SPEC OSG Mailserver Subcommittee
SPECmail2009 Benchmark Architecture White Paper

Revision: v1.0 **Date:** 22 December 2008

1. Introduction

1.1 Overview

SPECmail2009 is a software benchmark designed to measure a system's ability to act as an enterprise mail server servicing email requests, based on the Internet standard protocols SMTP and IMAP4. The benchmark concentrates on the workload encountered by corporate mail servers, with an overall user count in the range of 150 to 10,000 (or more) users. It models IMAP business users accessing IMAP servers over fast local area networks (LAN) instead of broadband, WAN or dialup access speeds. Two separate metrics measure e-mail servers with and without secure network connections (SSL).

SPECmail2009 has been developed by the Standard Performance Evaluation Corporation (SPEC), a non-profit group of computer vendors, system integrators, universities, research organizations, publishers, and consultants.

This paper discusses the benchmark principles and architecture, and the rationale behind the key design decisions. It also outlines the workload used in the benchmark, and the general steps needed to run a benchmark. However those aspects are covered in more detail in other documents.

1.2 Organization of this Paper

Chapter 2 discusses the basic goals and non-goals of the benchmark.

Chapter 3 introduces the two performance metrics for SPECmail2009 – IMAP sessions per hour - and how it relates to the transaction mix imposed on the system under test..

Chapter 4 explains the benchmark workload - how it was derived, how it translates into configuration parameters for the benchmark tool and size calculations for planning a benchmark, and how it relates to the benchmark metric.

Chapter 5 discusses some detail of aspects of the workload generation, namely the exact workload put on the server, and how the benchmark simulates communication with remote mail servers.

Chapter 6 defines the quality of service requirements of this benchmark.

Chapter 7 lists the references and sources (not cited elsewhere).

1.3 Related Documents

- *SPECmail2009 Run and Reporting Rules*
- *Workload Analysis for Enterprise Mail Servers*
- *SPECmail2009 User Guide*
- *SPECmail2009 Sample Result Disclosure*
- *SPECmail2009 FAQ*

All documents can be obtained from SPEC's SPECmail2009 benchmark documentation web links off of http://www.spec.org/mail2009/.

1.4 Run and Reporting Rules

The Run and Reporting Rules for the SPECmail2009 benchmark are spelled out in a separate document. They ensure execution of the benchmark in a controlled environment. The goal is repeatability by third parties with reasonable resources and knowledge. The rules maximize the comparability of results, leveling the playing field as much as possible. They also define which information needs to be included in published results, and which supporting information needs to be submitted to the SPEC community for potential review.

Under the terms of the SPEC license, SPECmail2009 results may not be publicly reported unless they are run in compliance with the Run and Reporting Rules. Results published at the SPEC web site have been reviewed and accepted by the SPEC Mail Server committee. For more information on publishing results at the SPEC web site, please send e-mail to: info@spec.org. The Run and Reporting Rules may be found on the SPEC web site; they are also part of the SPECmail2009 distribution kit.

2. Design of the SPECmail2009 Benchmark

SPECmail2009 benchmark tests the capacity of a system as an e-mail service that processes requests according to the Internet standard mail protocols ***SMTP*** (RFC 821) and ***IMAP4*** (RFC 2040). The SMTP protocol is the standard for sending e-mail from clients (users) to servers and between e-mail servers. The IMAP4 protocol allows users to access and retrieve messages from their message store. The mail server can consist of one host or a group of hosts that act as a single, logical entity – usually represented by a single e-mail domain.

The SPECmail_MSEnt2009 metric's user model describes a corporate employee that uses one of the popular IMAP4 clients to access either a mail server located within a local area network (LAN), or an outsourced e-mail service across a high-speed network connection (MAN). The details of an enterprise-type user's behavior will be discussed in a later section in this paper that covers the SMTP and IMAP work load profiles.

In this benchmark, both the Mail Server User behavior and the IMAP4 e-mail client software vary greatly. Therefore, it is very important to identify and distinguish actual IMAP4 client's human initiated actions from automated actions performed on behalf of each user. The specific combination of these behavior types determines how many users a mail server can handle.

SPECmail2009 simulates the work loads of four types of IMAP4 e-mail clients, defined in a fixed proportion across the user population. The benchmark observes the mail server behavior under that load. It enforces the adherence to a required level of quality of service. The goal is to simulate realistic mail server operation, and maximize the usefulness of the benchmark results as guidelines for actual sizing decisions.

2.1 Requirements and Goals

The key goal of SPECmail2009 is to show mail server performance in a realistic context. This means

Appropriate Mail Store	A mail server which handles the transaction load of a corporate user base needs to hold the mail data for the same amount of users. This includes a mailbox per user, as well as a defined number of folders and messages for each user.
Mail Store Folder Structures	The benchmark includes the construction and pre-population of multiple folders (IMAP *mailboxes*) and subfolders as part of the compliance requirements. Since IMAP is a server-centric storage service, access and manipulation of folders plays a critical part of the overall workload.
Message MIME Structures	The benchmark includes the construction and manipulation of e-mail messages with both simple and more complex MIME (messages with attachments) structures. Unlike POP3 servers, IMAP clients understand and can manipulate multi-part MIME messages.
Arrival Rate	Message activity requests are being issued against the server according to behavior patterns and arrival rate modeled after real world observations.

Access over Local Area and/or Broadband Networks All users will access the IMAP and SMTP servers via internal corporate or broadband network speeds.

Workload According to Real World Data The simulated workload (such as messages sent and received per day, the message size distribution, the folder structure distribution, the mailbox access frequency, message manipulation) is based on measurements derived from multiple data sources.

Peak Hour Simulation Mail traffic is distributed unevenly over the day. Sizing must focus on coping with the peak load. Therefore, the benchmark mail server load simulates the peak hour of the day.

Realistic Operation The mail server is required to operate realistically, e.g. perform at least a defined level of logging which many ISPs would use in practice. As is the rule in all SPEC benchmarks, no benchmark-specific optimizations are allowed.

2.2 Excluded Goals

Explicitly excluded goals and limitations in the scope of SPECmail2009 are:

Restricted client IP Range The benchmark does not require that there be 100's of small, remotely connected email clients. Instead, it allows simulation of the email clients on a small number of server-sized, locally connected client systems, in order to make the benchmark execution practical.

Administrative Overhead The benchmark does not include administrational activities like on-line backup and account provisioning. These are hard to standardize, and - more importantly - they do not necessarily happen during the peak hour of daily operation.

