:13:01 -> Decoding MIME message parts...
27 Nov 2001 10:13:01 Rescheduled as: 0084
XMV.finalrc: 3
27 Nov 2001 10:13:01 Processing file 0084 from LISTSERV@LISTSERV.EXAMPLE.COM
Part 1 [1-1]: TEXT/PLAIN; parent=0; boundary=""
Parts: 1 skipped: 0
27 Nov 2001 10:13:01 From joe@EXAMPLE.COM: SUBSCRIBE TEST Joe User
27 Nov 2001 10:13:01 Sent information mail to joe@EXAMPLE.COM
27 Nov 2001 10:13:01 Sending WELCOME message to joe@EXAMPLE.COM
27 Nov 2001 10:13:01 Sent information mail to joe@EXAMPLE.COM
```

### 10.3.26 Content Filter Rejection Message

```
5 Dec 2001 17:31:07 -> Rejected:
* Your posting to the TEST list has been rejected by the content filter. 000
* messages are not allowed on this list.
5 Dec 2001 17:31:07 Sent information mail to ncb@EXAMPLE.COM
```

## 10.4 Interpreting the SMTP Logs (Windows Servers Only)

These logs are for the SMTP.L.EXE "listener" service, and are called SMTP-yyyymmdd.LOG.<sup>1</sup> They are found in the /LISTSERV/LOG directory with the other LISTSERV system logs. A typical "listener" log looks like this:

*Figure 10-2 Typical SMTP Log for SMTP.L.EXE Listener*

```
13 Mar 1998 14:13:31 LISTSERV SMTP listener, version 1.0d
13 Mar 1998 14:13:31 Copyright L-Soft international, 1994-98
13 Mar 1998 14:13:31 Initialization complete.
13 Mar 1998 14:15:59 New connection (1) from 128.118.56.2
13 Mar 1998 14:18:50 New connection (2) from 199.3.65.5
13 Mar 1998 14:18:59 >>>(1) Connection closed by remote host.
13 Mar 1998 14:19:05 Closing connection (2) from 199.3.65.5.
13 Mar 1998 14:19:08 New connection (3) from 205.186.43.7
13 Mar 1998 14:20:08 Closing connection (3) from 205.186.43.7.
```

This log simply keeps track of incoming SMTP connections to your server. It adds an entry for each new connection as it opens and closes. Additionally, if a connection is closed abnormally (e.g., by the remote host before any data is sent), the condition is

1. If you are running L-Soft's legacy LSMTP product and do not have SMTP.L.EXE "listener" running, then these logs will not be generated.

logged. The SMTP log is generally useful only for debugging the SMTP listener, although it may also be of use in debugging problems with specific remote hosts.

### 10.5 Interpreting the SMTP Worker Log Entries (Non-VM Only)

If you have SMTP "workers" activated on a Windows or OpenVMS server, each "worker" creates a separate log for itself. These logs are found in:

- OpenVMS: `LISTSERV_ROOT:[LOG]SMTPSn-yyyymmdd.LOG`
- Windows: `LISTSERV\LOG\SMTPSn-yyyymmdd.LOG`

If you are using SMTP "workers" under unix, no `smtps*.log` files are generated. Rather, the `lsv "worker"` sub-processes log information to the main `listserv.log` file.

(LISTSERV automatically "turns" the OpenVMS and Windows SMTPS logs at midnight. "yyyymmdd" is the year, month and day of the log, for example, `LISTSERV-19980104.LOG` is the log for 4 January 1998; "n" refers to the worker that is generating the log. Each worker generates a log, so if you have 10 workers running, you will have logs for `SMTPS1` through `SMTPS10`.)

A typical SMTP "worker" log looks like this<sup>1</sup>:

*Figure 10-3 Typical SMTPS Log for the SMTPW.EXE SMTP "Workers"*

```
03 Jun 1996 13:49:21 *** LSMTMP extensions activated ***
03 Jun 1996 13:49:03 500 error reading data line
03 Jun 1996 13:49:04 Renaming 8891738.MAIL to 8891738.MAIL-ERR1.
03 Jun 1996 14:06:54 Error opening '8894361.MAIL': Permission denied
03 Jun 1996 14:09:55 Error opening '8894695.MAIL': Permission denied
03 Jun 1996 14:40:45 Error opening '8897761.MAIL': Permission denied
```

The SMTP worker logs keep track of events that occur while the SMTP workers are delivering mail to the external mail host(s) defined in the `SMTP_FORWARD_n` variables in the site configuration file. In general, the events logged will be errors of one kind or another. For instance, the first error in the example above indicates an SMTP error, and the second indicates that a file that caused an error has been renamed so that it can be examined for debugging.

The last three errors are normal and can generally be ignored. They refer to the fact that two workers have noticed a `.MAIL` file that exists in the queue, and that one of them grabbed it before the other one did. The first worker locks the file and the second worker is denied permission to open it. Sometimes the first worker may process the mail so quickly that the error will read "File not found" rather than "Permission denied"; either way, this should not be considered alarming unless mail is actually not being delivered.

### 10.6 Change Logs

*This feature and keyword are not available in LISTSERV Lite.*

This feature is available to track certain operations on lists with "Change-Log= Yes" coded into their headers. As noted elsewhere in this manual, setting the keyword to "Yes"

1. The first line in the example simply indicates that special extensions for use with L-Soft's legacy LSMTMP product have been activated. This message will appear only if you are licensed for LISTSERV-HPO.

causes LISTSERV to write a file called *listname.CHANGELOG* (or *listname CHANGELOG* for VM) into LISTSERV's A directory or A disk. CHANGELOG files are automatically available for list owners and site maintainers to GET and PUT (PUT normally being used to delete them) like any other file. It is not necessary to make catalog entries for CHANGELOG files.

It is also possible to define change-logs for regular mailing lists that rotate on a regular basis, either DAILY, WEEKLY, MONTHLY, or YEARLY. (The option SINGLE is provided for backward compatibility and is still the default, that is, LISTSERV does not ever rotate the change-log). The rotation periods are specified as a second parameter to the Change-Log= list header keyword, for example:

```
Change-Log= Yes,Monthly
```

– or –

```
Change-Log = Yes,Single
```

(the latter being equivalent to "Change-Log= Yes" as noted above). Rotated change-logs are renamed with the format *listname.CHANGELOG-yyyymm[dd[w]]* (depending on the rotation period selected) and may be retrieved with the GET command as usual.



**Documented Restriction:** NOLIST-\* change-logs are always SINGLE. In addition, Change-Log rotation is not available under VM.

The operations monitored are ADD, AUTODEL, BOUNCE, CHANGE, DELETE, POST, READD, RESUBSCRIBE, SET, SIGNOFF, and SUBSCRIBE. All abbreviations and synonyms are translated to their "official" forms, i.e., SUB, JOIN, and SIGNON are all translated to SUBSCRIBE for the purposes of the changelog. This makes it easy to write scripts to come up with statistics for a given list--you don't have to take variations of the commands into account.

Sample changelog entries are:

```
20011025100330 ADD xxxxxxxx@JPS.NET Lxxxx Pxxxxxxxxx
20011025120049 AUTODEL xxxxxxxx@EISA.NET.AU
20011025131221 BOUNCE xxxxxxxx@NONEXIST.COM
20011025214433 CHANGE xxxx@MCS.COM txxx@MCS.NET
20011025214434 DELETE xxxxxx@SINGNET.COM.SG
20011025060052 EXPIRE joe@EXAMPLE.COM
20011025232441 POST xxxxxxxx@M4.SPRYNET.COM Printer Drivers
20011025000400 RESUBSCRIBE xxxx@MAIL.MECHWART.MUMSZKI.HU MxxxxxZxxxxx
20011025113947 SET xxxxxxxx.xxxxxxxx@WDC.COM REPRO
20011025082712 SIGNOFF xxxxxxxx@STARNET.NET.AR
20011025085642 SUBSCRIBE xxxxxxxx@LYCOSMAIL.COM Kxx Wxxxxxxx
```

As you see, the SET entry tells you what options were set, and the POST entry tells you what the subject of the posting was. RESUBSCRIBE is a SUBSCRIBE operation that takes place when the user is already subscribed to the list, for example, to change the real name field in the user's subscription. BOUNCE is a special operation that takes place when using the "no-list" bounce-processing mechanism (described in the [Advanced Topics Guide for LISTSERV](#)). EXPIRE indicates that the renewal "grace period" for the subscriber in question has expired without the user sending a CONFIRM command and the user has been deleted from the list. Otherwise these entries are fairly self-explanatory.

You may also see

```
20011025165954 IMPORT 2 0 0
```

This type of record shows the basic results of an `ADD IMPORT` job. The numbers following the `IMPORT` recordtype stand for (1) recipients added, (2) entries changed, and (3) recipients forwarded to another host, in other words, the same results that are sent back to the `ADD IMPORT` invoker on completion of the job.

Additionally, the following entry is always written at the top of a new change-log file:

```
20011025120731 SUBCOUNT 111
```

This tells you how many subscribers existed on the list before this particular change-log was started. It is handy to know if you are trying to track historical subscription count trends.

The `BOUNCE` record reports the bouncing address and information about why the message bounced, with a syntax of

```
20020329174013 BOUNCE USER@ZYX.COM x.x.x Bounce Message Here
```

For example:

```
20021107112809 BOUNCE BOGUSUSER@RERUN.IN.LSOFT.COM 5.1.1 Mailer
[192.168.254.101] said: "550 5.7.1
<bogususer@RERUN.IN.LSOFT.COM>... Relaying denied"
```



**Note:** Even with the changelog rotation feature active, changelog files can get very large. It may be necessary to monitor the size of the changelogs on your server and delete them as disk space fills up. If a list owner wants the changelog information for his list, he should be instructed that it is his responsibility to `GET` old changelog files regularly and delete them himself each time. There is no facility in `LISTSERV` to delete the rotated changelog files automatically.

## 10.7 Using `LISTSERV` Logs and `SHOW CTR` to Extract Server Statistics

While `LISTSERV` does not provide a native method to display statistics, it is entirely possible to use scripts to post-process the `LISTSERV` console logs and changelogs to provide a wide range of statistics. Additionally, the native `SHOW CTR` command provides a breakdown of current and past `LISTSERV` traffic.

### 10.7.1 Sample Log-Processing Scripts

There are two unsupported REXX scripts available from L-Soft which can be used to extract various statistics from the `LISTSERV` console log and from changelogs. See

<ftp://ftp.lsoft.com/listserv/windows/contrib/cntpost.rexx>

<ftp://ftp.lsoft.com/listserv/windows/contrib/stats.rexx>

Both of these scripts were written for Regina REXX and are (in their current incarnations) Windows-specific, but could probably be ported to the unix version of Regina with work.



**Note:** These scripts are completely unsupported. Neither the script author nor the L-Soft support department is able to field support requests for unsupported scripts.

The first script is used to compile posting data from all lists on the server and issues a weekly report that looks like this:

```
LISTSERV posting statistics for LISTSERV.EXAMPLE.COM since 8 Feb 1999
-----
```

List Name	Indiv. Postings	Digests Generated	Indexes Generated	Total Recipients
=====	=====	=====	=====	=====
LIST1	223	7	0	14574
LIST2	0	0	0	0
LIST3	12	7	7	825
-----				
Totals:	235	14	7	15399

DISTRIBUTE and MAIL-MERGE jobs sent by individuals:

	Start Date/Time	End Time	Rcpts	Job Name	Invoker
=	=====	=====	=====	=====	=====
D	8 Feb 1999 14:07:08	14:07:09	29	DISTJOB1	USER@EXAMPLE.COM
M	12 Feb 1999 10:31:23	10:31:25	10334	MMDISTJO	USER@EXAMPLE.COM
	>>> Full job name: MMDISTJOB25				
	* FROM=mmdistjob25-nolist@listserv.example.com				

This report was generated by cntpost.rexx version 1.8-fix11c 1999/02/19

The second script is designed to be run against a listname.CHANGELOG file and produces a report similar to the old VM STATS command output (addresses have been changed to protect the innocent). There are two options--TOP (which produces posting information for only the top x number of posters--by default this is 20, the value can be raised or lowered by changing the variable setting in the code), and NOP (which suppresses the individual posting data altogether). If neither the TOP or NOP options are specified, individual posting data are produced for each and every address that posted to the list during the period represented by the changelog, which has the potential to produce very long reports. It is probably best to run this script with redirection to a file.

Sample output from the STATS.REXX script is shown below:

```
C:\rexx>rexx stats.rexx access-l top
Statistics for ACCESS-L from 25 Jun 1998 through 10 Mar 2000 (625 days)
Total lines processed: 27495
Total subscribers on list at start of period: 1465
Total subscribers on list at end of period: 2623
Increase: +1158

Number of expired subscriptions since start of period: 289
Last posting in this changelog was on 10 Mar 2000.
```

	Total units	Units/day
	=====	=====
ADD operations:	0	0.0000
AUTODEL operations:	1147	1.8352
BOUNCE operations:	0	0.0000
CHANGE operations:	153	0.2448
DELETE operations:	182	0.2912
IMPORT operations:	2	0.0013
Recipients added:	0	0.0000
Entries changed:	2	0.0013
Forwarded:	0	0.0000



POST operations:	16758	26.8128
READD operations:	0	0.0000
RESUBSCRIBE operations:	45	0.0720
SET operations (total):	2733	4.3728
ACK:	29	0.0464
NOACK:	240	0.3840
MSGACK:	0	0.0000
CONCEAL:	108	0.1728
NOCONCEAL:	4	0.0064
FILES:	2	0.0032
NOFILES:	1	0.0016
MAIL:	344	0.5504
NOMAIL:	688	1.1008
DIGESTS:	1162	1.8592
NODIGESTS:	160	0.2560
INDEX:	114	0.1824
NOINDEX:	5	0.0080
REPRO:	207	0.3312
NOREPRO:	97	0.1552
MIME:	78	0.1248
NOMIME:	274	0.4384
HTML:	213	0.3408
NOHTML:	79	0.1264
TOPICS:	0	0.0000
FULLHDR:	13	0.0208
SHORTHDR:	31	0.0496
DUALHDR:	5	0.0080
IETFHDR:	4	0.0064
SUBJECTHDR:	69	0.1104
FULL822:	0	0.0000
SHORT822:	0	0.0000
SIGNOFF operations:	2568	4.1088
SUBSCRIBE operations:	3908	6.2528

Ratio of SIGNOFF operations to SUBSCRIBE operations: 0.6571:1

Top 20 posters	Total		
	Units	Units/day	Units/mon(*)
=====	=====	=====	=====
Gxxxxxx.Wxxxxxx@xxxxxxxxxxxxxx.xx.xx	19	3.1667	0.6333
dxxxxxx@xxx.xxx.xx	18	3.0000	0.6000
dxx.exxxxx@xx.xxxxxxx.xx.xx	11	1.8333	0.3667
fxxxxxx@xx.xxx	11	1.8333	0.3667
mxxxxxx@xxxxxxxxxxx.xxx	8	1.3333	0.2667
cxxxxxx@xxxxxxxx.xxx	8	1.3333	0.2667
Hxxxxxx@xxxxxxxx.xxx	7	1.1667	0.2333
mxx@xxxxxxxxxxxxxxxxxxx.xxx	7	1.1667	0.2333
dxxxxx.xxxxxxx@xxx-xxx.xx	7	1.1667	0.2333
dxxxxxx@xxxxxxxxxxxxxxxxxxx.xxx	7	1.1667	0.2333
jxxxx@xxxxxxxxxxx.xxx	7	1.1667	0.2333
kxxxxxx@xxxxx.xxx	6	1.0000	0.2000
nxxxx_x@xxx.xxx	6	1.0000	0.2000
Pxxxxxx@xxxxx.xxx	5	0.8333	0.1667
G.F.G.Wxxxxxx@xx.xxxxxxx.xx	4	0.6667	0.1333
Nxxxxxx.Vxx@xxx.xxxxx.xx.xx	4	0.6667	0.1333
axxxxxxx@xxxxxxxxxxx.xxx.xxx	4	0.6667	0.1333
kxxxxxx@xxxxxxxx.xx.xx	4	0.6667	0.1333
txxxx.x.x@xx.xxx	4	0.6667	0.1333
xxxxxx@xxxxxxxxxxx.xxx.xx	3	0.5000	0.1000

-----

(\*) Units per month is total units / 30 days.

Top 20 posters	Last posted
=====	=====
Gxxxxxx.Wxxxxxx@xxxxxxxxxxxxxx.xx.xx	13-Jan-1999
dxxxxxx@xxx.xxx.xx	12-Jan-1999
dxx.exxxxxx@xx.xxxxxxx.xx.xx	12-Jan-1999
fxxxxxx@xx.xxx	13-Jan-1999
mxxxxxx@xxxxxxxxxxx.xxx	09-Jan-1999
cxxxxxx@xxxxxx.xxx	13-Jan-1999
Hxxxxxx@xxxxxx.xxx	13-Jan-1999
mxx@xxxxxxxxxxxxxxx.xxx	12-Jan-1999
dxxxxx.xxxxxxx@xxx-xxx.xx	13-Jan-1999
dxxxxxx@xxxxxxxxxxxxxxx.xxx	13-Jan-1999
jxxxx@xxxxxxxxxxx	13-Jan-1999
kxxxxxx@xxxxx.xxx	12-Jan-1999
nxxxx_x@xxx.xxx	09-Jan-1999
Pxxxxxx@xxxxx.xxx	11-Jan-1999
G.F.G.Wxxxxxx@xx.xxxxxxx.xx	13-Jan-1999
Nxxxxxx.Vxx@xxx.xxxxx.xx.xx	13-Jan-1999
axxxxxxx@xxxxxxxxxxx.xxx.xxx	10-Jan-1999
kxxxxxx@xxxxxx.xx.xx	13-Jan-1999
txxxx.x.x@xx.xxx	13-Jan-1999
xxxxxx@xxxxxx.xxx.xx	11-Jan-1999

This report was prepared by STATS.REXX 0.9 2002/01/14



**Note:** These scripts are not supported in any way by L-Soft, and your use of them is strictly at your own risk.

### 10.7.2 Interpreting the Output of SHOW CTR

LISTSERV provides certain raw statistics in response to the command `SHOW CTR yyyyymm` (where "yyyyymm" is the year and month for which you are requesting statistics). For instance, `SHOW CTR 200110` sent to one of L-Soft's hosting servers produces the following:

```
>SHOW CTR 200110
200110 1 0 102276 14829079 9414 351624 1686 5525 218474 189356 125315 29161
245922 59850 11234881 0 1654503 21893 40050348 302198 112853 63348 4373
291703 134 73552 END EOD
```

Unfortunately this is fairly obscure (it was originally intended to be used only by LISTSERV to compile network-wide statistics) and requires a certain amount of interpretation. The fields signify, in order:

- Month of report
- Version number (of this report format)
- Missing days
- Postings to mailing lists
- Number of recipients
- Digests issued

- Number of digest recipients
- Indexes issued
- Number of index recipients
- DISTRIBUTE jobs processed
- DISTRIBUTE jobs internally generated
- Outbound DISTRIBUTE jobs
- Outbound NJE files
- Outbound files to MAILER
- Outbound files to MAILER in non-BSMTP format
- Number of recipients in the outbound files to MAILER
- GLX requests
- Bandwidth usage register #1
- Bandwidth usage register #2
- CPU usage in milliseconds
- Number of bounces received
- Number of bounces received in non-standard format
- Number of bounces detected by heuristics
- Probes
- Number of virus scan operations (1.8e on)
- Number of viruses found (1.8e on)
- Number of copies of infected messages stopped (low bit)<sup>1</sup>
- Number of copies of infected messages stopped (high bit)<sup>2</sup>
- Future use
- Future use
- Future use
- Future use
- END (tells LISTSERV that this is the end of the regular data)
- EOD (tells LISTSERV that this is the end of the report)

The two bandwidth registers are used as follows:<sup>3</sup>

$$\text{Bandwidth used in bytes} = \text{first\_register} + (\text{second\_register}/100000)$$

The two "infected messages stopped" registers are used as follows:<sup>4</sup>

$$\text{Infected messages stopped} = (\text{low bit}/10000000) + (\text{high bit}/10)$$

- 
1. For each virus detected, this counter increases by the number of outbound copies of the virus that LISTSERV was about to send when the virus was stopped.
  2. A high-order bit was added to convert this counter from 32- to 64-bit in October 2003.
  3. Please note that there was an error in this formula in previous versions of the manual that was not discovered until version 15.0 was in development. The bandwidth formula has been corrected. Also, CPU usage is in milliseconds, not microseconds as previously documented.

The "future use" counters are undocumented, but may be observed to increment if the SPAM\_EXIT feature is enabled.

If you send a SHOW CTR command for the current month, LISTSERV inserts the following between the END and EOD markers:

```
XPOL
integer_value
integer_value
```

For instance, the output on 10 August 2005 at approximately 10:15 -0500 on one of L-Soft's hosting servers was