Provisioning Overhead Besides provisioning, the benchmark does not include modification to account data or deletion of accounts. These activities, although commonplace in the day-to-day operation of a mail server, are seen as testing the directory service rather than the mail server itself. SPECmail2009 emphasizes mail server benchmarking, and includes a directory server in the system-under-test only to the minimum extent necessary to handle normal mail flow.

Pure Relay SMTP Traffic The benchmark does not cover relay operation (forwarding of incoming SMTP traffic to other, remote MTAs). It does include simulation of messages from local users to remote MTAs, as well as incoming mail from remote MTAs. Relay operation is generally not allowed on enterprise e-mail systems.

High Availability Overhead There is no requirement for high availability or disaster recovery measures; this was beyond the scope of the benchmark.

Extra Content Filter/Processing, Security The mail server is not required to perform any extended features, like virus scanning, spam filtering and other security measures. This was beyond the scope of the benchmark.

SMTP Streamlining SMTP sessions can normally send one or several messages per session. The benchmark restricts itself to one message. The collected SMTP log data shows that this behavior dominates internally and externally generated e-mail traffic.

3. Benchmark Metrics

3.1 Unencrypted Benchmark Metric: SPECmail_MSEnt2009

SPECmail2009 Enterprise IMAP4 Sessions Per Hour

The basis of the benchmark's performance metric is *capacity at acceptable quality of service (QoS)*. The benchmark determines *Acceptable QoS* by measuring the interactive response time of the mail server to each protocol step (see Chapter 6). The boundary where a one or more critical states exceeds the *Acceptable QoS* determines how many users the SUT supports under this specific work load profile and over unencrypted network connections.

The IMAP protocol allows many combinations of session behavior and duration. Each IMAP user generate many IMAP sessions to perform different tasks – both in parallel and serially over the peak hour period. The number of both the actual and concurrent sessions varies by IMAP client software – usually some multiple of the active IMAP users across the specific IMAP client type distribution.

The SPECmail2009 benchmark uses a specific distribution of the four (4) IMAP4 client types that a mail server must support during the peak hour. This transaction mix is defined later on in this document. In general, the following conditions exist:

User Mail Store	Each user has at least three (3) folders – Inbox, Trash, Sent_Mail Each user has existing messages stored in these folders
SMTP Traffic	Each user send 5 messages during Peak Hour to 3.0.6 average recipients Each user receives 5.06 messages during Peak Hour
IMAP Session	Each user establishes at least one long lived session before the peak hour Some short lived sessions are initiated by human actions Some short lived sessions are initiated by automated tasks
Folder Activity	Each user polls one or more folders in each session at least once Some users will created new and/or delete old folders Some users will move messages between folders
Message Activity	Each user will retrieve all new messages during the Peak Hour Some users will also retrieve older messages Each user might delete some messages, both old and new A few users might search a folder

This metric cannot be compared to any other similarly named benchmark metric which does not follow exactly that workload definition and the same execution rules. Every aspect of each may affect the benchmark outcome.

3.2 Encrypted Benchmark Metric: SPECmail_MSEnt2009Secure

SPECmail2009 Enterprise IMAP4 Sessions Per Hour over TLS Connections

This benchmark variation uses the same work load profile as the unencrypted version but also incurs the additional cost of encrypting all transmitted data. It is possible to score a lower value due to this computing overhead.

4. IMAP4 Benchmark Decisions

4.1 Command Set

The various versions of IMAP4 RFCs as well as any number of optional extended commands made an analysis of the IMAP4 variants problematic. Because of the variety of client and server combinations, the subcommittee used the following guidelines to determine the IMAP commands used.

Decision Point	Justification
Restrict Command Set to RFC 2040 and ignore extended IMAP commands	The core IMAP commands in RFC 2040 was supported by all e-mail servers and clients. However, support for extended IMAP commands varied by both mail server and mail clients.
Reduce Parameters to Most Common Forms	Identify the most common forms of various IMAP commands. Identify the most frequent parameter combinations, ignoring sequence, to these commands. Map certain parameters to an equivalent form.
Ignored Commands	Ignore certain extended IMAP4 commands that affect/query meta-data information, and is not directly related to messages or folders. These include `CAPABILITY`, `GETACL`, `MYRIGHT`, and `NAMESPACE`.
Map Extended Commands to Base RFC Version	Certain extended IMAP4 commands duplicate functionality found in the original RFC 2040 command set. These commands were mapped accordingly: `AUTHENTICATE` to `LOGIN` and `IDLE/DONE` combination to `NOOP`.

4.2 Mail Store Structures

The various versions of IMAP4 RFCs as well as any number of optional extended commands made this analysis mandatory. The decision was to stay with a very strict interpretation of RFC 2040 and treat some of the extended commands in the following manner.

Mail Store Structure	Implementation Decisions
Message Set References	Some commands used message identifiers that were a combination of a range (A:C) and a discrete set (A,C,F). These will be mapped to just a message identifier range (A:C).
Specified Headers vs. Number of Headers	The message retrieval requests for specific headers incur the same cost regardless of which headers are retrieved. Therefore, certain headers that are not listed in RFC821 will be generalized into a small subset.
Restrict Folder Depth References to Five (5) Levels	The extremely low probability rates beyond Level 5 were collapsed into a single row at Level 5.
Reserved Folder Names	All IMAP4 client types referenced three folder names: Inbox, Inbox.Trash, Inbox.Sent. These will be created for all users as part of the mail store initialization step.
Message MIME Structure	Unlike POP3 where messages are retrieved in whole, a

and Part Sizes Predominate	significant number of IMAP4 commands retrieved specific MIME attachments. This means that message structures must exist. Once this condition exists, it is almost impossible to comply with the message size distributions defined by the SMTP logs. The choice was made to let message structure and attachment sizes be the primary factors, overriding raw message sizes.
Reduce MIME Depths to only Five (5) Levels	As with the folder nesting levels, depths below five (5) were extremely low probability. These were mapped to Level 5.

4.3 Compliant Run

The SPECmail2009 benchmark compliant conditions are as follows:

User Count	Set to at least **250** users in order to meet the folder structure distribution.
Quality of Service	remains at **5** seconds for "simple" commands
Encryption	determines which metric is published – unencrypted or encrypted TLS network connections.

Setting the IMAP4 and SMTP Secure Connections configuration key generates the MSEnt2009Secure metric in the disclosure. Any other combination generates the MSEnt2009 metric in the disclosure.

5. Benchmark Workload

The SPECmail2009 workload has two parts: the pre-population of the mail store with folders, sub-folders and messages, as well as the transaction workload during runtime. Both depend on the targeted benchmark rating.

It may be helpful at this point to list the basic steps of running a SPECmail2009 benchmark.

Benchmark Step	What Happens
Initialize Mail Store	Pre-populate the mail store according to the defined folder and message MIME distributions. An alternative is to restore a backup copy of a previously initialized mail server.
Restart SUT	Start from a cold system (processes, file systems, databases), so that all operating system caches are clean.
Verify Mail Store Compliance	Verify mail store has the desired folder, message quantity and message MIME structure distributions across all mailboxes.
Gather Mailbox Context	Benchmark gathers detailed information about actual mail store folder names and valid messages to be used later to generate valid IMAP commands.
Ramp-up Period	Benchmark runs for a configurable time at peak hour load levels, but no measurements are recorded. During this time, the benchmark establishes the pre-existing Command Sequence 1 and 2 sessions.
Peak Hour Load Test at 100%t	Internal results counters are cleared (except existing IMAP sessions) and the benchmark manager starts recording results returned by the load generators for the work load period.
Ramp-down Period	Benchmark finishes all existing IMAP sessions but does not initiate new sessions. Primary sessions (long lived) sessions log out of the mail server.
Record Results	Tabulate all collected results and record certain statistics to the official results file.
Report Generation	Use distributed reporter to generate human readable form of the benchmark test results.

The actual process is a bit more complex - refer to the Run and Reporting Rules for details.

The following sections discuss both configurable and embedded configuration keys that determine the benchmark's behavior and workload. Configuration key names are defined in ALL_UPPER_CASE. Names using UpperAndLower case are inserted only for editorial purposes.

5.1 Basis of Workload Definition

The workload profile has been determined based on actual SMTP and IMAP4 server log files gathered from multiple corporate sources. The IMAP4 data covers e-email traffic from two (2) universities, three (3) corporate e-mail servers and one (1) outsourced e-mail service. Parts of the profile remains unchanged from the SPECmail2008 benchmark. Message store data provided by Apple, Inc. replaced the Sun mailbox and message profile. Apple's SMTP traffic volumes are also incorporated into the SMTP volume, but did not contain enough information to change the message routing distribution.

The SMTP workload data extracted from MTA log files includes message arrival rates, recipient counts, message sizes and routing (local vs. remote). The recipient distribution includes exploded mailing lists as well as individually addressed recipients. (Sources: Mirapoint, Openwave)

The mailbox structures and contents were derived from the list of actual folders found during a mail server mail store survey. Every user's mail store folders were listed along with a count of the messages and subfolders inside. (Sources: Apple)

The message MIME structures and content types were derived from a complete snapshot of an E-mail server's message structures. (Source: Apple)

5.1.1 Overall Comparison to SPECmail2008

The SPECmail2009 benchmark changed both folder-to-folder and message attachment distributions. The folder tree is wider but shallower. The message attachments were larger, deeper, wider and just plain more of them. The benchmark also no longer pads extra text in an attempt to meet the message size distribution – a legacy left over from SPECmail2001.

Due to the sparser folder and message elaborations, the benchmark requires slightly more users for a minimal compliant run – from 200 to 250 users. The benchmark also uses variable compliance requirements based on number of users, derived from experimental experience.

The overall effect on disk storage is less space per user. The primary cause lies in fewer folders with which to elaborate the folder distribution. This *reduces* message disk space from 450 MB to 160 MB per user.

5.2 Non-Transaction Related Definitions

The typical Enterprise E-Mail server with IMAP users holds user messages stored in one or more folders. Unlike the POP3 e-mail clients, IMAP e-mail clients work with both new and existing messages. This means the folder structures and messages must already exist before the work load can start.

Mail Store Structures The SPECmail2009 benchmark defines a mail store structure model that replicates a complex mail store structure, derived from data collected from Apple. Some folders have up to two thousand (2000) messages inside. The mail server pre-population phase (-init) ensures that a system can handle the

	transaction and storage load for 250 Enterprise users. The pre-population uses both a folder distribution and a message distribution across all user mail stores.
Mail Server Storage Growth	The SPECmail2009 benchmark is designed to generate a steady mail server state, over time. However the very nature of a Server-side storage model means that the benchmark is ***not*** storage neutral. Collected IMAP session samples show fewer messages were deleted than created during the workload period. Note: message insertion and deletion happens in folders that are randomly and independently selected. Therefore, mailbox structures and content change during the load test period.
Message MIME Structures	The IMAP protocol understands the concept of MIME structures and many data samples show references to message substructures. This means these structures must exist to during the peak hour work load.

5.2.1 Folder Structures

A mail server that supports IMAP is likely to support a hierarchy of several mailboxes (also known folders) in addition to the default INBOX mailbox for each user. Below are several distributions to construct the structure of mailboxes contained within a mailstore supported by IMAP. The revised folder hierarchy data is extracted from a single large enterprise data sample (Apple).

Comparison to SPECmail2008: This benchmark folder hierarchy creates fewer folders than the previous version. For example, the current `LEVELXFOLDERS[0][0]` entry is 42.2% instead of the original 34.8%. This means 7.4% more users have only a single folder at the top level. Also at `LEVELXWITHSUB[0][0]`, folders have 96% (current) versus 59% (original) chance of zero (0) subfolder.

Configuration Key	Definition and Value
LEVELXFOLDERS[][]	Defines the probability that a mail store has one or more folders, at various depth.
`LEVELXFOLDERS[0]`	`"1,0.38%; 2,0.71%; 3,41.11%; 4,17.15%; 5,8.48%; 6,5.59%; 7,4.01%; 8,3.24%; 9,2.66%; 10,2.04%; 15,6.57%; 20,3.28%; 25,1.77%; 50,2.49%; 100,0.52%"`
`LEVELXFOLDERS[1]`	`"1,37.28%; 2,14.13%; 3,12.26%; 4,6.60%; 5,5.41%; 6,4.09%; 7,3.14%; 8,2.57%; 9,2.08%; 10,1.66%; 15,4.97%; 20,2.40%; 25,1.17%; 50,1.66%; 100,0.58%"`
`LEVELXFOLDERS[2]`	`"1,38.86%; 2,17.28%; 3,10.23%; 4,7.07%; 5,5.13%; 6,3.69%; 7,3.06%; 8,2.18%; 9,1.97%; 10,1.77%; 15,4.07%; 20,1.94%; 25,0.77%; 50,1.66%; 100,0.32%"`
`LEVELXFOLDERS[3]`	`"1,41.69%; 2,17.26%; 3,10.82%; 4,6.71%; 5,5.30%; 6,3.56%; 7,2.51%; 8,1.78%; 9,1.74%; 10,1.10%; 15,3.79%; 20,2.15%; 25,0.64%; 50,0.95%"`