```
>SHOW CTR 200508
200508 1 0 6151 6704302 1300 223483 704 9003 34593 17229 23136 17628 74524
64977 348776 0 266059 60477 26257717 302407 68659 24196 97223 754088 2614
39464990 0 0 0 0 0 END XPOL 744 227 EOD
```

The XPOL numbers are used by L-Soft to extrapolate data for the current month and can generally be ignored.

## 10.8 Using the System Changelog to Track Distributions

*This feature is not available in LISTSERV Lite.*

If enabled (see the [Site Configuration Keyword Reference](#) document, SYSTEM\_CHANGELOG) a file called `system.changelog` (SYSTEM CHANGELOG under VM) is generated in LISTSERV's A directory (A disk under VM), containing records of the following sort:

```
20000605014426 MAIL-MERGE 1 0 1,MM1,owner-XYZ@GUAVA.EASE.LSOFT.COM,Re: Whatever
20000803010016 MAIL 22544 8304 21,MYLIST-L,LISTSERV@GUAVA.EASE.LSOFT.COM,owner-
mylist-l@GUAVA.EASE.LSOFT.COM,How now brown cow?
20000804134805 DIST-NJE 1 0 1,X-SPAM.JOB,LISTSERV@PLUM.EASE.LSOFT.COM,LISTSERV@
LISTSERV.EXAMPLE.COM
20000804190101 DIST-NJE 1 0 1,Netwide_SIGNOFF,LISTSERV@PLUM.EASE.LSOFT.COM,LIST
SERV@LISTSERV.EXAMPLE.COM
20000804230002 DIST-NJE 0 119 1,SUPD,LISTSERV@GUAVA.EASE.LSOFT.COM,LISTSERV@GUA
VA.EASE.LSOFT.COM
```

The records consist of four comma-separated tokens:

1. Information about the job
2. Name of the job (or list to which the mail was sent)
3. Bounce address
4. Subject line

The first token contains multiple space-separated parameters. As of this writing the parameters are:

- Date and time the job was processed (yyyymmddhhmmss)
- Type of job - MAIL, MAIL-MERGE, or DIST-NJE. Typically you will see DIST-NJE only for interserver update jobs, for example, SUPD, X-LUPD, etc.
- Number of recipients processed locally

---

4. While normally the high bit of a counter comes first, this case is an exception. For backward compatibility with older statistics-gathering scripts, the new high bit had to come last.

- Number of recipients forwarded to another DISTRIBUTE server
- Size of the message rounded up to the nearest kilobyte

(More parameters may be added to the first token in future versions. If you write a local application to parse the changelog records, be sure to take this into account.)

A VIRUS job type is also available to track intercepted viruses. Assuming that 1) the `system.changelog` is enabled and 2) LISTSERV's anti-virus scanning facility is enabled, LISTSERV will write records like the following to the `system.changelog` file for each virus encountered:

```
20020110152955 VIRUS TEST 6 EICAR-Test-File
20020128144949 VIRUS *LSWAVDD* 1 EICAR-Test-File
```

The record consists of:

- Date and time the job was processed
- The VIRUS job type
- The list to which the message containing the virus was posted (if sent to a -request or -owner mailbox, this part of the record contains \*LSWAVDD\*, as in the second example above)
- LISTSERV's best guess of the number of outbound copies that have been suppressed. This is not always 100% correct, but very close. When LISTSERV does not know, the value 1 is assumed. A value of 0 is possible, for instance if the list had no recipients.
- The name or description of the virus as provided by F-Secure

## 10.9 Logging Changelog Information to a DBMS

A copy of changelog information may be stored in a DBMS. This requires a pre-existing DBMS connection configured in the usual way (see the [Advanced Topics Guide for LISTSERV](#) if you need guidance), and is controlled by three new site configuration parameters, and a table that must be created manually. The parameters, which are defined in LISTSERV's site configuration file, are `CHANGELOG_DBMS`, `CHANGELOG_DBMS_TABLE`, and `CHANGELOG_DBMS_CONNECTION`. The first two are mandatory while the third is optional.



**Note:** It is not possible for LISTSERV to write the changelog in multiple different tables based on various combinations of parameters. This can be accomplished on the DBMS side with a stored procedure if required.



**Important:** The DBMS copy is just that, a copy. The disk files (\*.changelog) are still generated. Naturally if they are not needed they may periodically be erased with a script.

- `CHANGELOG_DBMS` (mandatory)

This controls which entries are to be copied to the DBMS. Note that only entries that are actually generated can be copied. If a given change-log is disabled, it will not go to the DBMS even if you request it in this variable.

The value can be ALL or a space-separated combination of SYSTEM, NOLIST (matches any NOLIST-xxx), LISTS (matches any list) or the names of individual lists.

- CHANGELOG\_DBMS\_TABLE (mandatory)

This contains five space-separated names:

- The name of the table in which to store changelog entries.
- The name of a time-stamp column in which to write the current date and time. This must be a DATE (Oracle), TIMESTAMP (DB2), DATETIME (SQL Server), or whatever else can store both date and time. This cannot be a character string.
- The name of a VARCHAR or equivalent column storing the name of the list. The maximum size depends on the list names you choose, but it should be at least 40.
- The name of a VARCHAR column storing the record type (BOUNCE, etc.). This should probably be around 40.
- The name of a VARCHAR column storing the parameters. This ought to be 256 or so.

If any of these parameters is missing, the setting is ignored.

- CHANGELOG\_DBMS\_CONNECTION (optional)

This contains two optional space separated words:

- The type of driver to be used (CLI, OCI, ODBC). This defaults to your system default driver type.
- The server to connect to (similar to SERVER=). This defaults to the empty string, that is, the default server.

As an example, let us say that you have created a table called CHANGELOG in the database to which LISTSERV is connected. The CHANGELOG table has four columns, which are called TIMESTAMP, LISTNAME, RECORDTYPE, and PARAMETERS. So you would open your system configuration file in a text editor and add the following entries:

- Windows: (site.cfg)

```
CHANGELOG_DBMS=ALL
CHANGELOG_DBMS_TABLE=CHANGELOG TIMESTAMP LISTNAME RECORDTYPE PARAMETERS
```

- Unix: (go.user)

```
CHANGELOG_DBMS="ALL"
CHANGELOG_DBMS_TABLE="CHANGELOG TIMESTAMP LISTNAME RECORDTYPE PARAMETERS"
export CHANGELOG_DBMS CHANGELOG_DBMS_TABLE
```

- OpenVMS: (site\_config.dat)

```
CHANGELOG_DBMS "ALL"
CHANGELOG_DBMS_TABLE "CHANGELOG TIMESTAMP LISTNAME RECORDTYPE PARAMETERS"
```

If you wanted only to log the information from the system and NOLIST changelogs, you would change

- Windows: CHANGELOG\_DBMS=ALL
- Unix: CHANGELOG\_DBMS="ALL"
- OpenVMS: CHANGELOG\_DBMS "ALL"

to

- Windows: CHANGELOG\_DBMS=SYSTEM NOLIST
- Unix: CHANGELOG\_DBMS="SYSTEM NOLIST"
- OpenVMS: CHANGELOG\_DBMS "SYSTEM NOLIST"

and so forth.



## Section 11 Using the Web Administration Interface

LISTSERV 15.0 has a completely revamped Web Administration Interface, making LISTSERV administration significantly easier. It is now possible to change many LISTSERV site configuration settings "on the fly", although some changes may still require a restart of the server before they are recognized.

Most sites will be able to upgrade to LISTSERV 15.0 without losing local web customizations, although this is not optimal and will not generally expose new features to your users. To assist you in customizing the new LISTSERV 15.0 web interface, L-Soft has produced a Customization Guide, which is available in PDF format at the following location:

[http://www.lsoft.com/manuals/15.0/LISTSERV15.0\\_CustomizationGuide.pdf](http://www.lsoft.com/manuals/15.0/LISTSERV15.0_CustomizationGuide.pdf)

Since the comprehensive guide takes you through the whole gamut of interface customization, this section will concentrate solely upon a basic understanding of the new features of the interface.

The LISTSERV 15.0 Web Interface requires JavaScript to be enabled by default. However, those who prefer not to use JavaScript can set their Navigation Style user preference to "Non-Script Navigation" in the Preferences menu.

### 11.1 The Default LISTSERV Home Page

The default home page for LISTSERV typically is reached by using the URL:

- On unix: `http://yourhost.domain/cgi-bin/wa`
- On VMS: `http://yourhost.domain/htbin/wa`
- On Windows: `http://yourhost.domain/scripts/wa.exe` or `http://yourhost.domain/cgi-bin/wa.exe`

Of course this is not standardized; the location of the 'wa' script is determined by the value of `WWW_ARCHIVE_CGI` in LISTSERV's site configuration file. In any case, invoking 'wa' without any parameters returns the default home page.

### 11.2 Logging In

You can log into the list administration interface from any list's main web archive index page (assuming that this link has not been removed by the list owner; it exists in the `WWW_INDEX` mail template by default). The interface may also be reached by a link from the default LISTSERV home page mentioned in Section 11.1 [The Default LISTSERV Home Page](#).

To access the list administration interface without a link, you point your web browser to the "wa" script. Typically the interface is accessed as follows:

- On unix: `http://yourhost.domain/cgi-bin/wa`
- On VMS: `http://yourhost.domain/htbin/wa`
- On Windows: `http://yourhost.domain/scripts/wa.exe` or `http://yourhost.domain/cgi-bin/wa.exe`

and by default, users are directed to the main archives page for the server.

Figure 11-1 LISTSERV Archives Screen

LISTSERV 15.0  
Subscriber's Corner Email Lists Log In

**LISTSERV Archives**

**DRAGONFLY.DC.LSOFT.COM**

Options: [Log In](#) | [Get Password](#)

Resources: [About LISTSERV](#)  
[LISTSERV Documentation](#)  
[CataList Email List Search](#)

List Name	List Title
ANNOUNCEMENTS	Announcement list for those pesky announcements. (1 Subscriber)
ASA	ASA list (0 Subscribers)
BETAJOHAN-L	Johan's Beta test list (2 Subscribers)
DISCUSSION	Open Discussion List (11 Subscribers)
HAT-TEST	Holly's test list (1 Subscriber)
HAT-TEST2	Holly's second test list (1 Subscriber)
MODERATED	Moderated List (8 Subscribers)
NEWDIGEST	New digest list (1 Subscriber)
NEWINDEX	New index list (1 Subscriber)
NEWLIST	Newlist title (0 Subscribers)
NICTEST	Nic's Test List (1 Subscriber)
PRESS	L-Soft Press Releases (0 Subscribers)

Access Unlisted Archives:

To get to the administrative pages, you will have to log in. If you already have a personal LISTSERV password, then you will simply log in with your existing userid and password.

Figure 11-2 Login Screen

**Login Required**

Please enter your email address and your LISTSERV password and click on the "Log In" button. If this is the first time you see this dialog, or if you have forgotten your password, you will need to [get a new LISTSERV password](#) first.

Email Address:

Password:



**Note:** The userid you use here must be associated with the personal password you have from LISTSERV. If you have registered a password as joe@unix.host.com and try to log in here as joe@host.com with that password, LISTSERV will reject your login.

Logging in will cause LISTSERV to issue you a cookie that allows you to bypass this login screen (and incidentally to stay logged into the interface for longer than 15 minutes without having to log in again when your session expires). If your workstation is not physically secure, you will want to log out after each management session.

### 11.3 Setting a LISTSERV Password

If you do not already have a personal LISTSERV password (set with the PW ADD command or via the web interface) or cannot remember your password, you need to define one now. If you choose to do this via the web interface, simply click the hyperlink and you will get the following screen:

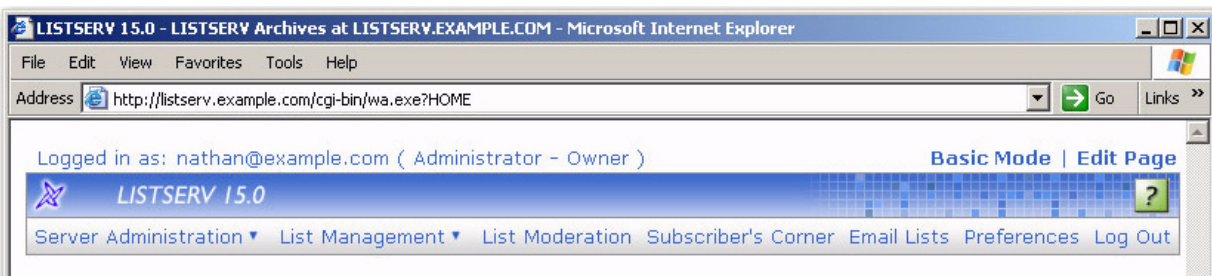
Figure 11-3 Registering a LISTSERV Password

When you hit the **[Register password]** button, you will be told that a confirmation has been emailed to you. You will have to activate your password by responding to the email (or clicking the link it contains).

### 11.4 Logging In and Setting Preferences

When you log in for the first time, you will be returned to the main list archives page, but with a difference: The top toolbar will have a lot more options.

Figure 11-4 Web Interface Toolbar



The interface tells you who you are logged in as, and what functions your logged-in address is entitled to perform. It also tells you what mode you're set to (the Basic Mode

by default; there are two other modes, "Expert" and "Tutorial"). If you are entitled to edit the page, you can go straight to the page editing wizard by clicking **Edit Page**.

Clicking the **LISTSERV logo** icon at the left takes you to L-Soft's home page. Clicking "**LISTSERV 15.0**" takes you to your default start page.

 Help pages are accessed by clicking the **Help** icon at the right hand side of the toolbar.

The **Server Administration** menu gives you access to the Server Dashboard, site configuration functions, mailing list creation and deletion, server reports, server customization (mail and web templates), and the ad-hoc LISTSERV command entry page.

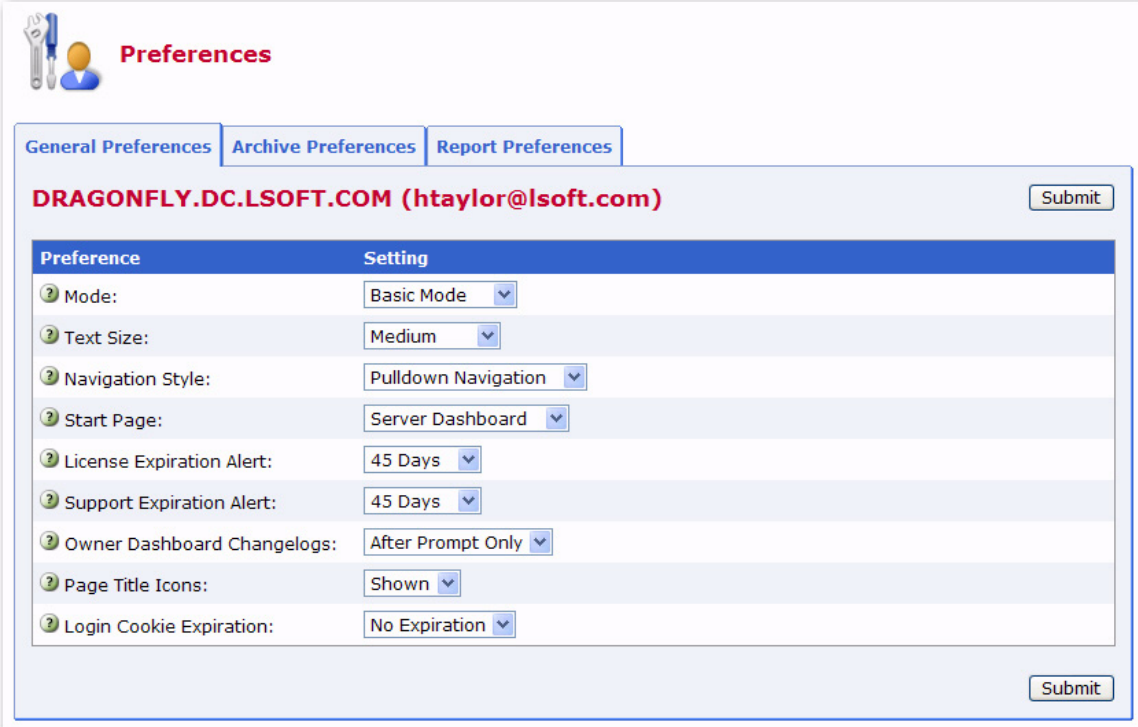
The **List Management** menu gives list owners access to the List Dashboard, list configuration, customization, and subscriber management.

**List Moderation** centralizes the moderation function, and it will show all messages needing moderation from the `userid@host` under which you are logged in.


**Email Lists** takes you back to the main list archives page.

The first thing you may want to do is click **Preferences**. One of the things you can set is the default start page. Site administrators will probably want to start at the server administration dashboard, while list owners will probably want to start at the list owner dashboard. Or they may be perfectly happy to start at the list archives page. In any case, clicking on **Preferences** brings up the following page:

Figure 11-5 Preference Screen



Preference	Setting
Mode:	Basic Mode
Text Size:	Medium
Navigation Style:	Pull-down Navigation
Start Page:	Server Dashboard
License Expiration Alert:	45 Days
Support Expiration Alert:	45 Days
Owner Dashboard Changelogs:	After Prompt Only
Page Title Icons:	Shown
Login Cookie Expiration:	No Expiration

 More information can be found by clicking the **Help** icons associated with each preference.

## 11.5 List Maintenance via the Web Interface



**Important:** The LISTSERV 15 Web Interface cannot be used to manage lists that are coded `Validate= Yes,Confirm,NoPW` or `Validate= All,Confirm,NoPW` because passwords are not accepted for validation in those cases.

The Subscriber Management screen lets list owners examine, delete, and add subscribers to a specific list. To access this screen, select **Subscriber Management** from the **List Management** menu. From this screen, select the list you want to work with, and then select either the Single Subscriber or Bulk Operations tab. The Single Subscriber tab lets the list owner examine or delete a subscription and add a new subscriber to the list. To add or delete many subscriptions at a time, use the Bulk Operations tab.



**Note:** To examine, modify, or delete multiple subscriptions at once, you can also use the Subscriber Reports screen.

### 11.5.1 Examining or Deleting a Subscription

This works very much like the "SCAN" command. Simply enter your criteria in the text box and click the **[Search in List]** button.

Figure 11-6 Examining or Deleting a Subscription

If there is no match for your entry, then you will get back the same page but with a Scan: No match message at the top. If, on the other hand, your search is successful, one of two things will happen.

If there are multiple matches for your criteria, a screen will be displayed with a scrollable list box containing all of the matches

Figure 11-7 Select Subscriber to Examine or Delete

Next, simply choose the user you want to examine or delete and click on the appropriate button. If you did not find what you were looking for, you can press the **[New Search]** button to get a new search screen.



If there was only a single match to your query, then the preceding screen will be bypassed and you will go directly to the next screen, which is the Subscriber Management screen for the subscription. It displays the values of all the settings for that subscription, including the subscription date and name. From the account management screen, you can delete the subscription or change the name, the email address, or the subscription options associated with the subscription. A sample of the screen is shown below.

Figure 11-8 Subscriber Management Screen

**Subscriber Management (HAT-TEST)** HAT-TEST Home

**View or Set Subscription Options**

**Notification Options:**

- Send Email Notification
- Do Not Notify the User

**Name:**

**Email Address:**

**Subscribed Since:** 13 Jul 2006

**Subscription Type**

- Regular [NODIGEST]
- Digest (traditional) [NOMIME DIGEST]
- Digest (MIME format) [NOHTML MIME DIGEST]
- Digest (HTML format) [HTML DIGEST]
- Index (traditional) [NOHTML INDEX]
- Index (HTML format) [HTML INDEX]

**Mail Header Style**

- Normal LISTSERV-style header [FULLHDR]
- LISTSERV-style, with list name in subject [SUBJECTHDR]
- "Dual" (second header in mail body) [DUALHDR]
- sendmail-style [IETFHDR]
- Normal LISTSERV-style (RFC 822 Compliant) [FULL822]

**Acknowledgements**

- No acknowledgements [NOACK NOREPRO]
- Short message confirming receipt [ACK NOREPRO]
- Receive copy of own postings [NOACK REPRO]

**Miscellaneous**

- Mail delivery disabled temporarily [NOMAIL]
- Address concealed from REVIEW listing [CONCEAL]
- User is exempt from renewal/probing [NORENEW]
- User may bypass moderation [EDITOR]
- All postings sent to list owner for review [REVIEW]
- User may not post to list [NOPOST]

### 11.5.2 Adding a New Subscriber to the List

To add a new subscriber, select the list you want to add the subscriber to. Then, in the **Add New Subscriber** section, enter the email address and name of the new subscriber,

select whether or not to send an email notification to this subscriber, and click the **[Add to List]** button.

Figure 11-9 Adding a New Subscriber



**Note:** The full name of the subscriber is optional. If omitted, then the user will be added anonymously to the list.

### 11.5.3 Bulk Operations

The Bulk Operations tab allows a list owner to upload an input file containing email addresses and (optionally) names, one address per line, and either add all the email addresses in the file to the list (optionally replacing the current subscribers) or remove them from the list.

The input file is created on your own machine with an ASCII text editor. After clicking the **[Import]** button you will see a command response similar to the following:

- If the **Add the imported address to “List”; do not remove any subscribers** option is selected:  
*ADD: no error, 202 recipients added, no entry changed, no duplicate, none forwarded.*
- If the **Remove all subscribers from “List”, and add the imported address** option is selected:  
*DELETE: 14 subscribers removed.*  
*ADD: no error, 38 recipients added, no entry changed, no duplicate, none forwarded.*  
(If this option is selected, but no input file is specified, then you will only get the DELETE message.)
- If the **Remove the imported addresses from “List”; do not add any subscribers** option is selected:  
*DELETE: 93 subscribers removed.*
- If the **Remove the imported addresses from all lists** option is selected:  
*DELETE: 243 subscribers removed.*  
*DELETE: 109 subscribers removed.*  
*Global deletion process complete, 352 entries removed.*
- If you do not supply an upload file where required, or if your browser does not support the RFC1867 file upload extension, you get the following message:



Your browser did not upload any file during the transfer. Assuming you did fill in the file input box, the most likely cause is that your browser does not support the file upload extension (RFC1867).

Figure 11-10 The Bulk Operations Tab



**Notes:** Bulk operations are not enabled by default. The site manager must enable this functionality explicitly. If you get an error 2 when you click on the **[Import]** button, this means that the "upload" directory has not been created. If you get an error 13 when you click on the **[Import]** button, this means that the "upload" directory has been created but the CGI program user does not have write permission in that directory. In addition, the input file must be a plain text file (not a word processor document or spreadsheet) and must contain one address per line, optionally followed with a space (or TAB) and the subscriber's name. In addition, the subscribers being added or deleted will not be notified.

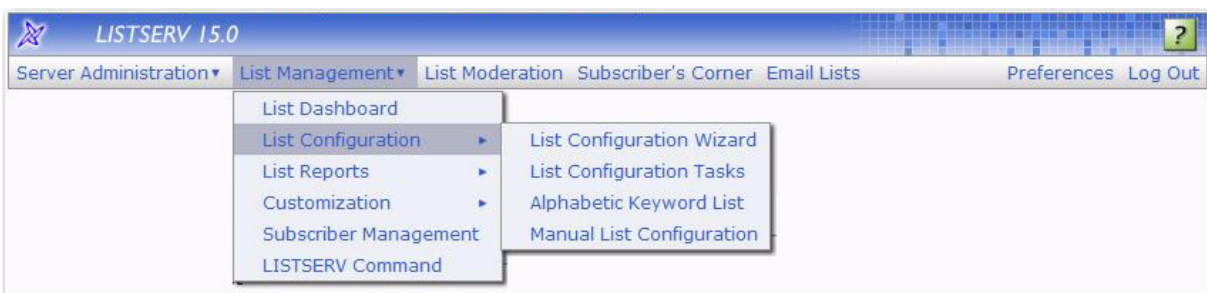
## 11.6 List Configuration

Lists can be configured using a wizard, which guides you step-by-step through the configuration process, or manually.

To open the List Configuration Wizard, click on the **List Management** menu, select **List Configuration**, and then select **List Configuration Wizard**.

To configure a list manually, click on the **List Management** menu, select **List Configuration**, and then select **Manual List Configuration**.

Figure 11-11 The List Management Menu



For those of you who want to configure the list manually, you can edit the list header in its "raw" state. This is only recommended for people who are very comfortable with the format of the LISTSERV list header and know the keywords and their parameters very well.

The list header appears in a multi-line text box that can be scrolled up and down. You simply type in the changes or added lines just as if you were using a regular text editor. When you are finished, click the **[Submit]** button to submit the changes. If you want to start over, you can click the **[Reload]** button to reload the header information from the server.

When you submit your changes with the **[Submit]** button, you will get the same kind of feedback from LISTSERV as you would if you sent a PUT operation by mail. The next screen will either say that the header of the list has been successfully updated, or it will indicate that it has found errors and that the header has not been stored. The feedback page also has a text box containing the header information you've just stored (or tried to store) so if you need to make further corrections to the header, you don't have to back up and start over.

The list header keywords and their parameters are documented in the [List Keyword Reference](#) document, in the online help, or (when using the configuration wizard) by clicking the **Help** icon for each keyword.

## 11.7 Maintaining Mail and WWW Templates via the Web Interface

The template editor allows the site administrator and list owner to customize the majority of the Web Interface Pages and Administrative Messages sent out by LISTSERV. There are two types of templates you can customize – Web and Mail.

**Web Templates** (also referred to as Dynamic Web Templates) control the pages produced by the Web interface. These pages are produced dynamically when they are accessed. What gets displayed by the browser depends on the circumstances and may change depending on who is accessing the interface, which list they are accessing, the settings of that list, and so on.

Commands in Web Templates begin with a plus sign "+" and variables begin with an ampersand followed by a plus sign "&+".

**Mail Templates** control text produced by LISTSERV itself. Although generally categorized as mail templates, they actually fall into three different types:

- Mail Templates control the contents of email messages sent by LISTSERV. A mail template is a complete email message. Formatting commands are available, substitutions that make sense in the context of the specific message are available, and while other templates may be imbedded with the .IM command, the message is in and of itself ready for LISTSERV to send.
- Message Templates supply text that will ultimately be shown to the user. These messages may be included in a mail template; or they may be included in an email sent by LISTSERV in response to LISTSERV commands sent by email to the LISTSERV command address; or they may be returned to the Web interface in response to commands sent through the Web interface. Limited formatting is available.

- Message Fragments templates are the lowest level of mail templates. Fragments are pieces of text produced by LISTSERV as parts of other messages or emails. For example, list digests must follow a certain format dictated by the Internet RFCs. Therefore, it is not possible to provide a complete mail template for digests. However, some of the text within the digest is not mandated by the RFCs, and so LISTSERV provides some fragment templates to control these parts, for example MSG\_DIGEST\_FRAGMENT\_DATERANGE1 to control the date range and MSG\_DIGEST\_FRAGMENT\_PREAMBLE1 to control the text preceding the table of contents. Formatting commands are generally not available in fragments.

Commands in Mail Templates begin with a period "." and variables begin with an ampersand "&".

To access the template editor, click on the **List Management** menu, select **Customization**, and then select either **Web Templates** or **Mail Templates**.

Once the template editor has opened, simply select the list and template you want to work with. The template editor also lets you create new template.

For more information about customization templates, see the [Customization Manual for LISTSERV 15.0](#) or click on the link(s) located in the **Tips** section at the bottom of the editor.

## 11.8 Sending Interactive Commands via the Web

The LISTSERV Command Interface is used for submitting LISTSERV commands that are not otherwise facilitated by the Web interface. See Appendix A: [Command Reference Card](#) for a listing of all commands.

For some commands, the response is automatically displayed by the Web interface. For others, a special command parameter must be used in order to display the response in the browser, otherwise the response is sent by email. In addition, other commands are only able to respond by email.

To access the LISTSERV Command Interface, click on the **List Management** menu, and then select **LISTSERV Command**.

The Command Interface can only be used for single line commands. In particular, the PUT command will not work through the Web interface. Multi-line commands must be sent by email.

A selection of frequently used commands is available at the bottom of the screen.

## 11.9 Mail-Merge

Advanced mail-merge features are available and can be accessed either by sending specially-formatted DISTRIBUTE jobs to LISTSERV or by using the web administration interface. The web interface is not a "wizard" but simply an interface that allows you to "cut and paste" a mail merge message and select different standardized groups of list subscribers to whom the message is to be sent.



**Notes:** LISTSERV's mail-merge functionality REQUIRES the use of LISTSERV's Embedded Mail Merge feature. For more information, see the EMM keyword in the [Site Configuration Keyword Reference](#) document. Mail-merge functions are documented fully in the [Advanced Topics Guide for LISTSERV](#).

## 11.10 Server Administration Interface

LISTSERV 15.0 also includes a maintainer-level server administration interface. From this interface it is possible to

- Create lists
- Change site configuration variable settings
- Enter the list management (list owner) area
- Customize the default LISTSERV home page
- Customize the default web interface layout
- Customize the site-wide web and mail templates
- Send a mail-merge message using the DBMS back-end (requires a DBMS interface; see the section on DBMS features in the Advanced Topics Guide for LISTSERV for more information)
- Execute an arbitrary LISTSERV command (as with LCMD). Note that certain commands will be answered by mail rather than through the interface.

To access the server administration interface, use the appropriate link from the LISTSERV home page described in Section 11.1 [The Default LISTSERV Home Page](#).

Each feature of the administration interface is fully self-documented.



**Important:** When creating lists via the web interface under unix and VMS (with PMDF), it is still necessary to make the mail aliases required in 7.2.1 (for unix) or 7.2.2 (for VMS), above. The web interface will not make these aliases for you.

The main entry page for the administration interface is the Administration Dashboard. This Dashboard concentrates important information about the server, the current version you are running vs. the latest available version, license information, anti-virus information, and technical support information.

The bottom section shows monthly statistics for the server. The default set of statistics can be changed to show only those statistics you really want to see on a regular basis. To change the stat set, click the **Edit Table** link at the top left of that section. Columns are removed by clicking the **[X]** under the caption, and columns are added by choosing a statistic from the **Columns** drop-down text box and clicking the **[Add Column]** button at the right. When done editing, click the **Stop Editing** link at the top left.


Figure 11-12 The Administration Dashboard

Logged in as: htaylor@lsoft.com ( Administrator - Owner ) Basic Mode | [Edit Page](#)

**LISTSERV 15.0** [?](#)

Server Administration ▾ List Management ▾ List Moderation Subscriber's Corner Email Lists [Preferences](#) [Log Out](#)

---


 **Server Administration Dashboard**

---

**DRAGONFLY.DC.LSOFT.COM**

**Server Information**


LISTSERV Type:               LISTSERV(R)  
LISTSERV Version:           15.0  
Build Date:                   8 Feb 2007  
Running Under:               Windows XP Professional  
Runmode:                     Standalone  
Alias File Version:          2006-09-01 16:56:00  
Virus Database Version:     2007-04-02 14:24:25 (F-Secure Anti-Virus 5.44)

 You have the most recent version of LISTSERV.  
LISTSERV Version: 15.0 (Build Date: 8 Feb 2007)  
Web Interface Version: 2.4.1 R2563

**License Information**


License Type:                Permanent  
Expiration Date:             31 Dec 2007  
Maintenance Until:         31 Dec 2007 (graduated license)  
Capacity:                    Unlimited  
Version:                     15.0  
Serial Number:               LSOFT-JKUMPULA-4T

Comments:                   LSOFT



 Your LISTSERV license is current.  
License Type: Permanent  
Expiration Date: 31 Dec 2007

---

**Anti-Virus Protection**

 Anti-virus protection is enabled on DRAGONFLY.DC.LSOFT.COM. All incoming and outgoing messages are scanned for viruses before delivery.  
Virus Database Version: 2007-04-02 14:24:25 (F-Secure Anti-Virus 5.44)

**Technical Support**

 Your technical support is current, which entitles you to assistance from L-Soft's support staff as well as free access to new versions and upgrades. 

Maintenance Until: 31 Dec 2007 (graduated license)

---

[Edit Table](#)

Year and Month ▲	Postings	Recipients	DISTRIBUTE Jobs	Bandwidth Usage (in MB)	Bounces Received	Bounces Detected	Infected Messages Stopped	Virus Scan Operations	Viruses Found	Spam Messages Stopped	Spam Scan Operations	Spam Detected
2006/09	27	33	20	1	4	0	2	126	2	0	0	0
2006/10	15	12	12	0	0	0	1	97	1	0	0	0
2006/11	55	88	37	3	0	0	0	304	0	0	2	0
2006/12	11	3	1	0	0	0	0	66	0	0	0	0
2007/01	0	0	0	0	0	0	0	2	0	0	0	0
2007/02	1	0	0	1	0	0	0	29	0	0	0	0
2007/03	0	0	0	0	0	0	0	10	0	0	0	0
2007/04	0	0	0	0	0	0	0	0	0	0	0	0



## Section 12 Distribution Features and Functions

For more information on these features, see also the [List Keyword Reference](#) document.

### 12.1 Controlling the Default Level of Acknowledgement to User Postings

You can control the default level of acknowledgement sent back to users when they post to the list with the "Ack=" list header keyword (see the [List Keyword Reference](#) document for details). This is particularly important because it also controls the acknowledgement level for users who are not subscribed to the list and cannot, therefore, set personal options. While the value set for "Ack=" can be overridden for subscribers both by the setting of the "Default-Options=" keyword (which sets the default at subscribe time) and by the "SET" command, this option will always be in effect when distributing mail from people who are not subscribed to the distribution list.

### 12.2 Controlling the Maximum Number of Postings Per Day

#### 12.2.1 Controlling Total Postings to the List Per Day

You can control the maximum number of postings per day on a list-by-list basis by setting the "Daily-Threshold=" list header keyword. The default value is 50 posts per day.

The following rules apply:

- If the Daily-Threshold value is not reached by midnight, the number of postings is reset to zero and starts over for the next day.
- If the Daily-Threshold is reached before midnight, and the list is not freed before midnight, postings released after midnight count toward the next day's quota. Thus, if the list is held on Tuesday evening and 15 postings accumulate before you free the list on Wednesday morning, the 15 postings count toward Wednesday's quota.

#### 12.2.2 Controlling the Number of Postings Per Day from Individual Users

You can control the maximum number of postings per day per subscriber on a list-by-list basis by setting the new (optional) second parameter of the "Daily-Threshold=" list header keyword. The default is to have no such limit.

If set, when the per-subscriber threshold is reached, the subscriber is told that his message cannot be processed because he has reached the limit for today, and that he should repost his message at a later time. The counter for this limit resets to zero at midnight for all lists.

This limit is waived for the list owner(s) and any list editors/moderators.

### 12.3 Controlling "Prime" Time

This feature reserves certain times and days of the week when you don't want LISTSERV to process postings. Although this is not usually necessary today, there can be certain applications for the use of the "PRIMETIME=" site configuration variable and the "Prime=" list header keyword.

"PRIMETIME=" controls the server-wide prime time setting. By default it is set to MON-SUN: -. There should be no need to change this setting under normal circumstances.

Please see the entry for PRIMETIME in the [Site Configuration Keyword Reference](#) document for further details.

The list header keyword "Prime=" controls prime time on a list by list basis. By default it is set to "Prime= Yes", meaning that it does not observe the PRIMETIME= variable. If explicitly set to "Prime= No", the value in the PRIMETIME= variable will be observed. It can be set to an explicit time definition if necessary. For instance, you might have a very large announce-only list (e.g., a newsletter) that should not be posted until after midnight (when network traffic is low and more machine resources are generally available). You might wish to set this list with a "Prime=" setting of