Configuration Key	Definition and Value
LEVELXFOLDERS[4]	"1,37.52%; 2,23.47%; 3,13.88%; 4,6.94%; 5,3.47%; 6,3.64%; 7,1.98%; 8,1.16%; 9,0.99%; 10,1.16%; 15,2.64%; 20,1.32%; 25,1.32%; 50,0.51%"
LEVELXFOLDERS[5]	"1,42.98%; 2,23.14%; 3,9.92%; 4,8.26%; 6,4.96%; 10,10.74%
LEVELXFOLDERS[6]	"1,35.42%; 2,16.67%; 3,22.92%; 4,12.50%; 5,12.49"
LEVELXFOLDERS[7]	"1,20.00%; 2,36.00%; 3,24.00%; 4,12.00%; 5,8.00%"
LEVELXWITHSUB[][]	Defines the probability distribution that a folder at each depth (indicated by the array index) has zero or more subfolders:
LEVELXWITHSUB[0]	"0,94.58%; 1,2.02%; 2,0.77%; 3,0.67%; 4,0.36%; 5,0.29%; 6,0.22%; 7,0.17%; 8,0.14%; 9,0.11%; 10,0.09%; 20,0.58%"
LEVELXWITHSUB[1]	"0,90.62%; 1,3.65%; 2,1.62%; 3,0.96%; 4,0.66%; 5,0.48%; 6,0.35%; 7,0.29%; 8,0.20%; 9,0.18%; 10,0.17%; 15,0.82%"
LEVELXWITHSUB[2]	"0,92.14%; 1,3.28%; 2,1.36%; 3,0.85%; 4,0.53%; 5,0.42%; 6,0.28%; 7,0.20%; 8,0.14%; 9,0.14%; 10,0.66%"
LEVELXWITHSUB[3]	"0,92.81%; 1,2.70%; 2,1.69%; 3,1.00%; 5,0.75%; 10,1.05%"
LEVELXWITHSUB[4]	"0,95.25%; 1,2.04%; 2,1.10%; 5,0.98%; 10,0.63%"
LEVELXWITHSUB[5]	"0,87.63%; 1,4.38%; 2,2.06%; 3,2.84%; 5,3.09%"
LEVELXWITHSUB[6]	"0,82.39%; 1,3.52%; 2,6.34%; 3,4.23%; 4,3.52%"
LEVELXWITHSUB[7]	"0,100.00%"

The folder probability distribution exists for eight levels. However due to the much lower probability of deeper subfolders, a graduated floor is used for the message store compliance verification test. The smaller the user count, the shallower the subfolder level used for message store verification.

5.2.2 Folder Message Population

The second stage message store initialization populates the created folders with some "old" messages. Each folder level has its own distribution set of messages and probability. The SPECmail2009 benchmark uses a Messages per Folder Distribution array of number pairs. Each row in this array corresponds to the *Folder Level.* Each pair defines a message count and probability.

Comparison to SPECmail2008: Another side effect of this benchmark's sparser folder hierarchy is the fewer number of subfolders available to fulfill the deeper messages per folder distributions. Therefore, SPECmail2009 uses a graduated messages-per-folder compliance scale. The smaller the user count, the shallower the folder depth used to judge messages-to-folder distribution compliance.

Configuration Key	Definition and Value
LEVELXMESSAGES[][]	Defines the probability distribution of how many messages reside in a folder by level (indicated by the array index). The couplet's left side indicates a message count. The couplet's right side is the probability. The number of rows should match the LEVELXFOLDERS array.
`LEVELXMESSAGES[0]`	`"0,0.84%; 5,4.90%; 12,5.00%; 22,5.07%; 35,5.19%; 51,5.01%; 70,5.15%; 95,5.16%; 127,5.10%; 165,5.09%; 212,5.01%; 274,5.02%; 356,5.05%; 466,5.03%; 623,5.02%; 855,5.01%; 1232,5.01%; 1922,5.00%; 3275,5.01%; 4000,8.33%"`
`LEVELXMESSAGES[1]`	`"0,32.83%; 1,6.79%; 3,6.89%; 6,6.08%; 10,5.27%; 16,5.28%; 25,5.03%; 40,5.21%; 65,5.01%; 111,5.00%; 212,5.01%; 524,5.00%; 2577,5.00%; 3000,1.60%"`
`LEVELXMESSAGES[2]`	`"0,15.35%; 1,8.21%; 2,5.70%; 4,7.83%; 6,5.53%; 9,6.08%; 13,5.73%; 18,5.19%; 25,5.09%; 35,5.07%; 51,5.06%; 77,5.06%; 126,5.05%; 239,5.00%; 654,5.00%; 2000,5.05%"`
`LEVELXMESSAGES[3]`	`"0,10.21%; 1,11.06%; 2,6.40%; 4,8.88%; 6,6.48%; 8,5.31%; 11,5.71%; 15,5.86%; 20,5.28%; 27,5.15%; 38,5.27%; 56,5.15%; 91,5.06%; 169,5.01%; 462,5.00%; 1000,4.17%"`
`LEVELXMESSAGES[4]`	`"0,9.45%; 1,9.64%; 2,7.93%; 3,6.06%; 4,5.14%; 6,8.62%; 8,6.25%; 11,6.61%; 14,5.47%; 19,5.95%; 26,5.51%; 36,5.12%; 55,5.11%; 104,5.05%; 359,5.01%; 500,3.08%"`
`LEVELXMESSAGES[5]`	`"0,6.67%; 1,8.51%; 2,6.82%; 3,5.45%; 4,8.82%; 5,6.20%; 6,6.04%; 8,7.84%; 10,5.76%; 13,6.47%; 17,5.88%; 24,5.25%; 35,5.06%; 54,5.06%; 114,5.02%; 2000,5.15%"`
`LEVELXMESSAGES[6]`	`"0,5.93%; 1,8.25%; 2,10.05%; 3,5.67%; 4,5.15%; 6,7.99%; 9,6.44%; 13,6.19%; 17,5.41%; 21,5.41%; 28,5.93%; 38,5.15%; 55,5.15%; 89,5.15%; 179,5.15%; 592,5.15%; 800,1.83%"`
`LEVELXMESSAGES[7]`	`"0,7.04%; 1,5.63%; 3,12.68%; 5,8.45%; 7,8.45%; 10,5.63%; 14,7.04%; 16,5.63%; 20,7.04%; 28,7.75%; 35,5.63%; 43,5.63%; 81,5.63%; 159,5.63%; 634,2.14%"`

5.2.3 Message Construction

The SPECmail2009 benchmark generates e-mail messages on the fly instead of using a fixed set of messages. Construction of each message follows these steps:

Construction Step	Explanation
Determine Number of MIME Parts at Top Level	Using the `MIME_TOP_PART_COUNT` distribution, compute how many overall parts make up the message.
Determine Each Part's Substructure	Using the `MIME_PART_LEVEL` distribution, compute the number of parts each Top level MIME part contains. No effort is made to distinguish between primary and alternate parts since this work occurs on the client, not server.
Determine Each Part's Size	Using the `MIME_PART_SIZE` distribution, compute the size of each individual MIME part. This is the primary message size factor.

SPECmail2009 prioritizes message structural and attachment size distributions over the total message size distribution, as did the SPECmail2008 benchmark. IMAP4 e-mail clients manipulate attachments and expect the e-mail server to provide message parts on demand. E-mail servers must evaluate the actual structure of each message at some point in the processing. Therefore, message MIME structure and types incur processing costs.

Comparison to SPECmail2008: The SPECmail2009 benchmark MIME structures and attachment sizes reflect the change from simple text messages to e-mail messages with both rich and alternative simple text content. The older data sample had 75.77% messages with only a single part. The new data sample shows that 25% more messages now have two (2) or more MIME parts at the top. Overall, the number and sizes of MIME attachments grew by almost three (3) times.

MIME Top Part Count	2003 Sample	2008 Sample
1	75.77%	50.46%
2	21.91%	46.20%

Table 1: MIME Structure Comparison

5.2.4 Multipurpose Internet Mail Extension (MIME) Profile

MIME is an internet attachment scheme, defined as a formal standard by RFCs 1521, 1522, and 1523. SPECmail2009 uses data collected in 2008 (provided by Apple) for probability distribution tables in both messages and message stores.

5.2.4.1 MIME Message Construction

The initial processing of all message sizes distinguished between single part sizes and multipart sizes. The SPECmail2009 benchmark prioritizes individual MIME part size over the global message size distribution.

MIME Part Count	Construction Rules
Single (1) Part	• Use “Content-type: text/plain” in message headers • Use subpart content size distribution
Multiple Parts	• Use “Content Type: multipart/mixed; boundary=”xxxxxxxxx-counter” in message headers • Use distributions for message part width and depth to help establish the set of multipart message bodies. • Categorize MIME messages to fall into one of these pre-defined multipart buckets. • Use subpart content size distribution to define the sub-part sizes in the fixed pool of pre-defined multipart messages.

The Top-Level Part Count distribution determines the probability that a message has N parts, where N is at least one (1). After computing the number of parts at any one level, the benchmark computes the probability that each message part contains further sub-levels, as defined by the MIME Part Depths distribution. Each message part size and MIME types uses the (configuration key) MIME_PART_SIZE and internal MIME Content Type Distributions to build the actual message content.

5.2.4.2 Benchmark MIME Message Distributions

The benchmark uses the MIME Parts, MIME Part Sizes and MIME Depth distribution tables to construct each message stored in the mail store. These configuration keys are fixed and cannot be changed for a compliant run.

Configuration Key	Definition and Value
MIME_TOP_PART_COUNT	Defines the probability that a message will have one (1) through five (5) MIME parts. Each is either a message text, zero or more attachments, or alternative views of the same message texts (example plain versus rich text versions of the same message): `MIME_TOP_PART_COUNT = "1,50.46%; 2,46.20%; 3,2.51%; 4,0.29%; 5,0.54%"`
MIME_PART_LEVEL	Defines the probability distribution of a message having up to five (5) nesting levels in the overall MIME structure: `MIME_PART_LEVEL = "1,90.18%; 2,9.14%; 3,0.62%; 4,0.04%; 5,0.02%"`
MIME_PART_SIZE	Defines the probability distribution of individual MIME part sizes: `MIME_PART_SIZE = "64,0.40%; 128,5.18%; 256,2.28%; 512,6.37%; 1024,9.22%; 2048,18.00%; 4096,28.97%; 8192,11.37%; 16384,6.46%; 32768,3.91%; 65536,3.02%; 131072,1.88%; 262144,1.21%; 524288,0.68%; 1048576,0.45%; 2097152,0.60%"`

5.3 Benchmark Management Definition

5.3.1 Benchmark Manager/Load Generators

The following configuration keys allows the benchmark manager to contact the actual load generators and distribute the workload definitions to each. The manager evenly divides the overall workload evenly among all load generators.

Configuration Key	Definition and Value
CLIENTS	List of hostname:port pairs representing one or more load generators
THREADS_PER_CLIENT	Maximum number of Java threads used by each load generator during initialization, verification and context gather phases. Default: `THREADS_PER_CLIENT = 1`
LOAD_FACTORS	Percent of computed load to use as part of the load test period. Default: `LOAD_FACTORS = 100`
RUN_SECONDS	How long to run and gather statistics at peak hour loads. Default: `RUN_SECONDS = 3600`
WARMUP_SECONDS	How long to run at peak hour loads before gathering statistics. Default: `RUN_SECONDS = 600`

Table 2: Benchmark and SUT Configuration Keys

5.3.2 Secure TCP Connection

During the five years between the first data sample and the 2008 sample, more stringent corporate security policies emerged. Many corporate electronic security policies require encrypted TCP connections between e-mail clients and server. The SPECmail2009 benchmark includes options to use secure TCP connections (SSL or TLS) instead of transmitting clear text sessions. Should this option be activated, then the metric name will reflect this fact.