```
* Prime= "MON-SUN: 06:00-23:59"
```

or, if you want the list only to be processed between midnight and 6 AM on weekends, you might code

```
* Prime= "MON-FRI: 00:00-23:59; SAT-SUN: 06:00-23:59"
```

The specification for "Prime=" must be enclosed in double quotes. In addition, the minutes specification is cosmetic only. LISTSERV checks on jobs held awaiting non-prime time only once each hour, on the hour. Thus if you have

```
* Prime= "MON-SUN: 06:00-21:00"
```

then jobs awaiting non-prime time will be executed at 22:00, not 21:00 as you might otherwise expect. On the other hand, if you code

```
* Prime= "MON-SUN: 06:00-20:xx"
```

where "xx" is any two-digit integer between 01 and 59, then jobs awaiting non-prime time will be executed when LISTSERV runs its hourly check of PRIME jobs at 21:00.

If you need to open only one short window during one or more days, you can do this by coding something like:

```
* Prime= "MON-FRI: 00:00-02:59 04:00-23:59; SAT-SUN: -"
```

This example allows LISTSERV to process mail for the list only between 2 AM and 4 AM Monday through Friday, and at any time on Saturday and Sunday. Note that there is no punctuation--just a space--between the time settings for a given day or day sequence.

Mail sent to lists during prime time is automatically held until non-prime time and then distributed normally, without requiring further intervention by anyone. This means that the newsletter editor of the example list can post their next issue on Friday afternoon and know that it won't be distributed until Saturday at midnight or shortly thereafter.

One of the most common misconceptions regarding the prime time settings is that prime time is the time during which LISTSERV will process postings for your list (or globally for the server if you change "PRIMETIME="). Please remember that when you set a "prime time" either for a list or globally for the entire server, you are setting the time during which LISTSERV does not process postings. It is "prime time" for the machine when it should be doing other things, for example, number crunching, daily backups, or any other function during which LISTSERV should not be using cycles.

When you are coding a prime time specification that LISTSERV's week starts on Monday and runs through Sunday. Thus, something like the following examples:



```
Prime= "MON-TUE: 00:00-23:59; WED: -; THU-SUN: 00:00-23:59"
```

```
Prime= "TUE: 01:00-4:59; THU-SUN: 00:00-23:59"
```

are correct syntax, whereas

```
Prime= "WED: -; THU-SUN: 00:00-23:59; MON-TUE: 00:00-23:59"
```

is not. Furthermore note carefully the weekdays must be specified in their correct order, that is,

```
Prime= "THU-FRI: 00:00-23:59; SAT-MON: 21:00-23:59"
```

is not correct because it starts on Thursday and ends on Monday. The correct specification in this case would be

```
Prime= "MON: 21:00-23:59; THU-FRI: 00:00-23:59; SAT-SUN: 21:00-23:59"
```

## 12.4 "Holding" and "Freeing" a List

### 12.4.1 Automatic List Holds

On occasion, LISTSERV will automatically "hold" a list, i.e., postpone processing new mail for the list until either the list owner or the LISTSERV maintainer manually intervenes. There are two circumstances under which a list will be automatically held:

- When the daily threshold of postings has been reached (see "Daily-Threshold=" in the [List Keyword Reference](#) document). A mailing loop may cause this to happen.
- When an error occurs resulting in LISTSERV being unable to process new postings for the list. LISTSERV always sends a traceback of the error to the list owner along with the notification that the list has been held.

In both cases, the list can be freed by either the list owner or a LISTSERV maintainer sending the command

```
FREE listname PW=yourpassword
```

to LISTSERV. However, note that for any hold, it may be wisest to wait until the LISTSERV maintainer has had a chance to check the server for anything untoward that might be causing the error (e.g., a mailing loop) before freeing the list.



**Note:** An error indicating that the current notebook archive LOG file is open and locked by another process may actually indicate that the archive file is in a directory accessible by anonymous FTP and that someone is in the process of FTPing the current notebook archive, making it impossible for LISTSERV to append the latest posting to the current notebook. This error generally occurs only on systems with FTPable notebook archives; by the time you receive the error, it is usually safe to issue the FREE command to release the list. If, on the other hand, this error persists, you may want to check the server for "zombie" processes that may have the file locked, or set the location parameter of the "Notebook=" keyword to a non-FTPable directory.

### 12.4.2 Manual List Holds

If you need to stop LISTSERV from processing new mail for a list for any reason, simply issue the command

```
HOLD listname PW=yourpassword
```

Then, to free the list, issue the FREE command as noted above.

## 12.5 Controlling the List Digest Feature

List "digests" are provided for those users who prefer to receive (typically) one large, comprehensive posting per digest period that includes all of the list traffic from that period, rather than receiving each post individually as processed by the server.

If "Notebook= Yes...", the digest feature defaults to daily digests cut at midnight with the "work" files kept in the same directory as the list's notebook archives. If "Notebook= No", digests are not enabled by default. However, note that lists without notebook archives can have digests; it is simply necessary to enable digests manually for such lists by using the "Digest=" list header keyword and specifying a valid path for the location of the digest's work files.

Digests can be set up to cut on a Daily, Weekly, or Monthly basis, and can be further configured to cut after a certain number of lines of data have been stored regardless of the digest period setting. This ability to generate "special issues" when the digest reaches a certain size may be necessary if people complain that your 10,000 line daily digest is getting truncated by their mail host to 1500 (or even 1000) lines.

For example, if a high volume list is set for Daily digests, and the "Size(nnn)" parameter of the "Digest=" keyword for the list is set to "Size(1000)", a "special issue" of the digest will be cut and mailed to digest subscribers whenever the listname.DIGEST file reaches 1000 lines of text. Note that LISTSERV will not cut the digest at exactly 1000 lines, thereby truncating the last message; LISTSERV will cut the digest after the end of the message that causes the digest file to go over its limit. Thus, if the digest file is 950 lines long and a 200 line message is received, the "special issue" digest will be 1150 lines long.

## 12.6 Defining List Topics

List topics provide powerful "sub-list" capabilities to a list. When properly set up and used, topics give subscribers the ability to receive list postings in a selective manner, based on the beginning of the "Subject:" line of the mail header. It is important to note the following points about topics:

- Topics are best employed on moderated lists. This makes it possible to review the "Subject:" header line to make sure that it conforms to one or more of the topics defined for the list before you forward the post to the list. Not only does this help catch simple errors (such as misspellings of the topic), but it also allows the moderator to add a topic into the subject line if one is not already there.
- If you employ topics on unmoderated lists, your subscribers must be well-educated in their use. Otherwise, there is no point in using them. Messages that do not conform to a specified topic are lumped into the reserved topic "Other" and are distributed only to subscribers who have explicitly defined "Other" as a topic they wish to receive. Therefore some subscribers will receive the message and some won't, and it is problematic as to whether the message will actually reach the entire audience for which it is intended.

The basic keyword syntax for defining list topics in the list header file is:

```
* Topics= topic1,topic2,...topic11
```

And the basic syntax used to set topics for users once they have been defined is:

```
SET listname TOPICS: xxx yyy zzz for userid@host
```

where *xxx*, *yyy*, and *zzz* can be:

- A list of all the topics the subscriber wishes to receive. In that case these topics replace any other topics the subscriber may have subscribed to before. For instance, after 'SET XYZ-L TOPICS: NEWS BENCH', the subscriber will receive news and benchmarks, and nothing else.
- Updates to the list of topics the subscriber currently receives. A plus sign indicates a topic that should be added, a minus sign requests the removal of a topic. For instance, "SET XYZ-L TOPICS: +NEWS -BENCH" adds news and removes benchmarks. If a topic name is given without a + or - sign, + is assumed: "SET XYZ-L TOPICS: +NEWS BENCH" adds news and benchmarks. The first topic name must have the plus sign to show that this is an addition, and not a replacement.
- A combination of the above, mostly useful to enable all but a few topics: "SET XYZ-L TOPICS: ALL -MEETINGS".

The colon after the keyword TOPICS: is optional, and TOPICS= is also accepted. The subscriber should not forget to include the special OTHER topic if you want to receive general discussions which were not labeled properly. On the other hand, if the subscriber only wants to receive properly labeled messages it should not be included. ALL does include OTHER.

Finally, it is important to note that topics are active only when the subscriber's subscription is set to MAIL. Digests and indexes always contain all the postings that were made, because the same digest is prepared and sent to all the subscribers.

With the "Default-Topics=" keyword, you can also set default topics for users that will be effective as soon as they subscribe to the list. For instance,

```
* Default-Topics= NEWS,BENCH,OTHER
```

would set the new user to receive topics NEWS, BENCHmarks, and any messages that are incorrectly labeled.

See the [List Owner's Manual](#) and the [List Keyword Reference](#) document for more information on setting up and using list topics.

## 12.7 Allowing/Blocking MIME Attachments

LISTSERV includes a MIME attachment-filtering feature which is configured by setting the Attachments= list header keyword. The keyword as first introduced allowed three distinct modes:

- Allow all MIME attachments, no filtering or blocking
- Reject MIME attachments with notice to the poster
- Filter MIME attachments out of messages transparently

In addition, you could configure specific MIME types to reject or filter while allowing other types through (for instance, you could block executable files but allow images or word processing files based on their MIME type).

In addition, the Attachments= keyword supports adding a filter for non-MIME, inline uuencoded files such as are sent by mail clients like Microsoft Outlook. The uuencode filter is strictly on/off; no attempt is made to determine the file type of such inline "attachments".

For information on the various settings, please see the section on the Attachments= keyword in the [List Keyword Reference](#) document.

## Section 13 Error Handling Features and Functions

### 13.1 Defining List-Level Error Handling Addresses

Every LISTSERV mailing list requires that an email address be defined to which all mail delivery errors are sent for disposition. The error handling address is defined by using the "Errors-To=" list header keyword.

The value for "Errors-To=" can be one of two things:

- An appropriate access-level, such as `OWNER`
- A specific internet-address, such as `someuser@somehost.com`

It is strongly recommended that "Errors-To=" point to a real person's mailbox, rather than to an alias that simply dumps errors into something like `/dev/null`. Mail delivery errors generally indicate that a problem exists, and it's always possible to filter out the ones that don't via procmail or by using the filtering features available in most POP mail clients. After a long enough period of time, unhandled errors can grow to a significant percentage of your server's traffic and seriously impact your production.

The internet address of the list is explicitly disallowed as an error-receiving address, and attempting to set Errors-To= to the internet address of the list will raise an error. The list should never be configured to receive its own errors as this is guaranteed to cause looping.

If not defined in the list header, "Errors-To=" defaults to "Errors-To= Owners".

It should be carefully noted that there is no way to automatically discard errors without sending them to some address. "Errors-To= No" and "Errors-To= None" are both invalid settings and will cause LISTSERV to revert to the default.

### 13.2 The Auto-Deletion Feature

LISTSERV includes a powerful auto-deletion filter that can be configured in several modes, depending on the level of error handling desired by the list owner. Currently, LISTSERV understands and can take action on errors generated by all mailers compliant with the so-called "Notary" format described in RFC1893. As more and more mailers conform to RFC1893, LISTSERV naturally will be capable of handling more and more errors intelligently.

To set up auto-deletion defaults for a list, use the syntax described in the [List Keyword Reference](#) document for the "Auto-Delete=" list header keyword.

A sample error monitoring report, generated daily and sent to the list's "Errors-To=" address if "Auto-Delete=" is activated, is shown below:

Figure 13-1 A Typical Daily Error Monitoring Report

```

Date:      Thu, 30 Oct 1998 00:00:48 -0400
From: "L-Soft list server at PEACH.EASE.LSOFT.COM (1.8d)"
          <LISTSERV@PEACH.EASE.LSOFT.COM>
Subject:   EXCEL-G: Daily error monitoring report
To:       EXCGERR@LINUS.DC.LSOFT.COM
The following 3 subscribers were deleted from the EXCEL-G list today:
  sluggo@OMNI.VOICENET.COM
  Last error was: Mailer quasar.voicenet.com said: "550
          <sluggo@OMNI.VOICENET.COM>... User unknown"
  crenaud@BOSS1.BOSSNT.COM
  Last error was: Mailer BOSS1.BOSSNT.COM said: "550
          <crenaud@BOSS1.BOSSNT.COM>... User unknown"
  Roger Giellis <rogerg@SUPSUN4.DEN.MMC.COM>
  Last error was: Domain "SUPSUN4.DEN.MMC.COM" doesn't exist.
The following 5 subscribers are currently being monitored:
Err First Last  Address
--- -----
  2 05/28 05/29 "Ronald D. Stepp" <MORITURI@INTELLISYS.NET>
          Last error: Mailer INTELLISYS.NET said: "550
          <morituri@INTELLISYS.NET>... User unknown"
  1 05/29 05/29 TEST@TEST.POWERNET.CO.UK
          Last error: Domain "TEST.POWERNET.CO.UK" doesn't exist.
  1 05/29 05/29 SIMONC@VOL.NET
          Last error: Mailer h02.VOL.NET said: "550 <simonc@VOL.NET>...
          User unknown"
  1 05/29 05/29 REG@NROBBO.CO.UK
          Last error: Domain "NROBBO.CO.UK" doesn't exist.
  2 05/29 05/29 jmanring@INETGW.LEGGMASON.COM
          Last error: Unavailable; notary status was 5.1.2
Err=   Number of delivery errors received thus far
First= Date first delivery error was received (mm/dd)
Last=  Date of most current delivery error (mm/dd)
Subscribers will be automatically deleted from the list when delivery errors
have been reported for a period of 2 days or more, or when 10 delivery
errors have been received, whichever occurs first. Monitoring will cease
after 3 days without any reported error.
Note: manually deleted subscribers may remain on the monitoring report under
an alias address. Such entries will expire eventually; you do not need to do
anything about them.

```

### 13.3 LISTSERV's Loop Detection Feature

LISTSERV has an extremely advanced loop detection heuristic that practically eliminates the chances of a mailing loop being propagated through one of its mailing lists. In general, L-Soft does not recommend that any loop-checking element be disabled, as any

one missing element might let a loop through, but under certain controlled circumstances it might be needful to do this. See the [List Keyword Reference](#) document under "Loopcheck=" for specific keyword options to turn off parts of the loop-checking feature.

### 13.3.1 The Anti-Spamming Filter

LISTSERV's anti-spamming filter is built into the loop-checking heuristic. Depending on your local circumstances, it may be desirable to disable the anti-spamming filter for certain lists, particularly if the lists are confidential (and thus not visible in the global list of lists, making it extremely unlikely that a spammer would target them), if the lists are set to reject or transfer to the list moderator postings from non-subscribers (e.g., "Send= Private" or "Send= Editor"), or if your LISERV server is not accessible from the Internet (e.g., you're running it on an internal LAN without Internet connectivity).

If you need to turn the anti-spamming filter off for a particular list, code:

```
* Loopcheck= NoSpam
```

One (tunable) aspect of the anti-spamming filter involves holding mail from non-subscribers for a pre-determined length of time (the default is 10 minutes) to see if further mail arrives from the same user for other lists that may trigger the anti-spamming filter. If you want anti-spamming protection, but want mail from non-subscribers to go directly to a list without being held up for this check, you can code

```
* Loopcheck= Spam-Delay(0)
```

If you want the check to be performed, but think 10 minutes is too long to hold the messages, you can change the value for "Spam-Delay()" to the preferred number of minutes. For instance, to hold non-subscriber messages for five minutes, code

```
* Loopcheck= Spam-Delay(5)
```



**Note:** You can configure the server-wide default for this "spam quarantine" feature by adding the SPAM\_DELAY variable to your site configuration file and specifying the number of minutes. For instance, setting the variable to a value of 15 would set the server-wide spam quarantine default to 15 minutes, while setting it to zero disables the feature.

You can avoid the problem of a legitimate user being identified as a spammer when cross-posting to multiple lists on your server by setting up a super-list that has as its sub-lists the lists to which the user needs to cross-post. This is the only supported method for cross-posting to multiple lists if the anti-spam feature is not disabled for all the lists in question.



**Note:** There is no command or method to lift the 48-hour anti-spam quarantine for addresses that are identified as spammers and blocked from posting.

The rest of the anti-spamming filter algorithm is proprietary and non-configurable.

## 13.4 RFC822 Mail Header Parsing

LISTSERV is designed to be 100% compatible with the Internet RFCs that govern how mail headers may be formatted (RFC822 et seq.), and includes a powerful RFC822 parser for this purpose. However, not all individual mail clients are compliant with RFC822, either because of poor design or because of mis-configuration. Because of this, LISERV postmasters may from time to time see bounces such as the example below:



Figure 13-2 Sample RFC822 Parser Error

```

Date: Tue, 5 Aug 1997 15:43:57 -0400
From: "L-Soft list server at Apple (1.8c)" <LISTSERV@MAIL.EWORLD.COM>
To: Nathan Brindle <nathan@OSF1.DC.EXAMPLE.COM>, ERIC@VM.SE.EXAMPLE.COM
Subject: Problem processing mail file from MAILER@MAIL.EWORLD.COM
An error occurred while processing file 1449650 from MAILER@MAIL.EWORLD.COM:
"Mail message sent to the LISTSERV address contains invalid RFC822 fields and
could not be parsed successfully".
RFC822 parser messages follow:
<W> Incorrect or incomplete address field found and ignored.
<W> Incorrect or incomplete address field found and ignored.
<W> Incorrect or incomplete address field found and ignored.
<E> Mail origin cannot be determined.
<E> Original tag data was -> "myuserid" <>
----- Message causing the problem (29 lines) -----
Received: from dfw-ix16.ix.netcom.com by home.ease.lsoft.com (LSMTP for
Windows
NT v1.1a) with SMTP id <0.628849B0@home.ease.lsoft.com>; Tue, 5 Aug 1997
15:43:56 -0400
Received: (from smap@localhost)
        by dfw-ix16.ix.netcom.com (8.8.4/8.8.4)
        id OAA06085 for <listserv@mail.eworld.com>; Tue, 5 Aug 1997 14:45:04
-0500 (CDT)
Message-Id: <199708051945.OAA06085@dfw-ix16.ix.netcom.com>
Received: from por-or10-22.ix.netcom.com(204.31.113.150) by
dfw-ix16.ix.netcom.com via smap (V1.3)
        id smaa06017; Tue Aug 5 14:44:47 1997
From: "myuserid" <>
To: <listserv@mail.eworld.com>
Date: Tue, 5 Aug 1997 12:47:20 -0600
...

```

The lines starting with "<W>" are warnings from the parser that indicate a non-fatal problem with one or more of the message's RFC822 headers. In this case the warnings apply to the second "Received:" header of the message, which is misformatted (it includes end-of-line characters and thus the parser treats the header as three separate RFC822 headers and attempts to parse them individually). If these were the only warnings, the message would still be accepted by LISTSERV.

However, there is a second problem in the message, and this one is fatal. The RFC822 "From:" header has a null value for the user's "userid@host" address. It is most likely that this user has decided to remove his address from his POP client's configuration in order to avoid being placed on spammers' mailing lists. However, this is not legal per RFC822, and when LISTSERV tries to determine the origin of the mail message, it can't be done. This is because the value for "From:" is invalid and there is no other header (such as "Sender:") that might be able to indicate where the message is coming from. Therefore LISTSERV writes two "<E>" (error) lines, one that says the mail origin can't be

determined and a second to specify what the data was in the one origin header it could find, and bounces the message to the LISTSERV maintainer(s) for further disposition.

Another error you might see is

```
<W> MESSAGE-ID field duplicated. Last occurrence was retained.
```



**Note:** This error commonly occurs when the inbound mail was generated with Eudora. Eudora versions 4 through 7.0.1.0 famously (and erroneously) generate duplicate Message-ID fields when using its "Send Again" feature to generate emails. The "last" commercial version of Eudora, 7.1.0.9, has finally addressed this problem.

## 13.5 Address Probing

There are two levels of automatic address probing available in LISTSERV.

### 13.5.1 Active Address Probing

*This functionality is not available in LISTSERV Lite.*

Active address probing is available for two reasons: first, to enhance subscription renewal functionality so that no "CONFIRM *listname*" response was required from subscribers in order to stay subscribed, and second, to enhance the ability of the auto-deletion feature to handle bounces that can't be parsed into something LISTSERV can recognize.

"Renewal= . . . ,Probe" activates this enhanced bounce processing feature, whereby subscribers are probed at subscription renewal time using the PROBE1 mail template. The "Probe" option makes subscription renewal passive rather than reactive; no "CONFIRM *listname*" response is needed from the user. In fact, the desired response from the user is to discard the message and do nothing, making the process very simple. LISTSERV also probes addresses that return mail delivery errors, and probe messages have a special signature in the return address that allows LISTSERV to uniquely identify any bouncing address, without having to understand the bounce itself.

If the probe bounces, LISTSERV first sends the PROBE2 template with a copy of the bounce, to show the user (if the account actually works in spite of the bounce) what garbage his mail system is sending people. LISTSERV then schedules a new probe for the next day, or deletes the user immediately, depending on the auto-delete policy. Every failure triggers a new daily probe until the user gets deleted or the problem gets fixed. The user can also save his subscription manually by sending a CONFIRM *listname* command (this is explained in PROBE2). This doesn't solve the underlying problem, so eventually the user should get tired of confirming in an emergency and notify his system administrators that the system is generating bounces saying (for instance) "Your message was registered at the MORONICUS mail gateway. Press F1 for more information" that cause the problem in the first place.

When used together with "Auto-Delete= . . . ,Full-Auto", the probe option deletes all delivery errors from bounced probes, even if LISTSERV can't understand the error. This means the list owner never ever has to see a single bounce from a probed address. The list, however, is kept clean because bad addresses are always detected. In fact, the biggest risk is that the users of the MORONICUS mail gateway will be deleted even though they do get their mail.

This being said, note carefully that all errors bounced by non-compliant mail hosts to the wrong address, and non-probe errors that are sent to the owner-listname address but are not in a format that LISTSERV can parse, will still show up in your error mailbox. If the bounce goes to the wrong address, LISTSERV never sees it and cannot probe it. If the error goes to the correct address (owner-listname) but isn't specific enough for LISTSERV to understand, while LISTSERV will be able to see it, it still won't be able to probe it. Finally, in some cases where the error is so vague (or constructed in a complicated manner that defies LISTSERV's attempts to parse it) the error may be passed to the LISTSERV postmaster, instead of to the list owner, for disposition, even if it was correctly returned to the owner-listname address.

Yet even with these restrictions, the author saw an error queue of 1300 errors/day shrink to under 50 errors/day by applying the ",Probe" parameter to seven high-volume lists, which in his opinion was much more acceptable.

If you have users who for whatever reason should not be probed, you can deactivate active probing (and any other renewal you have set for the list) with the `SET userid@host NORENEW` command.

### 13.5.2 Passive Address Probing

*This functionality is not available in LISTSERV Lite.*

Passive probing is very similar to active probing, but it is not tied to subscription renewal. Passive probing is enabled by default for small lists (e.g., <1K subscribers) but not for large ones due to the fact that passive probing does cost additional resources and large lists are often used for one-shot mailings where it is simply not effective to use those resources to probe addresses that will not be used a second time.

Passive probing operates by turning a certain percentage of your regular list messages into transparent probes that look like a normal message but also double as a probe, rather than sending out the explicit PROBE1 template as in active probing. You enable (or tune) passive probing by adding a ", Probe (xx) " parameter to the `Auto-Delete=` keyword setting. For instance,

```
Auto-Delete= Yes,Full-Auto,Probe(30)
```

where "30" is the number of days to wait between probes for any given user. Subscribers with working mail systems will not see any difference, subscribers with flaky mail systems will occasionally receive a message showing that their mail bounced and saying that they should report the problem to their ISP, and of course plain bad addresses will go away.

In order to disable passive probing you set the probe parameter to 0, i.e.,

```
Auto-Delete= Yes,Full-Auto,Probe(0)
```

If you have users who for whatever reason should not be probed, you can deactivate passive probing (and any other renewal you have set for the list) with the `SET userid@host NORENEW` command.

If a given list only has activity once in a while (e.g., a large weekly newsletter), passive probing works like this: If you have `Probe(p)` set in your `Auto-Delete=` keyword (where `p` is some integer value), and you have `n` subscribers, about  $(n / p)$  will receive a probe during the mailing. Normally you would want to probe 2-10% of your subscribers in this kind of scenario, so `p` would range from 10 to 50.



**Note:** LISTSERV ignores Probe(0) in list-based mail-merge jobs. Mail-merge messages are always sent as probes, and in list-based mail-merge there is no attempt made to parse the header of the list that is being used as a datastore for the mail-merge job.

### 13.5.3 OS-Specific Issues with Probing

Probing is supported automatically by the VM and Windows versions of LISTSERV without need for any special configuration other than noted above.

**On unix systems**, while LISTSERV itself does support probes, probes are not supported natively by most (if not all) unix MTAs, including sendmail, etc. In order to use probing on such systems the MTA must be patched to divert incoming probe bounces to the lsv\_amin mailer for delivery to LISTSERV. Otherwise incoming probes simply bounce since there is no way for the MTA to determine what to do with them.

User-contributed patches to support probing under sendmail are available on L-Soft's ftp site but are not supported by L-Soft--use at your own risk!

**On VMS systems**, probing is supported natively if you are using L-Soft's legacy LSMTP mailer version 1.1a or later, or any version of MX that includes support for the LISTSERV interface. PMDF® users should create a dedicated domain for LISTSERV (e.g., LISTSERV.XYZ.COM) and add a rewrite rule to redirect all traffic for that host to the LSV channel. This also simplifies the creation of new lists as with this setup, it is no longer necessary to define PMDF aliases for the lists.

## 13.6 Defining Server-Level Error Handling Addresses

### 13.6.1 BOUNCES\_TO=

It is possible to divert some of the server-level error traffic (that is to say, error traffic not specific to a given list, or errors that should never have been sent to LISTSERV to begin with) to a specified place other than the default (the non-quiet LISTSERV maintainers). This is done by adding the BOUNCES\_TO= variable to your site configuration file and restarting LISTSERV.

The following specific types of traffic are routed to the BOUNCES\_TO= address: spam alerts, spoofing alerts, quota errors (for sites running with the SCOPE=ISP option), and DISTRIBUTE error messages.

BOUNCES\_TO= can point to one or more users (use the same syntax as for POSTMASTER=) or to a mailing list set up for the purpose.

### 13.6.2 Crash Reports and CRASH\_MONITOR=

Following a severe system crash, LISTSERV running under VMS and Windows NT will generate a "crash report" containing:

- System-specific information about the immediate cause of the crash (access violation, division by zero, etc.).
- A traceback showing what LISTSERV was doing at the time of the crash.
- The last 100 lines in the LISTSERV log.

By default, this report is mailed to the LISTSERV maintainer. You can change the destination of these reports by adding a `CRASH_MONITOR` variable to your configuration file (`SITE.CFG` for NT, `SITE_CONFIG.DAT` for VMS) with the email addresses to which the report should be mailed. Note that `CRASH_MONITOR` replaces the entire recipient list, so make sure that all the necessary addresses are listed. This configuration variable follows the same syntax rules as `POSTMASTER`. Please do not add L-Soft mailboxes to `CRASH_MONITOR` without checking with our support group first. While we will be happy to receive these reports, we want to make sure that they are sent to the addresses where we can process them most efficiently. In particular, these reports should never be mailed to a support engineer's personal mailbox. Instead, we use special addresses where these reports are logged for future reference.



**Important:** Crash reports may contain company confidential information! Before forwarding a crash report to L-Soft, make sure that it does not contain any confidential information. L-Soft will not sign non-disclosure agreements related to crash reports. If you include an L-Soft address in your `CRASH_MONITOR` configuration variable, you are implicitly stating that none of the activity taking place on your server is confidential.

When reporting a crash to L-Soft, please forward a copy of the crash report with any confidential information removed or XXX-ed out. Note that the crash report is not actually mailed until LISTSERV is restarted. Crashes are usually caused by conditions which prevent LISTSERV from operating normally; furthermore, image termination may be necessary to cause the operating system to generate the traceback included in the report.



**Important (Windows Only):** In order for the crash report to be useful, the files `LSV.EXE` and `LSV.SYM` must be updated at the same time. This is done automatically if you install/update LISTSERV using the graphical installation procedure, however if you install patches manually you must ensure that both `LSV.EXE` and `LSV.SYM` are updated.

This feature was not provided for VM because all the information that is available is already gathered at the bottom of the console log, which is normally spooled to a maintenance userid and not accessible to LISTSERV.

Under unix, there is no portable way for an exception handler to obtain a call traceback; a system-specific debugger (which must be installed separately and often requires a separate license) must be run on the core file, which is not available until the process has aborted. LISTSERV does flush buffered log output to ensure that the `listserv.log` file contains all the relevant log information following a core dump.

## Section 14 List Maintenance and Moderation Features and Functions

### 14.1 Setting Up Edited/Moderated Mailing Lists

As noted in Section 7.13 [Setting Up Lists for Specific Purposes](#), you need only add the following lines to the list header file:

```
* Send= Editor
* Editor= userid@some.host.edu
```

where "userid@some.host.edu" should be replaced with the network address of the person who will be handling submissions to your list.

There can be multiple editors as well (and multiple Editor= lines, if desirable), and they do not have to be list owners:

```
* Send= Editor
* Editor= alex@reges.org, joe@foo.bar.edu
* Editor= tony@tiger.com
```

Normally, LISTSERV forwards submissions only to the first editor defined by the "Editor=" keyword. In the case above, all submissions would go to the primary list owner.



**Note:** The first editor CANNOT be an access-level; that is to say, you cannot use the notation "Editor= Owner" to define the first editor. LISTSERV requires that the primary editor of a list must be the e-mail address of a real person.

This does not apply to second and subsequent editors. For instance, in order to allow subscribers to post directly but have non-subscriber posts sent to an editor for approval, you can code something like:

```
* Send= Editor
* Editor= alex@reges.org, (MYLIST-L)
```

On a high-volume list, LISTSERV allows you to share the editing load via the "Moderator=" keyword. By default, this keyword is set to the same value as the first editor defined by "Editor=". When you define more network addresses with the "Moderator=" keyword, LISTSERV sends submissions to each moderator in sequence. The difference between the "Editor=" and "Moderator=" keywords lies in the fact that while any editor can post directly to the list, only moderators receive the forwarded submissions from non-editors.

Here is an example of a list with both Editor= and Moderator= keywords defined:

```
* Send= Editor
* Editor= joe@foo.bar.edu, tony@tiger.com, kent@net.police.net
* Moderator= kent@net.police.net, joe@foo.bar.edu
```

This list will "load-share" the editing duties between Kent and Joe. Tony is able to post directly to the list, but will not receive forwarded subscriber posts for editing.



**Note:** An Editor is not required to be a Moderator, but a Moderator should always be listed as an Editor. LISTSERV currently compares the contents of the "Editor=" and "Moderator=" keywords and consolidates the two sets of parameters if necessary, but coding lists this way is not considered good practice and the "compare/consolidate" feature may be removed in a future upgrade.



For more information on setting up edited lists, see "Send=", "Editor=" and "Moderator=" in the [List Keyword Reference](#) document, as well as Section 7.13.2 [Private Discussion Lists](#) where setting up edited lists is discussed further.

## 14.2 Restricting the Size of Messages Posted to the List

Using the "Sizelim=" list header keyword, you can restrict the size (in lines) of messages posted to a given list. This may be particularly desirable for lists discussing programming topics where the posting of uuencoded binaries to the list is discouraged, or simply to encourage economy in posting. In any case, if this feature is desired, simply add the keyword

\* Sizelim= nnn

to the list header, where "nnn" is the maximum number of lines (including Internet delivery and addressing headers) to be accepted. Note that unlike the "Size()" parameter of the "Digest=" keyword, LISTSERV will not allow a post to go over the "Sizelim=" setting, but will reject it if it is even a single line over the allowable threshold. When a posting is rejected for size, the original poster receives a notification that his post was too large.

Additionally, list owners may specify a maximum message size in either kilobytes or megabytes, rather than in lines, if preferred. For instance:

- `Sizelim= 100K`  
Reject messages over 100Kbytes
- `Sizelim= 1M`  
Reject messages over 1Mbyte

As before, the limit operates against the entire message file, including all Internet header lines.



**Note:** Some misconfigured mail hosts will try to bounce delivery errors, complete with the text of the message that bounced, back to the list address rather than to the RFC821 MAIL FROM: address. Setting "Sizelim=" to a reasonable level (say, 400 lines, or 25-30 kilobytes) will usually prevent a mail host from bouncing a whole digest back to the list.

## 14.3 Restricting the Number of Posts Per User Per Day

You can restrict the number of posts to the list per user per day. This is done with a new second parameter for the "Daily-Threshold=" list header keyword. For instance, setting "Daily-Threshold= 100,5" would tell LISTSERV to hold the list after 100 postings, and additionally, to stop accepting new postings from any individual subscriber after that subscriber had posted 5 messages during the 24h period from midnight to midnight (server time). After reaching the user threshold, the subscriber simply receives a message to the effect that he has reached the daily limit and that he should try to repost the message later (i.e., after midnight). Please see the entry for "Daily-Threshold=" in the [List Keyword Reference](#) document for further information.



### 14.4 Moving a List to a New Location (New-List= Keyword)

When a list is moved to a different LISTSERV host, this keyword can be added to the list header left on the original host. This facilitates forwarding of administrative commands and postings from the original host to the new host. Users posting to the old address will also receive a short note in return listing the new address.

**Notes:** This only works for a move from one L-Soft LISTSERV server to another, not for a move from a LISTSERV server to a server running another mailing list manager.

See "New-List=" in the [List Keyword Reference](#) document for more information about this keyword and how to use it.



## Section 15 Security Features and Functions

LISTSERV's security options are wide ranging, from almost no protection (easiest to administer a list, but also most open to hacker attacks) to total protection requiring validation of each and every command sent to LISYSERV for the list. It is also possible to limit access to various aspects of the list, such as who can subscribe, who can review the list of subscribers, and who can access the list archives. The list can be hidden from the LIST command, either at the global level or from all requests, including those from users on LISYSERV's local machine, or from a definable range in between.



**Note:** LISYSERV does not set any file system permissions for any files. LISYSERV's security features are meant to allow and deny access to LISYSERV's various files depending on how various keywords are set, but in order to make your system completely safe, you must ensure that you have also set file and directory permissions appropriately for your operating system. For instance, LISYSERV may deny GET access to certain list archive files to non-subscribers, but if you have not set the appropriate file system permissions at the operating system level, you may have left open a window for someone to reach these files via anonymous ftp.

### 15.1 The VALIDATE= Keyword

The `VALIDATE=` keyword controls the level of command validation desired for the list. The default, `VALIDATE= NO`, requires password validation only for storing the list on the server. This is often sufficient for general needs. However, when a list is set this way, LISYSERV only compares the RFC822 "Sender:"/"From:" headers against the `Owner=` keyword(s) in the list header to determine whether or not the person ostensibly sending the commands has authority to do so. Otherwise at this level LISYSERV does not validate commands it receives for the list, under the assumption that the mail it receives is genuinely coming from a list owner. This level of validation does not protect the list from commands issued by hackers who have forged mail in the name of the list owner. If you run a list on a controversial topic or just don't feel comfortable without at least some security, `VALIDATE= NO` is probably not for you.

The next level is `VALIDATE= YES`. At this level, LISYSERV requires a password for all of its "protected" commands. This password is the sender's personal LISYSERV password as defined by the `PW ADD` command. The commands protected by this level are those that affect subscriptions or the operation of the list, for example, `DELETE` or `ADD`. Users will also have to validate most commands that affect their subscriptions, but generally can do so using the "OK" mechanism rather than defining a personal password. Note that some user commands will be forwarded to the list owner for validation rather than accepting password validation from the user.

The next level is `VALIDATE= YES, CONFIRM`. At this level, LISYSERV will require validation with the "OK" mechanism (see below) by default, but will still accept passwords where appropriate. While the less-secure passwords are still accepted, this is considered a good compromise between list security and list owner and user convenience.

The next level is `VALIDATE= YES, CONFIRM, NOPW`. At this level, LISYSERV will no longer accept passwords as validation for protected commands. The logic is that because of the way the "OK" mechanism is implemented, passwords are not as safe as "magic cookies". This is the recommended setting for lists that must be kept secure.

Two other levels are `VALIDATE= ALL, CONFIRM` and `VALIDATE= ALL, CONFIRM, NOPW`. These levels require "OK" validation for all commands that cause a change in state except for the `PUT` command. If `NOPW` is not specified, passwords are accepted where appropriate. With these levels, commands that do not cause a change in state (e.g., `QUERY` and other strictly-informational commands) do not require validation.



**Note:** `LISTSERV` requests coming from the local system via `CP MSG` or `CP SMSG` on VM systems or via `LCMD` on VMS or Unix systems never require validation, as they cannot be forged.

Lists which are set to either `Validate= Yes, Confirm, NoPW` or `Validate= All, Confirm, NoPW` may not be managed via the web administration interface, which is password-driven.

See the [List Keyword Reference](#) document for complete information on the `VALIDATE=` keyword.

## 15.2 Controlling Subscription Requests

Subscription requests are controlled by use of the `SUBSCRIPTION=` keyword. By default, this keyword is set to `SUBSCRIPTION= BY_OWNER`, meaning that all subscription requests will be forwarded to the list owner for disposition. Subscription requests can be refused completely by setting `SUBSCRIPTION= CLOSED`.

To code a list for open subscriptions without list owner intervention, set `SUBSCRIPTION= OPEN`. If it is desired to add protection against forged subscription requests or bad return mailing paths, code `SUBSCRIPTION= OPEN, CONFIRM`. The latter will cause a subscription confirmation request to be sent to the prospective subscriber, which he or she must respond to using the "OK" confirmation mechanism.

In order to restrict subscriptions to persons in a specific service area, see the next section.

## 15.3 Controlling the Service Area of the List

*The `Service=` keyword is not available in `LISTSERV Lite`.*

It may be desirable to restrict access to a list to people in a small area. For instance, you probably would not want a list for students in a class section at a university to be advertised or accessible by people all over the world. However, without setting certain keywords appropriately, such a list will be visible to a `LISTS GLOBAL` command.

The local list of (public) lists can be retrieved only by those users who are considered local, per the setting of the server-wide `LOCAL=` variable in `LISTSERV`'s site configuration file. All other users will be told that none of the lists on the server are visible via the `LISTS` command, and will be referred to the use of the `LISTS GLOBAL search-text` command or to the `Catalist`. This is regardless of the setting of `Confidential=` as outlined below.

To simply hide a list from a `LISTS` command, but still allow people to subscribe to it if they know it is there, use the keyword `Confidential= YES`. Note that users subscribed to the list as well as the list owner(s) will be able to see the list if they issue a `LISTS` command. In addition, all other non-subscribers, including users on the local machine, will not be able to determine that the list exists via a `LISTS` command.

To hide a list from and refuse subscription requests from users outside the local area, you define two keywords:

```
* Service= bitnode1,bitnode2,some.host.edu
* Confidential= SERVICE
```

`Service=` can also be set to `Service= LOCAL`, meaning it will use either LISTSERV's global definition of which machines are `LOCAL`, or the machines defined by the list keyword `Local=`. The LISTSERV maintainer should define hosts and nodes that are considered local with the server-wide `LOCAL=` variable in the site configuration file. If the global definition is not suitable, it can be overridden by defining the `Local=` list header keyword:

```
* LOCAL= bitnode1,bitnode2,some.host.edu,another.host.co
m* SERVICE= LOCAL
* CONFIDENTIAL= SERVICE
```

If there are many subdomains within your primary domain, it may be preferable to use the wildcard when defining the `LOCAL` or `SERVICE` list header keywords. For instance:

```
* SERVICE= *.HOST.COM
```

defines the service area for a specific list as "all subdomains ending in `.HOST.COM`".



**Note:** Defining a service area for a list controls only from which domains subscription requests may be accepted. It does not control who may post to the list. Depending on local circumstances, it may be desirable to set lists with controlled service areas to `Confidential= Service`.

## 15.4 Controlling Who Reviews the List of Subscribers

For whatever reason, it may be desirable to restrict the ability to review the subscriber list either to subscribers or to list owners. This is done by setting the `REVIEW=` keyword appropriately.

To allow anyone, including non-subscribers, to review the list, set `REVIEW= PUBLIC`.

To restrict reviews of the list to subscribers only, set `REVIEW= PRIVATE`. This is the default.

To restrict reviews of the list to list owners only, set `REVIEW= OWNERS`.

Reviews can also be restricted to users within the list's service area by setting `REVIEW= SERVICE`, and defining the `SERVICE=` keyword appropriately (see the preceding section).



**Note:** Unless the list is set to `"Confidential= Yes"` or `"Confidential= Service"`, a request to `REVIEW` a list by someone who is not allowed to do so will result in the header of the list being sent to the user along with a note to the effect that the `REVIEW` command is restricted for this list. See Section 15.9 [Hiding Selected Header Lines](#) regarding hiding header lines if you want to hide parts of the header but do not want to use the `"Confidential="` keyword.

## 15.5 Controlling Access to the Notebook Files

Restricting access to the list's notebook archive files is similar to controlling who may review the list. It is accomplished by setting the fourth parameter of the `NOTEBOOK=` keyword to an appropriate value. For instance,

```
* NOTEBOOK= Yes,A,Monthly,Public
```

defines a monthly notebook on LISTSERV's A disk that is accessible by anyone. Change Public to Private if you wish only subscribers to be able to access the notebooks. The same access-levels are available for this keyword as for REVIEW=. (See the [List Keyword Reference](#) document for a discussion of access-levels.)

It is possible to define "Service=" in terms of IP address blocks in order to limit access to list archive notebooks. See "Service=" in the [List Keyword Reference](#) document for details.

If enabled, notebook archives are private by default.

## 15.6 Controlling Who Can Post Mail to a List

The Send= list header keyword is the basic control for who may post mail to the list. If the list allows non-subscribers to post, set Send= Public. (This is the default.)

For a list that does not allow non-subscribers to post, set Send= Private.

For a list where all posts should be forwarded to a moderator/editor, there are two settings:

- Send= Editor forwards all postings to the list editor (see the Editor= and Moderator= keywords). This setting allows the editor to make changes before forwarding the message back to the list. Note that your mail program must be capable of inserting "Resent-" header lines in your forwarded mail—if it is not capable of this, all such posts forwarded to the list will appear to be coming from the editor. Check with your system administrator if you are not sure whether or not your mail program inserts the "Resent-" headers.
- Send= Editor, Hold forwards a copy of the posting to the editor but differs from Send= Editor in that LISTSERV holds the posting for a period of time (usually 7 days) until the editor confirms the message with the "OK" mechanism (see below). Unconfirmed messages simply expire and are flushed by LISTSERV, so there is no need to formally disapprove a posting. This method of message confirmation is well-suited to lists where it is not often necessary to modify the text of a posting, and also is an excellent workaround if the editor's mail program does not generate "Resent-" headers in forwarded mail.

Below is a sample of the editor-header for a list set to Send= Editor, Hold:

*Figure 15-1 The Editor-Header for a List Set to Send= Editor, Hold*

```
Date: Tue, 4 Aug 1998 10:47:21 -0500
From: "L-Soft list server at PEACH.EASE.LSOFT.COM (1.8d)"
      <LISTSERV@PEACH.EASE.LSOFT.COM>
Subject: B5-L: approval required (9723A0DD)
To: Joe ListOwner <joe@PRUNE.EXAMPLE.COM>

This message was originally submitted by jack@UNIX.FOO.COM to the B5-L list
at PEACH.EASE.LSOFT.COM. You can approve it using the "OK" mechanism, ignore
it, or repost an edited copy. The message will expire automatically and you
do not need to do anything if you just want to discard it. Please refer to
the list owner's guide if you are not familiar with the "OK" mechanism;
these instructions are being kept purposefully short for your
convenience in processing large numbers of messages.

----- Original message (ID=9723A0DD) (13 lines) -----
```

A final method (called "self-moderation") exists for lists where subscribers should be allowed to post freely, but non-subscriber posts should always be sent to an editor for approval. To enable self-moderation, set

```
Send= Editor          (or Send= Editor, Hold)
Editor= userid@host, (listname)
```

Ensure that "listname" is in parenthesis. Note that self-moderation will catch all posts from non-subscribers—including posts from subscribers who are posting from a different address. For instance, if the subscriber originally signed up as joe@foo.com but is posting from joe@unix1.foo.com, LISTSERV will treat his mail as non-subscriber mail. Self-moderation may require some slight changes in individual user subscriptions in order for it to work seamlessly. See also the `Default-Options=` list header keyword description.

## 15.7 The "OK" Confirmation Mechanism

Depending on the setting of the `Validate=` list header keyword, certain LISTSERV commands have always required a password for execution. However, with a recognition that mail can be forged ("spoofed") by just about anyone on the Internet today, L-Soft introduced a "magic cookie" method of command validation that is considered much more secure than passwords.

In essence, the "magic cookie" method requires that the sender of the command must confirm his command via a reply containing only the text "OK". (This is actually simplistic; see below.) If mail is spoofed from the list owner's user id, the command confirmation request will always be sent to the list owner's user id, thus preventing the spoofer from confirming the command. Moreover, the "cookie" itself (an eight-digit hexadecimal number) is registered to the "From:" user id of the original command. A typical command confirmation request looks like this:

*Figure 15-2 A Typical Command Confirmation Request*

```
Date:      Wed, 5 Aug 1998 09:50:06 -0400
From:      "L-Soft list server at LISTSERV.EXAMPLE.COM (1.8d)"
           <LISTSERV@LISTSERV.EXAMPLE.COM>
Subject:   Command confirmation request (5C019D91)
To:        joe_user@EXAMPLE.COM

Your command:

                PW REP XXXXXXXX

requires confirmation. To confirm the execution of your command, simply
point your browser to the following URL:
    http://listserv.example.com/scripts/wa.exe?OK=5C019D91

Alternatively, if you have no WWW access, you can reply to the present
message and type "ok" (without the quotes) as the text of your message. Just
the word "ok" - do not retype the command. This procedure will work with any
mail program that fully conforms to the Internet standards for electronic
mail. If you receive an error message, try sending a new message to
LISTSERV@LISTSERV.EXAMPLE.COM (without using the "reply" function - this
is very important) and type "ok 5C019D91" as the text of
your message.