Configuration Key	Definition and Value
IMAP_SECURE **SMTP_SECURE**	Determines if the benchmark uses secure TCP connections for IMAP or SMTP sessions. Default: `off`
IMAP_SECURE_INIT **IMAP_SECURE_VERIFY** **IMAP_SECURE_GATHER** **IMAP_SECURE_CLEAN**	These keys determine whether or not the benchmark should use a secure TCP connection during phases OTHER than the actual load test phase. The possible values are `none`, `STARTTLS` and `SSL`. Default: `none`
IMAP_SSL_PROTOCOL **IMAP_STARTTLS_PROTOCOL**	Security protocol to use with IMAPS and STARTTLS. Choices are: SSLv3, TLSv1 Default: `TLSv1`

Configuration Key	Definition and Value
`IMAP_SSL_CIPHER` **`IMAP_STARTTLS_CIPHER`**	Cipher suite to use with IMAPS and STARTTLS. Default: `SSL_RSA_WITH_RC4_128_MD5`
`SMTPS_PORT`	Sets the SUT's SMTPS port number under TLS mode. Default: `465`
`IMAPS_PORT`	Sets the SUT's IMAPS port number under TLS mode. Default: `993`

When the IMAP_SECURE key is _on,_ the benchmark uses encrypted TCP connections for all IMAP sessions. However by default, only the actual load test phase uses SSL/TLS. If the e-mail SUT requires all IMAP connections be encrypted, then set the IMAP_SECURE_INIT, IMAP_SECURE_VERIFY, IMAP_SECURE_GATHER, or IMAP_SECURE_CLEAN configuration keys to _SSL_ or _TLS_, instead of _off_.

When the SMTP_SECURE key is _on_, the benchmark uses encrypted TCP connections for all internal SMTP sessions. However, the benchmark uses unencrypted TCP connections for any messages that interact with a remote domain.

5.4 Transaction Workload

The Transaction Workload defines the number and type of transactions issued against the system under test during the measurement phase of the benchmark. It scales linearly with the number of users and with the **SPECmail_MSEnt2009** metric. Its parameters are:

Transaction	Description
User Mail Store	The overall folder (mailbox) quantity and depths distributions (see prior section)
SMTP	The number of new SMTP sessions established per second and the percent of SMTP traffic between local and remote domains
IMAP	Number of existing IMAP4 sessions present as well as the number of new IMAP4 sessions established per second
Message	The distribution and quantity of existing messages entering the peak hour, using the specified message MIME structure and part size distributions.

5.4.1 SPECmail2009 Enterprise Workload Profile

The definition of the transaction workload starts with an assessment of the per-user, per-day load profile. The following table shows assumptions for that profile, as well as the semantics for the elements in that profile.

The benchmarker should set these configuration keys to match the names, ports and values of the System Under Test.

Configuration Key	Definition and Value
`USERNAME_PREFIX`	First part of all test user names used as SMTP addresses and IMAP4 login names. Default: `USERNAME_PREFIX = spec`

Configuration Key	Definition and Value
USER_END	Maximum value appended to USERNAME_PREFIX and must be at least 200 larger than USER_START: Default: USER_END = 200
USER_START	Minimum value appended to USERNAME_PREFIX: Default: USER_END = 1
USERNAME_AS_PASSWORD	Use the generated account name as the login password. Default: USERNAME_AS_PASSWORD = 1 or 0
USER_PASSWORD	Use this value as the login password for all accounts. Example: USER_PASSWORD = <string>
IMAP_SERVER	Hostname (not IP) of SUT's IMAP4 server
IMAP_PORT	Socket number of SUT's IMAP4 server
SMTP_SERVER	Hostname (not IP) of SUT's SMTP server
SMTP_PORT	Socket number of SUT's SMTP server
LOCAL_DOMAIN	Domain name associated with local accounts
REMOTE_DOMAIN	Domain name used to relay message

Table 3: Benchmark and SUT Configuration Keys

5.4.2 SMTP Workload Profile

The following table summarizes the compliant SMTP workload used during the load test period.

Configuration Key	Definition and Value
PEAK_PCT_USERS	Percent of provisioned users who receive messages during the peak hour (also known as Active users). Default: PEAK_PCT_USERS = 78%
MSG_RECEIVED_PER_PEAK_HOUR	Number of messages received by Active users during peak hour: Default: MSG_RECEIVED_PER_PEAK_HOUR = 5.06
MSG_RECP_DISTRIBUTION	Distribution defining number of recipients per message. The average is 3.0.6 Default: MSG_RECP_DISTRIBUTION = 1,46.3875%; 2,11.00%; 3,9.00%; 4,8.00%; 5,7.00%; 6,6.00%; 7,5.00%; 8,4.00%; 10,2.00%; 11,1.00%; 12,0.30%; 13,0.10%; 14,0.05%; 15,0.05%; 16,0.05%; 30,0.05%; 50,0.01%; 100,0.0025%

Table 4: Peak Hour SMTP Normalized User Profile

5.4.3 IMAP Workload Profile

IMAP sessions are fairly complex in nature due to their long life times, the ability to manage multiple folder layers, and the ability to append, retrieve, and delete messages. In addition, IMAP e-mail clients used different paradigms to monitor and manage user message stores. The IMAP benchmarks needed a more complex load generation structure due to this combination of changing behavior, different session lengths, and even the number of concurrent sessions connected to the IMAP server.

The SPECmail2009 IMAP Workload is defined by the combination of two categories: *client-type* and *command sequences.*

Command Sequence A series of IMAP commands issued after the initial `LOGIN` command, and terminated either a `LOGOUT` command or an idle session timeout value. These differentiate into many functional sessions – depending on the e-mail client type. Some Command Sequences work together. Other Command Sequences work independently of each other. Some have very short durations, while others can span days or weeks.

Client Type A collection of one or more command sequences that characterizes the type of IMAP sessions different IMAP4 clients implement.

The following table describes the criteria for each command-sequence.