Finally, your command will be cancelled automatically if LISTSERV does not
receive your confirmation within 48h. After that time, you must start over
and resend the command to get a new confirmation code. If you change your mind
and decide that you do NOT want to confirm the command, simply discard the
present message and let the request expire on its own.
```



The general method of replying to a command confirmation request is to use the web browser confirmation method outlined in the confirmation request.

If you prefer, you can use the old method of responding by mail:

- REPLY to the command confirmation request with the text "ok" in the body of the reply. (Non-case-sensitive) LISTSERV reads the "cookie" from the subject line and if it corresponds to a held job, the job is released and processed.

If this does not work, it is possible that the Subject: line was corrupted in transit and you may need to try the following:

- SEND a new message to LISTSERV with the text "ok xxxxxxxx" (where xxxxxxxx is the command confirmation number from the original confirmation request) in the body of the reply.

It is also possible to confirm multiple command confirmation requests with a single message (for instance, if you have `Send= Editor, Hold` and have a number of requests to be responded to). This eliminates multiple "Message approved" mails from LISTSERV. However, make sure that you send the confirmations in a new mail message rather than replying to one of them. (See the "bracketed OK" syntax mentioned below.)

You can send the "OK" from any address, which helps when the address field of your mail gets changed somewhere along the line. For instance if you are logged into the web administration interface as `joe@example.com` and issue a command that requires mail confirmation, LISTSERV will send the request to `joe@example.com` (as expected). If your mail system expands `joe@example.com` to `Joe_Doakes@mail.example.com`, the "OK" will still succeed and `Joe_Doakes@mail.example.com` will get a message that says

```
> ok
Confirming:
> QUIET DELETE * jane@example.com
[reply sent to joe@EXAMPLE.COM]
```

while as a protection against "spoofed" commands the actual command response will be sent to `joe@example.com` like this:

```
jane@EXAMPLE.COM has been removed from the TEST list. No notification
has been sent.
```

```
Global deletion process complete, one entry removed.
```

The "OK" confirmation mechanism also has the following features:

- An "OK" without an argument (confirmation number) flushes the job stream so any text following an "OK" on a line by itself will not be seen by the LISTSERV command processor.
- Bracketed "OK" functionality. This feature allows you to send multiple commands for which LISTSERV will request only a single "OK" (where normally you would expect to have to "OK" each individual command). The syntax is as follows:

```
OK BEGIN
command1
command2
...
command3
OK END
```

- A command confirmation ("OK") may now be sent by clicking on a web URL provided in the command confirmation request (mailed "OK"s are still perfectly acceptable, of course).



**Note:** In a "bracketed OK" the aggregate length of the data stream (that is, the total number of characters in the command lines falling between OK BEGIN and OK END) MUST be less than 32K characters. In practice you should use bracketed OKs for limited numbers of commands only, say no more than 10-12 at a time. In particular, if you have many ADD or DELETE commands to send, it is far more efficient (and strongly preferred) to use the bulk ADD and bulk DELETE syntaxes described in Section 7.17 [Bulk Operations \(ADD and DELETE\)](#).

### 15.7.1 Explicitly Cancelling "OK" Cookies

It is possible to explicitly cancel an OK confirmation cookie if so desired. The command is simply

```
OK CANCEL xxxxxxxxxx
```

(for instance, "OK CANCEL 8F2E8F4B"), and if the cookie is valid, LISTSERV will respond "Confirmation code 8F2E8F4B cancelled." If the cookie is not valid (e.g. has expired, has already been cancelled, or is simply incorrect), LISTSERV will send its standard message telling you in part that "The confirmation code 8F2E8F4B does not correspond to any pending command."

## 15.8 Denying Service to Problem Users

In addition to methods listed above for restricting service areas and the like, LISTSERV has a varied set of methods whereby "problem users" may be denied service on several levels.

### 15.8.1 The "Filter=" List Header Keyword

List owners or LISTSERV maintainers may filter specific userids (or users matching a wildcard specification) on a list-by-list basis by using the `FILTER=` list header keyword. For more information, see the [List Keyword Reference](#) document.

### 15.8.2 The "FILTER\_ALSO" Configuration File Variable

LISTSERV maintainers may add specific or wildcarded userids to LISTSERV's built-in filter by using the `FILTER_ALSO=` configuration file variable. Users matching this specification will be denied service to all LISTSERV services on your server.

### 15.8.3 The "SERVE" Command

The LISTSERV maintainer may selectively "serve out" specific userids with the SERVE command. This use of the SERVE command creates a "hard" serve-out which only a LISTSERV maintainer may override, as opposed to the "soft" serve-out that occurs when 51 consecutive bad commands are sent to LISTSERV which any user (other than the served-out user) can override. This function is particularly useful for temporary suspensions of service to certain userids, as it does not require modifying the site configuration file, but naturally it may be used to serve users out permanently. The

SERVE command does not accept wildcards, so if you need to suspend service to a class of users rather than to a specific user, you should use the `FILTER_ALSO` configuration file variable as described in Section 15.8.2 [The "FILTER\\_ALSO" Configuration File Variable](#).

LISTSERV writes an entry for each served-out user to its `PERMVAR` FILE. (PERMVAR FILE is not a plain-text file; do not edit it by hand.) To serve a user out manually, the LISMSERV maintainer sends the command:

```
SERVE internet-address OFF PW=password
```

The password is required. Acceptable passwords are the list creation password (`CREATEPW`), the system file modification password (`STOREPW`), or your personal password set with the `PW ADD` command.



**Note:** Prepending `QUIET` to the `SERVE` command suppresses notification to the user in question. If you are serving out an ID that is causing a mailing loop, you will probably want to use `QUIET` to avoid sending a fresh message to the looping ID.

Because the postmaster does receive notifications when a served-out user attempts to send mail to the server, the `SERVE` command also has a `DROP` sub-command that can be used to tell LISMSERV to "drop on the floor" all mail from that served-out address. For instance,

```
SERVE internet-address OFF DROP PW=password
```

To restore service to a user previously served out, the LISMSERV maintainer sends the command:

```
SERVE internet-address PW=password
```

Again, the password is required, and the `QUIET` modifier may be used.

When a user is served out by the LISMSERV maintainer, assuming the `QUIET` modifier is not prepended to the `SERVE` command, a message similar to the following is sent:

```
Date: Thu, 6 Dec 2001 09:45:40 -0500
From: "L-Soft list server at LISMSERV@LISMSERV.EXAMPLE.COM (1.8e)"
      <LISMSERV@LISMSERV.EXAMPLE.COM>
Subject: Message ("Your access to LISMSERV has just been suspended...")
To: baduser@somehost.com
```

Your access to LISMSERV has just been suspended by nathan@example.com. Commands and postings from you will be ignored from now on.

#### 15.8.4 The `POST_FILTER` List Exit Point

See the [Advanced Topics Guide for LISMSERV](#) for information on programming list exits and the `POST_FILTER` list exit point.

### 15.9 Hiding Selected Header Lines

It is possible to hide part or all of a list header (except for the list title) from users who send the `REVIEW` command or who try to view the list's configuration via the CataList. The following syntax is used:

```
* My very own list
*
* blah blah blah
*.HH ON
```

```
* This line is hidden
* This line is also hidden
*.HH OFF
* This line is not hidden
```

The sequence can be repeated as many times as required. GET will return the unedited header with the .HH sequences, REVIEW will replace hidden lines with a note saying that lines were hidden. You can't hide the fact that some lines were hidden because it would lead to people spending hours trying to figure out problems which only appear to be problems because some of the keywords are not visible. L-Soft will not field support inquiries with hidden headers; you must send the entire raw header (including the .HH lines) when requesting support.

### 15.10 Tracking Subscription Changes with the Change-Log Keyword

*This feature and keyword are not available in LISTSERV Lite.*

LISTSERV includes a "Change-Log=" list header keyword which, when set to "Yes", causes the server to log information about changes to individual subscriptions for a given list to a *listname*.CHANGELOG file (*listname* CHANGELOG on VM). Changelogs also log postings to the list. Please see Section 10.6 [Change Logs](#) and the [List Keyword Reference](#) document for more information on the "Change-Log=" keyword.



## Section 16 Subscription Features and Functions

### 16.1 Setting Up Subscription Confirmation

For lists coded "Subscription= Open", you can require confirmation on all new subscription requests, thus ensuring that LISTSERV has a clear mailing path back to the subscriber. In the past, a user could send a subscription for an open subscription list to LISTSERV, which upon acceptance would immediately start sending the user list mail. If the user was located behind a "broken" or one-way gateway, this produced immediate bounced mail until the list owner noticed and deleted the subscription. Note that requiring confirmation at the time of subscription does not guarantee that the clear mailing path will continue to exist permanently.

"Subscription= Open,Confirm" makes LISTSERV send a Command Confirmation Request to the potential subscriber before actually adding the user to the list. The subscriber is requested to reply to the request by sending a validation "cookie" back to LISTSERV (this "cookie" being the hexadecimal number pulled from the subject line).

The Command Confirmation Request, while straightforward, has the potential to cause confusion if users do not read carefully the instructions that make up the request. LISTSERV expects confirmation codes to be sent in a specific way because some mail gateways add lines to the header of the message that LISTSERV doesn't understand. If a user forwards the request back to LISTSERV, or creates a new mail message to send the 'cookie' back, it usually will not work correctly. The sequence should be as follows:

1. SEND the subscription request to LISTSERV.
2. REPLY to the confirmation request ('ok')
3. SEND the confirmation code (if necessary) ('ok 23CBD8', for example)
4. Send mail to list owner (not to the list) if the subscription request fails after step 3.



**Note:** If a list owner adds a user manually, the confirmation process is bypassed.

### 16.2 Defining Default Options for Subscribers at Subscription Time

The LISTSERV maintainer or the list owner may specify subscribe-time defaults for many subscriber options by using the "Default-Options=" keyword. This keyword takes regular SET options as its parameters. Examples include:

```
* Default-Options= DIGEST,NOREPRO,NOACK
* Default-Options= REPRO,NONE
```

You may have more than one "Default-Options=" line in your header, as needed.

If setting "Default-Options=" for an existing list, you will probably want to issue QUIET SET commands for existing subscribers, particularly if you are defaulting an option such as REPRO. Setting "Default-Options=" will not affect current subscribers.

Any Default-Options that you set are applied to non-subscribers. This is particularly useful if you want to run a public list but disallow non-subscribers from posting, or at least force their mail to go through a moderator first. For the former you could code "Default-Options= NOPOST", and for the latter, "Default-Options= REVIEW". These settings would also apply to new subscribers, and the latter particularly can help you ensure that the people who join your list are legitimate and haven't just joined the list so they can spam it.



**Note:** Any default topics are set with the "Default-Topics=" keyword. See the [List Keyword Reference](#) document for details on this keyword.

### 16.3 Setting Up Subscription Renewal

(In addition to the information in this section, see Section 13.5 [Address Probing](#), which is related to subscription renewal.)

You can code subscription renewal into your lists. This is one method to keep lists "pruned down" and avoid having large lists that are actually distributing mail to only a fraction of the users. For instance, you may have a number of subscriptions set to NOMAIL for one reason or another. NOMAIL user(a) may have forgotten that he has a subscription; user(b) may have set NOMAIL instead of unsubscribing; user(c) may no longer exist because she graduated or no longer works for the service provider; you may have set user(d) to NOMAIL because of recurrent mail delivery errors. Requiring a periodic confirmation of subscriptions is therefore a reasonable course of action for large, non-private lists.

To add subscription renewal, add the following keyword to the header of the list:

```
* Renewal= interval
```

– or –

```
* Renewal= interval,Delay(number)
```

where interval is a period of time such as Weekly, Yearly, 6-monthly, or something similar, and Delay(number) is an integer corresponding to how many days LISTSERV will wait for the renewal confirmation to arrive. (See the [List Keyword Reference](#) document for more information on renewal and delay periods.)

The confirmation request mailing asks the subscriber to send the command CONFIRM listname back to LISTSERV. If the subscriber does not do so within a certain length of time, LISTSERV automatically deletes the subscription. The default delay time is 7 days. If you wish to use the default delay time, it is not necessary to code Delay() into your Renewal parameters.



**Note:** You may wish to increase the delay time to accommodate users whose subscriptions expire over holidays (such as the Christmas/New Year's week) in order to avoid accidental deletions. Also, be aware that confused subscribers can and will send the CONFIRM command back to the list, rather than to LISTSERV. LISTSERV's default filter will catch these commands and forward them to the userid(s) defined by the "Errors-To=" keyword.

It is possible to waive subscription renewal for certain users (such as list owners, editors, redistribution lists, etc.). In order to do this, simply issue the command

```
[QUIET] SET listname NORENEW FOR net-address
```

to LISTSERV. (Remove the brackets around "QUIET" if you want to use the QUIET option. The brackets indicate that QUIET is an optional part of the command.) It is most advisable to do this in the case of redistribution lists, as they broadcast the renewal notice to their users, who a) cannot renew the subscription and b) become very confused when they see the notice, often sending "what does this mean?" mail to the list.

The LISTSERV maintainer or the list owner can also issue the CONFIRM command for a subscriber:



```
[QUIET] CONFIRM listname FOR net-address
```

LISTSERV creates a daily report on its subscription renewal activities, telling you what users were requested to confirm subscriptions, and what users were deleted for failing to confirm the renewal request. This report is sent to the "Errors-To=" address for the list. A typical subscription renewal monitoring report is reproduced below:

*Figure 16-1 Typical Daily Subscription Renewal Monitoring Report*

```
Date: Thu, 30 Oct 1998 06:00:01 -0400
From: "L-Soft list server at PEACH.EASE.LSOFT.COM (1.8d)"
      <LISTSERV@PEACH.EASE.LSOFT.COM>
Subject: ACCESS-L: Subscription renewal monitoring report
To: ACCERR@LINUS.DC.LSOFT.COM

The following 3 subscribers were deleted from the ACCESS-L list today:
  AHUIE@FIN-AID.CSUHAYWARD.EDU
  byetter@ECLAT.UCCS.EDU
  mgill1@IX.NETCOM.COM

The following 5 subscribers were requested to confirm their
subscription to the ACCESS-L list:
  Ernest_HILL%KN=NYC=ZE@MCIMAIL.COM
  dave@MAIL.DERBY.K12.KS.US
  dianna@INTEX.NET
  koa@KOAMAR.COM
  leider@STR.DAIMLER-BENZ.COM
```



## Section 17 Other Features and Functions

### 17.1 Setting Up National Language Mail Templates

While LISTSERV is not shipped with non-English language mail templates, it is possible to create such "national language" templates and make them available on a list-by-list basis by using the "Language=" list header keyword. The procedure to use national-language templates is as follows:

- Translate DEFAULT.MAILTPL (or a desired subset of the templates in DEFAULT.MAILTPL) into the desired language.
- Call the translated template file *idiom*.MAILTPL, where "idiom" is replaced by the name of the language, for example, FRANCAIS.MAILTPL, ESPANOL.MAILTPL, etc.
- Store the file on the server with a PUT command.
- For lists that will use the national language template, code "Language= idiom" in the list header. For instance, if the national language template is called FRANCAIS.MAILTPL, code "Language= Francais". If you've called it FRENCH.MAILTPL instead, then code "Language= French".



**Note:** LISTSERV's hard-coded messages will continue to be issued in English, and any editable template not present in your national language template will be pulled from DEFAULT.MAILTPL (or the list's own MAILTPL file, if it exists) when needed.

See Section 9 [Creating and Editing Mail and Web Templates](#) for more information on creating and editing LISTSERV's mail templates. L-Soft does not provide national language mail templates.

### 17.2 Translating Control Characters Included in List Mail

LISTSERV removes control characters from mail messages by default and expands tabs in the process. Control characters usually have undesirable and unexpected results, as they seldom survive the double ASCII->EBCDIC->ASCII translation and cause the recipient's terminal to execute sequences different from the ones meant by the message sender - not to mention the case of ASCII control characters on EBCDIC terminals. Furthermore, certain combinations of control characters were found to create problems with IBM's SMTP (due to unexpected CRLF sequences appearing in the middle of a record after translation). Lists that absolutely require control characters must have a "Translate= No" keyword added to their header.

### 17.3 Communicating with List Owners

The LISTSERV maintainer may have occasion to communicate with some or all of the list owners who run lists on his server. This does not require that the LISTSERV maintainer keep a "super-list" of list owners, but only that he or she use certain addresses when communicating with list owners.

#### 17.3.1 The Listname-REQUEST Alias

If you need to communicate with all of the list owners of a single list, simply address your mail to listname-REQUEST. This special address will be expanded by LISTSERV to include all non-quiet list owners of the specified list. For instance, mail to the list owners

of the PEKINESE list on LISTSERV.SIRIUS.NET would be addressed to PEKINESE-REQUEST@LISTSERV.SIRIUS.NET.

To limit the possibility of random spam being sent to listname-REQUEST addresses, all persons sending mail to them will be required to respond to an OK confirmation request before the mail is forwarded to the appropriate person(s).

### 17.3.2 The ALL-REQUEST Alias

If you need to communicate with all of the non-quiet list owners on your server, simply write to the special ALL-REQUEST alias. ALL-REQUEST is expanded by LISTSERV to include all non-quiet list owners of all the lists on your server.



**Note:** ALL-REQUEST should only be used in emergencies, or when all non-quiet list owners need to be informed that something is happening, such as a migration from one server to another.

For general news it is probably better to create an administrative mailing list for your list owners rather than to use ALL-REQUEST.

The ability to post to the ALL-REQUEST alias, which expands to all non-quiet list owners, is restricted as follows:

1. The site configuration variable, ALL\_REQUEST\_ALLOWED\_USERS, can be used to specify who can mail to ALL-REQUEST. This variable uses the same syntax as POSTMASTER=, in other words, a space-separated list of userid@host specifications.
2. Postmasters are always allowed to mail to ALL-REQUEST, even when not listed in ALL\_REQUEST\_ALLOWED\_USERS.
3. The determination is made conclusively on the RFC821 MAIL FROM because:
  - a. This avoids dealing with header errors. Header error = don't know who sent this = discard silently = unhappy admin who had to send an urgent notice to all his owners.
  - b. The main point of this change is to block spammers, and spammers typically have non-working or null MAIL FROM: addresses. Needless to say, null doesn't pass the test.
4. When the MAIL FROM is not null, the REQNAK1 mail template form is sent when the message is rejected.

In addition, all persons sending mail to the ALL-REQUEST alias will be required to respond to an OK confirmation request before the mail is forwarded to the appropriate person(s).

Although a further request was considered for a way to disable ALL-REQUEST entirely, it was decided not to implement it, as the main concern was to block spammers and general users from posting to the alias at random.

### 17.3.3 Required Configuration for Unix and VMS Servers Running PMDF



**Note:** VM servers, VMS servers running MX, and Windows servers do not require that any special aliasing be added for these aliases. This functionality is built-in for those installations.

You should already have coded "*listname-request*" aliases into your */etc/aliases* file (unix) or your PMDF aliases file (VMS running PMDF) for each list on your server. See Section 7 [Creating and Maintaining Lists](#) for more information on coding these aliases.

If you are running a unix server, or a VMS server running PMDF, you will probably have to code an alias for "*all-request*" into your aliases file, as this is not normally done at install time. See your installation guide for information on how to code the base-level "*listserv*", "*owner-listserv*", and "*listserv-request*" aliases and follow those instructions to add an alias for "*all-request*".

### 17.3.4 Other Aliases Used by LISTSERV

The following aliases need to be added for lists running on VMS (with PMDF) and unix. All other configurations handle them automatically.

In addition to the aliases for "*listname*" and "*listname-request*", LISTSERV also uses the following aliases for specific purposes:

- *owner-listname*

This alias is placed by LISTSERV in the RFC821 MAIL FROM: header. The expectation is that, per RFC821 et seq., remote hosts will send any delivery errors back to the RFC821 MAIL FROM: address. LISTSERV considers anything sent to the "*owner-listname*" address as a delivery error (which can cause a problem with some older mail clients, such as Microsoft Mail, which use the RFC821 MAIL FROM: rather than the RFC822 From: header as the originator of the mail). This alias maps to the value in the "Errors-To=" list header keyword. In general this address should not be used to communicate with the list owner (*listname-request* is the preferred alias for contacting list owners) as mail sent to this address is always handled as an error.

- *listname-server*

This alias is an artefact from the days when it was not always clear what the name of the account running the mailing list manager was. In general, if you had forgotten if the list was running on LISTSERV or Majordomo or whatever, you could write to *listname-server@host* and your commands would reach the server. For LISTSERV's purposes, this alias maps to *LISTSERV@host*. While this alias is not absolutely required, we recommend that it be made available because old documentation may still indicate that this is a legitimate way to write to the mailing list manager.

- *listname-search-request*

This alias handles search requests coming from INDEX mode subscriptions and must be present for each list for INDEX to work properly.

- `listname-signoff-request` and `listname-unsubscribe-request`

Mail sent to these aliases will cause a signoff event for the `userid` in the From: line (no command is required and the body of the message is discarded).

- `listname-subscribe-request`

Mail sent to this alias will cause a subscribe event for the `userid` in the From: line (no command is required and the body of the message is discarded).

## Section 18 Special Functionality for ISP's

These functions require that your site is appropriately licensed for them. Specifically, your LAK must contain the ISP "Scope" option. Contact the L-Soft sales department for details.

Currently the ISP functionality is under development and does not include a complete suite of tools that an ISP might find useful. If you have suggestions for useful tools (other than accounting tools which are in development), please feel free to write to [SUPPORT@LSOFT.COM](mailto:SUPPORT@LSOFT.COM) with your comments, which will be turned over to the developers.

### 18.1 Directory Quotas for Individual Lists

Currently there is no warning message when a list hits a preset percentage of its storage quota, so this function should be used with care.

#### 18.1.1 The QUOTA.FILE

LISTSERV uses a file called quota.file to store quota information for individual lists. The quota.file must be installed in LISTSERV's "A" directory (the same directory where LISTSERV keeps list files and its other standard data files). The file is a flat text file with the information for each list kept on one line, as follow:

```
/HOME/LISTS/MYLIST-L      1024      Owner
(1)                       (2)       (3)
```



**Notes:** The directory where the notebook archives and any other user-maintained files belonging to the list are kept. This specification should be the same as the specification in the "Notebook=" keyword (for lists with notebook archives) or the same as the specification for the file archives directory for the list in site.catalog (for lists without notebook archives).

The size (in kilobytes) of the list's quota. Note that 1024 kilobytes = 1 megabyte, so multiply the desired number of megabytes by 1024 to set this value.

The person who should be notified when the list goes over quota; in this case, the access-level "Owner" means that the list owner is notified. This value can also be a regular internet-address.

#### 18.1.2 Displaying Quota Information

To display current quota information, issue the command