Command Sequence	General Characteristic	Comments
1	• Create connection • Perform several operations using a variety of commands (probe folder for new messages, deleting, and moving messages, updating flags, list available folders, appending messages, searching for messages, checkpointing, etc.) • Occasionally probe folders for new messages • Fetch headers if any messages arrived • Occasionally fetch body (whole or parts of body) • Focuses on a specific folder • Does not log out session	**Example Clients**: Netscape (Mozilla), Pine, Mulberry) This is one of the “primary” sessions that tend to stay logged into the IMAP server for many hours or days. Netscape uses UID commands, Pine and Mulberry do not. Probing folders is accomplished by: 1. Netscape: NOOP; UID FETCH n:* (FLAGS) 2. Mulberry: SEARCH UNSEEN; SEARCH DELETED; FETCH 1:m (FLAG ENVELOPE BODYSTRUCTURE, …) 3. Pine: NOOP
2	• Create connection • Perform several operations using a variety of commands (probe folder for new messages, deleting, and moving messages, updating flags, list available folders, appending messages, searching for messages, checkpointing, etc.) • Occasionally fetch headers • Occasionally fetch header and whole body • Does ***not*** focus on a specific folder • Does not log out of session	**Example Clients**: Outlook, Outlook Express, Mulberry This is one of the “primary” sessions that tend to stay logged into the IMAP server for many hours or days. Probing folders is accomplished by these IMAP commands: • UID FETCH n:* (UID, BODY.PEEK[HEADER], …) • UID FETCH 1:n-1 (UID FLAGS)
3	• Create connection • Fetch headers	**Example Clients**: Fetchmail, Outlook Express

Command Sequence	General Characteristic	Comments
	• Fetch whole body • Logout	These sessions are very sporadic and show dependency on results returned from Command Sequence 4.
4	• Create connection • Occasionally probe folders for new messages • Occasionally issue other IMAP commands that does not alter the state of the mailstore (such as UNSUBSCRIBE or LIST) • Sometimes logs out, not always	**Example Clients**: Outlook, Outlook Express, Netscape - periodic or triggered actions These sessions show very automated behavior and are generated at fixed intervals for each user. Probing folders is accomplished by: • Outlook 2002 – Inbox: UID FETCH m:* (UID, BODY.PEEK[HEADER], …); or UID FETCH 1:n (UID FLAGS) • Outlook 2002 – Others: LSUB "" "*"; or STATUS "mailbox name **1**" (UNSEEN); ..; STATUS "mailbox name **n**" (UNSEEN); • 2. Outlook Express: STATUS "mailbox name" (MESSAGES UNSEEN)
5	• Create connection • Occasionally list or probe folders • Perform specific tasks, such as deleting, messages, or appending messages, etc. • Alters the state of the mail store • Logout	**Example Clients**: Mulberry, Netscape These sessions tend to focus on a specific set of tasks and then log out of the IMAP server.

Table 5: IMAP Command Sequence Definition

IMAP4 clients tend to use one or more of the five (5) command sequences, connecting one or more times to the IMAP server. The IMAP4 benchmark emulates four (4) client types, with each client type session cluster representing a single user.

Client Type	Constituent	Comments
1	**CS1** **CS4**	These two (2) command sequences operate independently and concurrently. Some of these clients use a message index number while others use the message UID.
2	**CS1** **CS4** **CS5**	These three (3) command sequences operate independently and concurrently. Some of these clients use message index number while others use the message UID.
3	**CS2** **CS3** **CS4**	These three (3) command sequences depend on both user and automated actions. The CS2 primary session tend to govern the tasks done in CS4. The CS3 sessions are automatic and periodic. Its results influences the other two.

Client Type	Constituent	Comments
4	**CS2** **CS4** **CS5**	These three (3) command sequences operate independently and in parallel. The client uses message index number instead of message UID.

Table 6: IMAP Client Type Definitions

The compliant run uses the following combination to determine sequencing and dependencies.

Configuration Key	Definition and Value
`PEAK_LOAD_PERCENT`	Percent of the daily IMAP load occurring during the peak hour: Default: `PEAK_LOAD_PERCENT = 32`
`CLIENT_TYPE_DISTRIBUTION`	Defines the probability that a message will have one (1) through five (5) MIME parts. Each is either a message text, zero or more attachments, or alternative views of the same message texts (example plain versus rich text versions of the same message): Default: CLIENT_TYPE_DISTRIBUTION = "1,31.373%; 3,32.353%; 4,3.922%; 5,2.941%; 13,3.922%; 14,10.784%; 15,1.961%; 24,0.980%; 34,2.941%; 45,2.941%; 134,0.980%; 145,3.922%; 1245,0.980%"
`CS3_MEAN_IA`	Interarrival rate of new CS3 sessions: Default: `CS3_MEAN_IA = 274350`

Table 7: IMAP Client Type Load and Distributions

Each `CLIENT_TYPE_DISTRIBUTION` couplet defines the command sequence grouping. The left element is a list of Command Sequence numbers (1 == CS1, 34 == CS3+CS4). The right element is the percentage of overall load generator client threads that will implement each combination. The number of IMAP sessions varies as this matrix changes. Each load generator thread is assigned one specific combination.

The total number of IMAP sessions is related directly to the total number of active users, as distributed by CS1, CS2, CS4 and CS5 percentages. The effective Client Type distribution applied to the base UserCount results in at least 136% concurrent logins. The total does not include *CS3* because folder subscriptions and message sizes control their initiation.

CS1	CS2	CS3	CS4	CS5
53.92%	1.96%	40.20%	27.45%	12.75%

Table 8: Effective IMAP Client Type Distribution

5.5 SMTP Benchmark States

The SPECmail2009 benchmark adjusted the SMTP traffic profile based on the Apple data. However, the actual SMTP states did not change. Each SMTP protocol step is mapped directly to an internal state, and tracked accordingly.

5.5.1 SMTP Message Arrival Rates

The message inter-arrival time computation uses a simplified model because the total number of new messages tends to be insufficient to fulfill a complex distribution. Therefore, the time between message delivery attempts is computed as the total number of messages to be delivered over the duration of the load test run time, divided by that run time.

```
SMTP Inter-arrival Time = (Number of Active Users)
            X (Messages per User)
            X (Recipients per Message)
            / (Test Run Time (s))
```

5.5.2 SMTP Message Routes

The normalized distribution of SMTP traffic that flows between internal and external recipients is reflected in the following table.

Configuration Key	Definition and Value
LOCAL_TO_LOCAL_PCT	Percent of total messages sent from Local users to Local users Default: LOCAL_TO_LOCAL_PCT = 56
REMOTE_TO_LOCAL_PCT	Percent of total messages sent from Remote users to Local users Default: REMOTE_TO_LOCAL_PCT = 31
LOCAL_TO_REMOTE_PCT	Percent of total messages sent from Local users to Remote users Default: LOCAL_TO_REMOTE_PCT = 13

Table 9: Peak Hour SMTP Traffic Flows

5.6 IMAP Benchmark's States

The analysis process classified individual extracted IMAP sessions according to the rules in Table 5: IMAP Command Sequence Definition, and the actual IMAP4 command and parameter combination mapped to common states. The state-to-state transitions were then collated and the corresponding probabilities (represented as percentages) collected. The analysis process created a large number of states (234), and a wide variety of possible state transitions (from 1 to 24). However, further analysis reduced the large number of states to only 64 states, maximum. The restrictions include

- commands needed to establish one of the five Command Sequences
- commands present during the peak hour
- commands that represented a more than 5 percent of the total number

- commands between 1 and 5% that should incur a disproportional computing resource, such as SEARCH or moving messages between folders

Table 10 lists all derived IMAP command state names, their numeric state ID code, and which data source used it. Some states are variations of each other (same command but slight parameter variation) because of the four different IMAP client types. These different clients used these variants for the same purpose. The benchmark treats these as unique command states, based on Client Type affiliation.