```
SHOW QUOTA
```

LISTSERV will respond with a listing of the lists for which quotas are set, along with the quota setting and percentage of quota used. A typical SHOW QUOTA report is reproduced below, for lists called ALIST, BLIST, and so forth:



Figure 18-1 Typical Output of a SHOW QUOTA Command Issued by a Privileged User

```

Date:      Mon, 17 Jun 1996 17:09:38 -0400
From:      "L-Soft list server at HOME.EASE.LSOFT.COM (1.8d) "
           <LISTSERV@HOME.EASE.LSOFT.COM>
Subject:   Output of your job "NATHAN"
To:        Nathan Brindle <NATHAN@EXAMPLE.COM>

> show quota
Directory          Quota  Usage
-----          -
E:\FTP\LISTS\ALIST      4M    0k  ( 0.0%)
E:\FTP\LISTS\BLIST      4M    4k  ( 0.0%)
E:\FTP\LISTS\CLIST      4M   323k ( 7.8%)
E:\FTP\LISTS\DLIST      1M   11k  ( 1.0%)
E:\LISTS\ELIST          4M   11k  ( 0.2%)
E:\LISTS\FLIST          4M  940k (22.9%)
E:\LISTS\GLIST          50M  9.8M (19.5%)
E:\LISTS\HLIST          4M   66k  ( 1.6%)

```

Your list owners (or the person(s) indicated by the third parameter in quota.file for the list) can also issue the SHOW QUOTA command, but they will receive quota information only for the list(s) for which they appear in that third parameter.

### 18.1.3 Reloading Quota Information After Making Changes

Whenever you change values or add or delete lists from quota.file, you must issue the command

```
SHOW QUOTA RELOAD
```

to reload the quota file. (Rebooting LISTSERV also reloads the information.)

## 18.2 Limiting the Number of Subscribers to a List

Using the special `Limits=` keyword, you can limit a list to a specified number of subscribers. Only a LISTSERV maintainer may raise, lower, or disable this limit. An attempt by a list owner to change or disable the limit will result in an error message being returned to the invoker and no change being made to the list header.

To enable the subscriber limit in the list header, code

```
* Limits= Sub(nnn)
```

where `nnn` is the number of subscribers you want to limit the list to. For instance, a list coded `Limits= Sub(200)` would be limited to 200 subscribers.



	DIGests/INDEX/NODIGests/NOINDEX	-> Ask for digests or message indexes rather than getting messages as they are posted
	REPro/NOREPro	-> Copy of your own postings?
	TOPICS: ALL	-> Select topics you are subscribed to (add/remove one or replace entire list)
	<+/->topicname	
Options for mail headers of incoming postings (choose one):		
	FULLhdr or FULL822	-> "Full" (normal) mail headers
	IETFhdr	-> Internet-style headers
	SHORThdr or SHORT822	-> Short headers
	DUALhdr	-> Dual headers, useful with PC or Mac mail programs
	SUBJecthdr	-> Normal header with list name in subject line
CONFIRM	listname1 <listname2 <...>>	Confirm your subscription (when LISTSERV requests it)
Other list-related commands		
-----		
GETPOST	listname refl <ref2 <...>> <opt>	Order individual messages from list archives
There is a single option:		
	NOMIME	Retrieve messages in "raw" form, ie, do not re-encode MIME attachment links (pre-1.8e behavior)
INDEX	listname	Sends a directory of available archive files for the list, if postings are archived
Lists	<option>	Send a list of lists as follow:
	(no option)	-> Local lists only, one line per list
	Detailed	-> Local lists, full information returned in a file
	Global /xyz	-> All known lists whose name or title contains 'xyz'
	SUMmary <host>	-> Membership summary for all lists on specified host
	SUMmary ALL	-> For all hosts (long output,

		send request via mail!)
	SUMmary TOTAL	-> Just the total for all hosts
Query	listname	Query your subscription options for a particular list (use the SET command to change them)
	*	-> Query all lists you are subscribed to on that server
REGister	full_name	Tell your name to LISTSERV, so that you don't have to specify it on subsequent SUBSCRIBE's
	OFF	Make LISTSERV forget your name
REView	listname <(options)>	Get information about a list
	BY sort_field	-> Sort list in a certain order:
	Country	by country of origin
	Date	by subscription date
	Name	by name (last, then first)
	NODEid	by hostname/nodeid
	Userid	by userid
	BY (field1 field2)	-> You can specify more than one sort field if enclosed in parentheses: BY (NODE NAME)
	Countries	-> Synonym of BY COUNTRY
	Topics	-> Include breakdown of subscribers per topic
	LOCal	-> Don't forward request to peers
	Msg	-> Send reply via interactive messages (BITNET users only)
	NOHeader	-> Don't send list header
	Short	-> Don't list subscribers
	ALL	-> List both concealed and non-concealed subscribers (list owners/site maintainers only)
SCAN	listname text	Scan a list's membership for a name or address
SEArch	listname word1 <word2 <...>>	Search list archives
	or: word1 <word2 <...>> IN listname	
	FROM date1	-> From this date
	TODAY	-> From today
	TODAY-7	-> In the last 7 days
	TO date2	-> To this date

	WHERE	
	SUBJECT CONTAINS xxxx	-> Only this subject
	AND/OR	
	SENDER CONTAINS xxxx	-> Only this author
		Complex boolean operations are supported, see database guide
STats	listname <(options>	Get statistics about a list (VM)
	LOCal	-> Don't forward to peers
	Informational commands	
	-----	
	Help	Obtain a list of commands
INFO	<topic>	Order a LISTSERV manual, or get
	<listname>	a list of available ones (if no topic was specified); or get information about a list
Query	File fn ft <filelist> <(options>	Get date/time of last update of a file, and GET/PUT file access code
	FLags	-> Get additional technical data (useful when reporting problems to experts)
RELEASE		Find out who maintains the server and the version of the software and network data files
SHOW	<function>	Display information as follows:
	ALIAS node1 <node2 <...>>	-> BITNET nodeid to Internet hostname mapping
	BITEARN (VM only)	-> Statistics about the BITEARN NODES file
	DISTRIBUTE	-> Statistics about DISTRIBUTE
	DPATHs host1 <host2 <...>>	-> DISTRIBUTE path from that server to specified host(s)
	DPATHs *	-> Full DISTRIBUTE path tree
	FIXes (VM only)	-> List of fixes installed on the server (non-VM see LICENSE)
	HARDWare or HW	-> Hardware information
	LICense	-> License/capacity information and software build date

LINKs node1 <node2 <...>>	-> Network links at the BITNET node(s) in question
NADs node1 <node2 <...>>	-> Addresses LISTSERV recognizes as node administrators
NETwork (VM only)	-> Statistics about the NJE network
NODEntry node1 <node2 <...>>	-> BITEARN NODES entry for the specified node(s)
NODEntry node1 /abc*/xyz	-> Just the ':xyz.' tag and all tags whose name starts with 'abc'
PATHs snode node1 <node2 <...>>	-> BITNET path between 'snode' and the specified node(s)
POINTs <ALL   list1 list2...>	-> Graduated license point information for planning
STATs	-> Usage statistics (default option)
VERSion	-> Same as RELEASE command
(no function)	-> Same as SHOW STATS

Commands related to file server functions

-----

AFD		Automatic File Distribution
ADD	fn ft <filelist <prolog>>	Add file or generic entry to your AFD list
DELeTe	fn ft <filelist>	Delete file(s) from your AFD list (wildcards are supported)
List		Displays your AFD list
	For node administrators: FOR user ADD/DEL/LIST etc	Perform requested function on behalf of a user you have control over (wildcards are supported for DEL and LIST)
FUI		File Update Information: same syntax as AFD, except that FUI ADD accepts no 'prolog text'
GET	fn ft <filelist> <(options)>	Order the specified file or package
	PROLOGtext xxxx	-> Specify a 'prolog text' to be inserted on top of the file
GIVE	fn ft <filelist> <TO> user	Sends a file to someone else

INDEX	<filelist>	Same as GET xxxx FILELIST (default is LISTSERV FILELIST)
PW	function	Define/change a "personal password" for protecting AFD/FUI subscriptions, authenticating PUT commands, and so on
	ADD firstpw	-> Define a password for the first time
	Change newpw <PW=oldpw>	-> Change password
	RESET	-> Reset (delete) password
SENDme		Same as GET
Other (advanced) commands		
-----		
DATABASE	function	Access LISTSERV database:
	Search DD=ddname <ECHO=NO>	-> Perform database search (see INFO DATABASE for more information on this)
	List	-> Get a list of databases available from that server
	REFRESH dbname	-> Refresh database index, if suitably privileged
DBase		Same as DATABASE
DISTRIBUTE	<type> <source> <dest> <options>	Distribute a file or a mail message to a list of users (see INFO DIST for more details on the syntax)
	Type:	
	MAIL	-> Data is a mail message, and recipients are defined by '<dest>'
	MAIL-MERGE	-> Data is a mail-merge message. See the Advanced Topics Manual to LISTSERV for specifics.
	POST	-> (non-VM only) Same as MAIL except that the message is pre-approved. See the Advanced Topics Manual to LISTSERV for specifics.
	FILE	-> Data is not mail, recipients are defined by '<dest>'
	RFC822	-> Data is mail and recipients are defined by the RFC822



		'To:/'cc:' fields
Source:		
DD=ddname		-> Name of DDname holding the data to distribute (default: 'DD=DATA')
Dest:		
<TO> user1 <user2 <...>>		-> List of recipients
<TO> DD=ddname		-> One recipient per line
Options for the general user:		
ACK=None/MAIL/MSG		-> Acknowledgement level (default: ACK=NONE)
CANON=YES		-> 'TO' list in 'canonical' form (uid1 host1 uid2 host2...)
DEBUG=YES		-> Do not actually perform the distribution; returns debug path information
INFORM=MAIL		-> Send file delivery message to recipients via mail
TRACE=YES		-> Same as DEBUG=YES, but file is actually distributed
AV=YES[,FORCE]		-> Check the message for viruses. See the Advanced Topics Manual for LISTSERV for specifics.
DKIM=NO/YES		-> Sign the message with a DomainKeys signature. (default: DKIM=NO)
Options requiring privileges:		
FROM=user		-> File originator
FROM=DD=ddname		-> One line: 'address name'
PRE-APPROVED=YES		-> Pre-approve message (with DISTRIBUTE POST only)
FOR	user command	Execute a command on behalf of another user (for node administrators)
SERVE	user	Restore service to a disabled user
THANKs		Check the server is alive
UDD		Access the User Directory Database (there are 18 functions and many sub-functions, so the syntax is not given here)

File management commands (for file owners only)		
-----		
AFD/FUI		Automatic File Distribution
	GET fn ft <filelist>	Get a list of people subscribed to a file you own
GET	fn FILELIST <(options> CTL	Special options for filelists: -> Return filelist in a format suitable for editing and storing back
	NOlock	-> Don't lock filelist (use in conjunction with CTL)
PUT	fn ft <filelist <NODIST>> <CKDATE=NO>	Update a file you own -> Accept request even if current version of the file is more recent than the version you sent
	<DATE=yyymmddhhmmss>	-> Set file date/time
	<PW=password>	-> Supply your password for command authentication
	<RECFM=F <LRECL=nnn>>	-> Select fixed-format file (not to be used for text files)
	<REPLY-TO=user>	-> Send reply to another user
	<REPLY-TO=NONE>	-> Don't send any reply
	<REPLY-VIA=MSG>	-> Request reply via interactive messages, not mail
	<"parameters">	-> Special parameters passed to FAVE routine, if any
	Standard parameters supported for all files:	
	TITLE=file title	-> Change file "title" in filelist entry
REFRESH	filelist <(options> NOFLAG	Refresh a filelist you own -> Don't flag files which have changed since last time as updated (for AFD/FUI)
UNLOCK	fn FILELIST	Unlock filelist after a GET with the CTL option if you decide not to update it after all

## List management functions

-----

Commands that support the QUIET keyword are marked (\*)

ADD(*)	listname user <full_name>	Add a user to one of your lists, or update his name
	listname DD=ddname	-> Add multiple users, one address/name pair per line
	listname DD=ddname IMPORT <PRELOAD>	-> Bulk add multiple users, one address/name pair per line PRELOAD option loads addresses into memory before adding to speed up operation
ADDHere(*)		Same as ADD, but never forwards the request to a possibly closer peer
CHANGE(*)	listname * oldaddr pattern newaddr *@newhost	Change a subscriber's address (List owner's version)
DELeTe(*)	listname user <(options)>	Remove a user from one of your lists, or from all local lists
	listname DD=ddname <BRIEF>	Bulk delete multiple users, one address per line. BRIEF option omits verbose response of who was deleted.
	Options:	
	GLobal	-> Forward request to all peers
	LOCAl	-> Don't try to forward request to closest peer if not found locally
	TEST	-> Do not actually perform any deletion (useful to test wildcard patterns)
EXPLODE	listname <(options)>	Examine list and suggest better placement of recipients, returning a ready-to-submit MOVE job
	BESTpeers n	-> Suggest the N best possible peers to add
	Detailed	-> More detailed analysis
	FOR node	-> Perform analysis as though

		local node were 'nodeid'
	PREFer node	-> Preferred peer in case of tie (equidistant peers)
	SERVice	-> Check service areas are respected
	With(node1 <node2 <...>>>)	-> Perform analysis as though specified nodes ran a peer
	WITHOut (node1 <node2 <...>>>)	-> Opposite effect
FREE	listname <(options> GLobal	Release a held list -> Forward request to all peers
GET	listname <(options>  GLobal HEADer	Get a copy of a list in a form suitable for editing and storing list and lock it -> Forward request to all peers -> Send just the header; on the way back, only the header will be updated
	NOLock	-> Do not lock the list
	OLD	-> Recover the "old" copy of the list (before the last PUT)
HOLD	listname <(options>  GLobal	Hold a list, preventing new postings from being processed until a FREE command is sent -> Forward request to all peers
LISTs	OWNed	Send back a list of local lists owned by the invoker
	MODerated	Send back a list of local lists moderated by the invoker
MOVE(*)	listname user <TO> node  listname DD=ddname	Move a subscriber to another peer -> Move several subscribers to various peers
PUT	listname LIST	Update a list header from the file returned by a GET command
PUTALL	listname LIST	Similar to PUT but lets you store the entire list, header and subscribers together
Query	listname <WITH options> FOR user	Query the subscription options

		of another user (wildcards are supported)
	* <WITH options> FOR user	Searches all the lists you own
SET(*)	listname options <FOR user> *	Alter the subscription options of another user or set of users (when using wildcards)
	Additional options for list owners:	
	NORENEW/RENEW	-> Waive subscription confirmation for this user
	NOPOST/POST	-> Prevent user from posting to list
	EDITor/NOEDITor	-> User may post without going through moderator
	REView/NOREView	-> Postings from user go to list owner or moderator even if user is allowed to post
STats	listname (RESET	Resets statistics for the list
UNLOCK	listname	Unlock a list after a GET, if you decide not to update it after all
Site management functions -----		
CMS	command_text	Issue a CMS command and get the last 20 lines of response sent back to you, the rest being available from the console log
CP	command_text	Issue a CP command and get up to 8k of response data sent to you (the rest is lost)
DATABase	function DISAble  ENABle SHUTDOWN	Control operation of databases: -> Disable interactive database access, without shutting down existing sessions -> Re-enable interactive access -> Shut down all interactive database sessions, and disable interactive access
INSTALL	function	Software update procedure:

	CLEANUP shipment	-> Remove an installed shipment from the log
	CLEANUP BEFORE dd mmm yy	-> Remove all shipments installed before that date
	PASSWORD shipment PW=instpw	-> Confirm installation of a shipment, when requested by LISTSERV
	RELOAD shipment	-> Attempt to reload a shipment which failed due to a disk full condition
	STATus	-> Get a list of installed "shipments"
LISTs	OWNed <BY> userid@host	Send back a list of local lists owned by the address supplied (wildcards acceptable)
	MODerated <BY> userid@host	Send back a list of local lists moderated by the address supplied (wildcards acceptable)
NODESGEN	<WTONLY>	Regenerate all LISTSERV network tables, or just compile the links weight file (debugging command)
OFFLINE		Suspend processing of reader files and disable the GET command
ONLINE		Cancel OFFLINE condition
PUT	listname LIST	Create a new list
PUTC	fn ft <fm cuu dirid> <RECFM=F LRECL=nnn>	Update a CMS file on one of LISTSERV's R/W minidisks; note that this is similar to SENDFILE + RECEIVE or LINK + COPYFILE and should NOT be used to update file-server files
PWC	function	Password file management:
	ADD user newpw	-> Define a password for the specified user
	DELeTe user	-> Delete password for that user
	Query user	-> Query the password of the

		specified user
REGister	name OFF FOR user	Set a user's SIGNUP FILE entry
SENDfile	fn ft <fm cuu dirid>	Request the server to send you a file from one of its disks
SERVE	user OFF	Permanently suspend access from an abusive user or gateway (restore with 'SERVE user')
	user OFF DROP	Permanently suspend access from an abusive user or gateway and drop all further inbound mail from this sender on the floor (restore with 'SERVE user')
	LIST	Return a list of all addresses that are currently served off or which are spam-quarantined
SF		Same as SENDFILE
SHOW	BENCHmarks	-> CPU/disk/paging benchmarks
	EXECload	-> Statistics about EXECLOADED REXX files
	LSVFILER	-> Statistics about LSVFILER file cache
	PREXX	-> Statistics about PREXX functions usage
	STORage	-> Information about available disk space and virtual storage
SHUTDOWN	<REBOOT REIPL>	Stop or reboot the server (the two options are synonyms)
STOP		Same as SHUTDOWN

Note: some debugging commands and options have been omitted.

#### Syntax of parameters

-----

filelist = 1 to 8 characters from the following set: A-Z 0-9 \$#@+\_-:  
 fformat = Netdata, Card, Disk, Punch, LPunch, UUencode, XXencode, VMSdump,  
 MIME/text, MIME/Appl, Mail  
 fn = same syntax as 'filelist'  
 ft = same syntax as 'filelist'



full\_name = firstname <middle\_initial> surname (\*not\* your e-mail address)  
host = Internet hostname  
listname = name of an existing list  
node = BITNET nodeid or Internet hostname of a BITNET machine which  
has taken care of supplying a ':internet.' tag in its BITEARN  
NODES entry  
pw = A password with characters from the set: A-Z 0-9 \$#@\_?!|%  
user = Any valid Internet address not longer than 80 characters; if  
omitted, the 'hostname' part defaults to that of the command  
originator

## Appendix B: Sample Boilerplate Files

So-called "boilerplate" files are handy for list owners who find themselves answering the same questions over and over again. Usually these questions refer to basic LISTSERV usage. You can save yourself a lot of time by keeping files on-line such as the ones below to cut and paste into replies. Feel free to edit these to suit your own tastes (or compose your own!).

(Be sure to insert the appropriate list names and LISTSERV hosts as required.)

### Subscription Requests Sent to the List

LISTSERV subscription requests need to be sent to the LISTSERV address rather than to the list itself. You do this by sending mail to `LISTSERV@host` with the command

```
SUB listname Your Name
```

as the body of the message. If you are unfamiliar with LISTSERV and its associated commands, I suggest that you add the commands

```
INFO GENINTRO
INFO REFCARD
```

as additional lines of your message. LISTSERV will then send you a file containing a General Introduction to Revised LISTSERV that will give you some instruction on the service and a Quick Reference Card of the various commands.

Thanks for your interest. If you have trouble subscribing with this method, please let me know and I will attempt to help.

If you have `Subscription= Open,Confirm` you might want to add the following:

*Because LISTSERV verifies mailing paths for new subscribers (a process not implemented when the list administrator adds people manually), it is preferred that users subscribe themselves by the method outlined above.*

### Sending Other Commands to the List or to the \*-REQUEST Address for the List

On Sun, 20 Mar 1994 22:44:25 -0800 (PST) you said:

```
>"INFO REFCARD"
```

*You need to redirect LISTSERV commands like the above (minus the double quotes by the way), to `<listserv@host>`. The \*-request type addresses are for reaching the person that run the list.*

[another version:]

*You've sent mail that appears intended for a mailing list to one of the addresses used to reach the list owner. That is, rather than sending your mail to `listname@host` you've sent the note to `OWNER-listname@host` or `listname-REQUEST@host`. Please re-send the appended note to the list address if you haven't done so already.*

----- original message follows:

## Unsubscribed User Still Getting Mail

Use this one after you have done an exhaustive search of the list and determined that the person simply isn't on the list. Typically the user is subscribed to a redistribution list and doesn't realize it.

*Unfortunately I can't unsubscribe you from listname because you aren't subscribed to listname@host. I have run a check to see if you might be subscribed under a slightly different network address and have not found anything.*

*There are a few possibilities you should look into. Are you getting a digest? Are you perhaps getting a redistributed copy of postings, possibly from a redistribution list? If you look at the mail headers, and there is an indication that you may be getting the postings from another source, you will have to ask the people that run the other source to remove you from their list.*

Use this one if the user unsubscribed successfully, BUT they are still getting list mail.

*I've done a search of listname for a possible duplicate subscription for you and have not found anything. It's possible that the mail you are receiving was actually sent from listname before your unsubscribe request was processed. Depending on the routing, it could take anywhere from 24 to 48 hours for all such messages to get through the network, so please be patient.*

## Quoted Replies Include Message Headers Causing them to Bounce

When quoted replies from a user's mail client includes message headers in the mail body, the reply will be bounced back to the list owner.

If you forward such messages to the list, or back to the sender, you can add the following at the beginning. I ran across this one in the CBAY-L mailing list archives, and edited it slightly.

*This message was sent to me from LISTSERV instead of the list. The original message included the entire message being replied to, including the mail headers. These headers in the body pointed to the list itself. LISTSERV has mail-loop avoidance code and when it sees headers that it thinks it generated itself, it bounces the message to the list owner. If your mail client does this, please remember to delete such "included header lines" from the body of your list replies.*

*-----original message-----*

D.6.

Add the following to ask a postmaster for help on a bounced address you've set to NOMAIL; don't forget to include a cc: to the bounced address.

*Postmaster(s),*

*Can you shed any light on the following error message? Please let me know what you find as I have removed the e-mail address from the mailing list in question and would like to restore service as soon as is feasible.*

*Thanks.*

*Aside to user: Should this note reach you (meaning that the mail delivery problems have been resolved), you can re-enable your mail service by sending mail to listserv@host with the following command: SET listname MAIL*

## Delivery Error with Unknown User Account

If you get a delivery error that doesn't specify which user account is causing the bounce, then use the following:

*Postmaster,*

*I received the appended mail delivery report from your system and need help isolating the e-mail address that is causing the error. That is, there are multiple recipients from your system on the list but the delivery error doesn't explicitly mention any of the users on the list. I'm including a list of subscribers from your system. If any of them are no longer valid, or aren't usable address for some other reason, please let me know.*

*---- list of e-mail address on the indicated list follows:*

## Setting a User to DIGEST because of Bouncing Mail

If you've set a user to DIGEST because of bouncing mail, and the user is asking why he/she is now getting the digest, then use the following:

*I received a mail delivery error for your address and issued a*

*SET listname DIGEST*

*on your behalf to minimize the number of bounce messages. I also sent a copy of the error I received to your postmaster (or the postmaster of the mail gateway that generated the error), asking for help. And since such delivery problems are often transient, I CC'd a copy of that note to your address, and included instructions for turning your mail back on. Apparently I didn't hear anything from your postmaster, or he/she said not to turn your mail back on until the problem was resolved. If they had responded and said the problem was resolved, I would have set you back to MAIL.*

*The other possibility is that I received a mail message indicating that there was some temporary problem with your account. In that case, for example if you had exceeded your disk quota and couldn't receive any new mail, I would not have bothered your postmaster. I have a different form letter that I send when that happens. Again it explains what has occurred and includes instructions for re-enabling your mailing list subscription. But I only send that one to the address the list member. Either way, whatever was wrong has been corrected, and you'd probably like to start receiving mail again. So, here's how you can restore your mail service. If you have any problems doing so, please let me know and I'll help. But since I don't know which of the three mail service options you had chosen before, I can't do it for you without guessing. You can re-enable your mail service by sending mail to listserv@host with one of the following commands*

*SET listname MAIL*

*SET listname DIGEST (if you want digest-format mail)*

*SET listname INDEX (if you want digest-index-format mail)*

*in the \*body\* of the mail message. Please note that these settings are mutually exclusive, you can't choose more than one.*

## A Sample "Your List has been Created" Boilerplate

### Mailing List Setup Confirmation

I have created the XXXXX-L list on LISTSERV.MYHOST.COM per your setup sheet.

If you are new to LISTSERV, you will probably want to download L-Soft's Quick Start manual for list owners. Simply point your web browser to

<http://www.lsoft.com/manuals/QS-index.html>

and view online or choose the version appropriate for your word processor or viewer.

Formal documentation of list owner commands and other list ownership issues can be found in the List Owner's Manual, which is available at the URL

<http://www.lsoft.com/manuals/ownerindex.html>

Per your list service agreement, support for your list is handled through a mailing list, LIST-SUPPORT@LISTSERV.MYHOST.COM. You have been added to that list. Please direct all support questions to the LIST-SUPPORT list.

You may also be interested in subscribing to the LSTOWN-L mailing list for LISTSERV list owners. To do so, send a mail message to LISTSERV@LISTSERV.NET with the command

```
SUB LSTOWN-L Your Name
```

in the body (not the subject) of the message. There are a number of extremely experienced LISTSERV list owners subscribed there who are more than willing to share their expertise. Don't hesitate to ask for help.

You now need to instruct LISTSERV to add personal passwords for the list owner account(s). These passwords are used to validate privileged commands (such as the PUT command for storing your list "header" on the server after making changes to it). This is done by sending mail from each account to LISTSERV@LISTSERV.MYHOST.COM with the command

```
PW ADD password
```

(again, "password" is whatever you want it to be) in the body of the message. LISTSERV will request confirmation of this operation; simply reply to the confirmation request with the word "ok".

Adding these passwords will considerably lessen the chance that someone will "spooof" mail from you to make changes on your list. It is very unlikely that this will happen, but it never hurts to be cautious.

Sincerely,

Joe Smith

LISTSERV Maintainer

# Index

## A

- Address Probing
  - active 207
  - operating system issues 209
  - passive 208
- Administration Interface
  - customizing web pages 35
  - installing 30
  - log files 169
  - see *Web Administration Interface*
- Announce-Only Lists
  - creating 97
- Anti-Spamming Filter 205
- Anti-Virus Scanning 44
- Archive Directories
  - setting up 43
- Auto-Deletion Feature 203
- Automatic File Distribution 127
- Auto-Responder Lists
  - creating 96

## B

- BITNET 237, 238, 239, 248
  - network table files 24
- Boilerplate Files
  - samples 249
- Bulk Operations 112
  - ADD 112
  - DELETE 113
  - via the Web Administration Interface 191

## C

- Catalist
  - adding HTML to a list header 88
- Change Logs 174
  - logging information to a DBMS 182
  - tracking subscription changes 223
  - using to track distributions 181
- Cloning
  - lists 104
- command confirmation 219
- Command Confirmation Request 219, 225
  - cancelling the OK cookies 221
- Command Line Utilities
  - non-VM 27
- Commands 51
  - ADD 239, 240, 243, 246
  - ADDHere 243
  - advanced commands 62
  - AFD 239, 240, 242

- cancelling the OK cookies 221
- CMS 245, 246
- CONFIRM 236
- confirmation request 219
- CP 245
- DATABase 240, 245
- defining passwords 73
- DELeTe 239, 243, 246
- DISTRIBUTE 238, 240
- EXPLODE 243
- file management 65
- File Server 60
- FOR 239, 241, 243, 244, 245, 247
- for templates 140
- FREE 244
- FUI 239, 240, 242
- GET 238, 239, 240, 242, 244, 245, 246
- GIVE 239
- HELP
  - modifying 156
- Help 238
- HOLD 244
- INDEX 236, 240
- informational 58
- INSTALL 245
- List Management 66
- list related 55
- List Subscription 51
- Lists 236
- LISTSERV Maintainer 69
- MOVE 243, 244
- NODESGEN 246
- OFFLINE 246
- ONLINE 246
- PUT 238, 240, 242, 244, 246
- PUTC 246
- PW 240, 242, 246
- PWC 246
- Query 237, 238, 244, 246
- REFRESH 240, 242
- REGister 237, 247
- RELEASE 238, 239
- REView 237, 245
- SCAN 237
- SENDFile 247
- sending to LISTSERV 72
- sending via the Web Interface 194
- SENDme 240
- SERVE 221, 241, 247
- SET 235, 237, 245
- SF 247

- SHOW 238, 239, 247
- SHUTDOWN 245, 247
- SIGNOFF 235
- STOP 247
- SUBscribe 235
- THANKs 241
- UDD 241
- UNLOCK 242, 245
- Web Functions 60
- Control Characters
  - translating in list mail 229
- Cookies
  - cancelling the OK confirmation cookies 221
- D**
- Distribution Features 197
  - allowing MIME attachments 201
  - blocking MIME attachments 201
  - controlling PRIMETIME 197
  - controlling the default level of acknowledgement to user postings 197
  - controlling the list digest feature 200
  - controlling the maximum number of postings per day 197
  - defining list topics 200
  - freeing a list 199
  - holding a list 199
- DomainKeys 45
  - configuring DNS 46
  - configuring LISTSERV 48
  - creating 46
  - for lists 116
  - starting LISTSERV with DKIM support 48
  - using with LISTSERV 49
- Dynamic Templates 153
- E**
- Edited Lists
  - creating 92, 211
  - creating Private Edited Lists 96
- Error Handling Features 203
  - address probing
    - active 207
    - operating system issues 209
    - passive 208
  - auto-deletion 203
  - defining list-level error handling addresses 203
  - defining server-level addresses 209
  - loop detection 204
  - RFC822 Mail Header Parsing 205
- External Data Files 25
- F**
- File Archives 117
  - adding FAC Codes
    - VM Systems 118
  - adding file descriptions to the Filelist
    - VM Systems 119
  - adding files to the SITE.CATALOG 122
  - creating a filelists
    - VM Systems 118
  - creating a sub-catalog 124
  - delegating file management authority 123
  - deleting file descriptions from the Filelist
    - VM Systems 121
  - deleting files from the Host Machine 127
  - file access codes for user access
    - VM Systems 120
  - indexing the sub-catalog 126
  - listname.CATALOG System
    - Non-VM System 121
  - retrieving the Filelist
    - VM Systems 118
  - starting for your list 117
  - storing files on the Host Machine 126
  - storing the Filelist
    - VM Systems 121
  - updating the Sub-Catalog 124
- File Packages 128
- File Update Information 127
- Filtering
  - list content 113
- G**
- GUI Site Configuration Utility
  - Windows 28
- H**
- Host Machine
  - deleting files from 127
  - storing files on 126
  - storing the listname.MAILTPL file 151
- Host Name Registration
  - using ALIASES NAMES 44
- I**
- ISP Functionality 233
  - directory quotas for lists 233
  - limiting the number of subscribers to a list 234



- J**
- K**
- L**
- List Configuration
    - via the Web Interface 192
  - List Exits
    - POST\_FILTER 222
  - List Header
    - adding HTML for the CataList 88
    - defining list topics 200
    - hiding selected lines 222
    - updating 89
  - List Header Keywords 82
    - CHANGE-LOG= 223
    - command confirmation 219
    - DEFAULT-OPTIONS= 225
    - FILTER= 221
    - LIMITS= 234
    - New-List 213
    - NOTEBOOK= 217
    - PW= 240, 242, 246
    - RENEWAL= 226
    - REVIEW= 217
    - SEND= 218
    - SERVICE= 216
    - SUBSCRIPTION= 216, 225
    - VALIDATE= 215
  - List of Lists 236
  - List Owners
    - ALL-REQUEST alias 230
    - communicating with 229
    - listname-REQUEST alias 230
    - sample "boilerplate" files 249
  - List Topics
    - defining 200
  - Lists
    - adding a list password 85
    - adding a new subscriber via the Web Interface 190
    - adding HTML to a list header for the CataList 88
    - aliases needed for PMDF 231
    - backup 86
    - bulk operation
      - ADD 112
    - bulk operations 112
      - DELETE 113
    - changing the name of an existing list 111
    - cloning 104
    - configuring a list via the Web Interface 192
    - controlling subscription requests 216
      - controlling the list digest feature 200
      - controlling the service area 216
      - controlling who can post to a list 218
      - controlling who reviews the list of subscribers 217
    - creating 75
      - checklist 79
    - creating Announce-Only Lists 97
    - creating Auto-Responder Lists 96
    - creating Edited Lists 92, 211
    - creating Moderated Lists 93, 96, 211
    - creating Peered Lists 99
    - creating Private Discussion Lists 91
    - creating Private Edited lists 96
    - creating Public Discussion Lists 90
    - creating Restricted Subscription Lists
      - with auto-generated questionnaire 98
    - creating Self-Moderated Lists 95
    - creating Semi-Moderated Lists 95
    - creating split lists 99
    - creating Sub-Lists 102
    - creating Super Lists 102
    - defining error handling addresses 203
    - deleting 87
    - deleting a subscription via the Web Interface 189
    - denying service to problem users 221
    - directory quotas 233
    - editing 83
    - enabling 35
    - examining a subscription via the Web Interface 189
    - filtering content 113
    - freeing 199
    - holding 199
    - limiting the number of subscribers to a list 234
    - linking 89
    - linking two or more 100
    - maintaining 75
    - maintenance via the Web Interface 189
    - merging 106
    - migrating from databases 111
    - migrating from one site another 107
    - migrating from Sendmail Alias Files 111
    - moving to a new location 213
    - moving users from one (peer) server to another 100
    - naming conventions 80
      - "L" convention 80
      - maximum length of name 82
      - reserved characters 81
      - reserved names 80
      - user-friendly 82

- OpenVMS
    - creating required PMDF Aliases 79
    - restricting the number of posts 212
    - restricting the size of posted messages 212
    - reviewing 83
    - sample list header file
      - List Header File
        - sample 86
    - servicing up custom web pages 155
    - setting up a Notebook Archive 129
    - storing on the host machine 85
    - translating control characters included in mailings 229
  - Unix
    - creating required Sendmail Aliases 77
  - LISTSERV
    - adding a postmaster 10
    - changing a postmaster 10
    - command line utilities for non-VM 27
    - configuring to activate the web archive interface 34
    - configuring your site 13
    - defining passwords 73
    - deleting a postmaster 10
    - enabling individual lists 35
    - enabling web-based bulk operations 37
    - external data files 25
    - GUI site configuration utility for
      - Windows 28
    - initial configuration 10
    - install 9
    - installing a web server 32
    - installing the administration interface 30
    - installing the web archive interface script 32
    - installing the www archive interface 30
    - inter-server updates 42
    - principles of operation 7
    - program executables 23
    - reference material 26
    - required files 23
    - sending commands to 72
    - starting 10
    - stopping 10
    - supported operating systems 5
  - LISTSERV Maintainer
    - see *Postmaster*
  - LISTSERV vs LISYSERV Lite 4
  - Log Files
    - Administration Interface logging 169
    - administrative mail 163
    - change logs 174
    - cleaning your log files 159
    - command forwarded via GLX 168
    - content filter rejection message 173
    - daily error monitoring reports 162
    - DISTRIBUTE jobs from remote hosts 164
    - distributing a digest 161
    - expiring cookies 161
    - extracting server statistics 176
    - FIOC cache notifications 169
    - global list of lists updates 165
    - interpreting 160
    - interpreting the output of SHOW CTR 179
    - interpreting the SMTP logs
      - Windows Servers 173
    - interpreting the SMTP worker log entries
      - Non-VM 174
    - invalid confirmation received 167
    - kept by LISYSERV 159
    - logging changelog information to
      - DBMS 182
    - making daily logs 159
    - managing 159
    - MIME parser messages 171
    - netwide DELETE 168
    - non-command text in mailings to
      - LISTSERV 167
    - processing mail for local lists 163
    - reindexing a list 161
    - releasing and reallocating a disk slot 161
    - requesting confirmation for
      - commands 164
    - response to list owner or maintainer
      - commands 168
    - response to posts to a HELD or PRIMETIME list 168
    - sample log processing scripts 176
    - spam alerts 170
    - subscription summary updates 164
    - user already subscribed to a given list 167
    - using the system changelog to track
      - distributions 181
    - valid confirmation received 166
    - Web Archive logging 169
  - Logs Files
    - interpreting 159
  - Look Detection
    - anti-spamming filter 205
  - Loop Detection 204
- M**
- Mail Templates 136
  - Mail-Merge
    - using the Web Interface 194
  - Merging

- lists 106
  - Message Fragments 137
  - Migrating
    - lists from databases 111
    - lists from one site to another 107
    - lists from Sendmail Alias Files 111
  - MIME Attachments
    - allowing 201
    - blocking 201
  - Moderated Lists
    - creating 93, 211
    - creating Private Moderated Lists 96
- N**
- Network Table Files
    - BITNET 24
    - Internet and Peer 25
  - Notebook Archives 117, 129
    - controlling access 217
    - deleting 132
    - indexing 133
    - migrating to a new site 130
    - setting up for a list 129
  - Notebook Directories
    - setting up 43
- O**
- P**
- parameters 242, 247
  - Passwords
    - defining 73
    - setting for the Web Administration Interface 187
  - Peered Lists
    - creating 99
    - linking two or more 100
    - moving users from one server to another 100
  - PMDF
    - creating required aliases
      - OpenVMS 79
      - unix 231
      - VMS 231
    - required configuration
      - for Unix servers 231
      - for VMS servers 231
  - Postings
    - controlling PRIMETIME 197
    - controlling the default level of acknowledgement 197
    - controlling the list digest feature 200
    - controlling the maximum number per day 197
    - freeing 199
    - holding 199
    - restricting the number of posts 212
    - restricting the size of posted messages 212
  - Postmaster
    - adding 10
    - changing 10
    - creating a password for 11
    - deleting 10
  - Preferences
    - setting in the Web Administration Interface 187
  - Private Discussion Lists
    - creating 91
  - Private Edited Lists
    - creating 96
  - Program Executables 23
  - Public Discussion Lists
    - creating 90
- Q**
- R**
- Restricted Subscriptions Lists
    - creating with auto-generated questionnaire 98
  - RFC822 240
    - mail header parsing 205
- S**
- Security 215, 219
    - cancelling the OK cookies 221
    - controlling access to the notebook files 217
    - controlling subscription requests 216
    - controlling the service area of a list 216
    - controlling who can post to a list 218
    - controlling who reviews the list of subscribers 217
    - denying service to problem users 221
    - hiding selected header lines 222
    - tracking subscription changes 223
    - validation level 215
  - Self-Moderated Lists
    - creating 95
  - Semi-Moderated Lists
    - creating 95
  - Server Administration Dashboard 195
  - Server Administration Interface 195
  - Server Registration 40
    - automatic registration for Lite servers 42

- Classic HPO servers 40
  - finding an already registered server 40
  - LISTSERV Classic servers 40
  - participating in the LISTSERV
    - Backbone 41
  - Site Configuration Files 13, 23
  - Site Configuration Keywords
    - DATABASE 240
    - NODE 237
  - Site Configuration Variables 14
    - FILTER\_ALSO= 221
  - Spam
    - alerts 38
    - anti-spamming filter 205
    - log files 170
    - quarantine 38
    - subscription anti-spoofing feature 39
  - Split Lists
    - creating 99
    - linking two or more 100
    - moving users from one server to another 100
  - Sub-Catalog
    - creating 124
    - indexing 126
    - updating 124
  - Sub-Lists
    - creating 102
  - Subscription Anti-Spoofing Feature 39
  - Subscriptions
    - defining default options for subscribers 225
    - setting up confirmation 225
    - setting up renewals 226
  - Super Lists
    - creating 102
- T**
- Templates
    - \$SITE\$.MAILTPL file 158
    - calculating the value for DAYSEQ 150
    - commands 140
    - common variable substitutions 138
    - condition processing 143
    - creating 135
    - customizing the web pages 35
    - DEFAULT MAILTPL 147
    - DEFAULT.WWWPTL file 153
    - DIGEST-H file 151
    - dynamic 153
    - editing 135
    - editing list-level default templates 146
    - for the WWW Interface 152
    - forms 152
    - getting copies of the Default Template File 136
    - INDEX-H file 151
    - INFO template form 146
    - mail template formats 138
    - maintaining via the Web Interface 193
    - modifying the HELP Command 156
    - naming conventions 137
    - national language template files 154
    - precedence 154
    - rotating the bottom banner 149
    - serving up custom web pages for your list 155
    - setting up National Language mail templates 229
    - SITE.WWWTPL file 153
    - storing the listname.MAILTPL file on the Host Machine 151
    - subscription renewal 150
    - tips for using 148
    - types
      - Mail 136
      - Message
        - Message Templates 136
        - Message Fragments 137
      - using 8-bit characters 145
      - using the DAYSEQ(n) function 149
- U**
- V**
- VM vs Non-VM 3
- W**
- Web Administration Interface
    - adding a new subscriber 190
    - bulk operations 191
    - configuring a list 192
    - deleting a subscription 189
    - examining a subscription 189
    - list maintenance 189
    - logging in 185, 187
    - mail-merge 194
    - maintaining templates 193
    - sending interactive commands 194
    - setting a password 187
    - setting preferences 187
    - the default home page 185
    - using 185
  - Web Archive Interface Script
    - installing 32

Web Pages  
    customizing 35  
    serving up custom pages for your list 155  
Web Server  
    installing 32  
Web-Based Bulk Operations  
    enabling 37  
WWW Archive Interface  
    creating a subdirectory 34  
    installing 30  
    log files 169  
WWW Interface Templates 152

**X**

**Y**

**Z**