Table 10: IMAP4 Command-States Used in Peak Hour

State ID	State Name
1	APPEND
2	CHECK
6	CREATE
13	EXPUNGE
15	FETCH_NUM_BODYALL
17	FETCH_NUM_BODYPEEK
18	FETCH_NUM_BODYPEEK_HEADER
20	FETCH_NUM_BODYSTRUCTURE_FLAGS
25	FETCH_NUM_ENVELOPE_BODYPEEK_HEADERFIELDS_FLAGS_INTERNALDATE_RFC822SIZE_UID
26	FETCH_NUM_FLAGS
31	FETCH_NUM_UID
32	FETCH_NUM_UID_BODYPEEK_HEADERFIELDS_ENVELOPE_FLAGS_INTERNALDATE_RFC822SIZE
33	FETCH_RANGE_UID
37	FETCH_RANGE_FLAGS_BODYSTRUCTURE_ENVELOPE_INTERNALDATE_RFC822SIZE_UID
38	FETCH_RANGE_UID_BODYPEEK_HEADERFIELDS_ENVELOPE_FLAGS_INTERNALDATE_RFC822SIZE
44	LIST
45	LOGIN
46	LOGOUT
47	LSUB_NULL_FOLDER
50	LSUB_NULL_WILDCARD
51	LSUB_WILDCARD_WILDCARD
52	NOOP
55	SEARCH_ALL_DELETED
59	SEARCH_DELETED
62	SEARCH_UNSEEN
64	SELECT_FOLDER
66	SELECT_INBOX
67	SELECT_INBOXSENT
94	STORE_NUM_SET_FLAGS_DELETED
105	UID_COPY_NUM_FOLDER
108	UID_COPY_RANGE_FOLDER
111	UID_COPY_SERIES_FOLDER
113	UID_FETCH_NUM_BODYALL
114	UID_FETCH_NUM_BODYPARTS
115	UID_FETCH_NUM_BODYPEEK
116	UID_FETCH_NUM_BODYPEEKALL
117	UID_FETCH_NUM_BODYPEEK_HEADER
119	UID_FETCH_NUM_BODYSTRUCTURE
120	UID_FETCH_NUM_BODY_BODYMIMEALL_BODYMIMEPARTS_HEADER
121	UID_FETCH_NUM_BODY_BODYMIMEALL_HEADER
125	UID_FETCH_NUM_RFC822SIZE

State ID	State Name
129	UID_FETCH_NUM_UID_BODYPEEK_HEADERFIELDS_FLAGS_RFC822 SIZE
131	UID_FETCH_NUM_UID_BODYPEEK_RFC822SIZE
132	UID_FETCH_NUM_UID_BODY_RFC822SIZE
134	UID_FETCH_RANGE_UID_BODYPEEK_HEADERFIELDS_FLAGS_RFC822SIZE
135	UID_FETCH_RANGE_UID_BODYPEEK_RFC822SIZE
142	UID_FETCH_SERIES_UID_BODYPEEK_RFC822SIZE
147	UID_FETCH_UNTILEND_FLAGS
148	UID_FETCH_UNTILEND_UID_BODYPEEK_HEADERFIELDS_FLAGS_RFC822SIZE
153	UID_SEARCH_DELETED
157	UID_SEARCH_HEADER_UNDELETED
160	UID_SEARCH_RFCHEADER_UNDELETED
171	UID_SEARCH_UNSEEN
172	UID_SEARCH_UNTILEND_
173	UID_STORE_NUM_SET_FLAGS_AN SWERED
176	UID_STORE_NUM_SET_FLAGS_DELETED
178	UID_STORE_NUM_SET_FLAGS_FLAGGED
179	UID_STORE_NUM_SET_FLAGS_SEEN
181	UID_STORE_NUM_SET_FLAGS_SEEN_DELETED
183	UID_STORE_NUM_UNSET_FLAGS_ANSWERED
184	UID_STORE_NUM_UNSET_FLAGS_DELETED
185	UID_STORE_NUM_UNSET_FLAGS_FLAGGED
188	UID_STORE_NUM_UNSET_FLAGS_SEEN
199	UID_STORE_RANGE_SET_FLAGS_ANSWERED
200	UID_STORE_RANGE_SET_FLAGS_DELETED
209	UID_STORE_SERIES_SET_FLAGS_DELETED
215	SEARCH_ALL_CALL_INFORMATION
216	UID_COPY_NUM_
217	UID_COPY_NUM_TRASH
218	UID_COPY_RANGE_TRASH
219	UID_COPY_SERIES_TRASH
220	UID_FETCH_NUM_BODYPEEK_RFC822SIZE_UID
221	UID_FETCH_NUM_BODY_RFC822SIZE_UID
222	UID_FETCH_NUM_UID_BODYPEEK_HEADER_FLAGS_RFC822SIZE
223	UID_FETCH_RANGE_UID_BODYPEEK_HEADER_FLAGS_RFC822SIZE
224	LSUB_
225	LSUB_.
226	SUBSCRIBE_TRASH
227	UID_COPY_NUM_.
228	UID_COPY_RANGE_.
229	UID_COPY_SERIES_.
230	UID_FETCH_NUM_BODYMIMEALL
231	UID_FETCH_NUM_UID_BODYSTRUCTURE
232	UID_FETCH_RANGE_BODYPEEK_HEADERFIELDS
233	UID_FETCH_UNTILEND_FLAGS_RFC822SIZE
234	SESSION_START

5.6.1 SPECmail2009 State Engine Components

The benchmark's state engine is driven by a set of internally defined tables, found in the CS1State.java, CS2State.java, CS3State.java, CS4State.java, and CS5State.java files.

Benchmark Global Table	Definition and Value

TOSTATE[][]	An array of State Identifiers. Each row in the array corresponds to the integer State ID. Each column represents one of the possible "next" states.
	
	Example:
	
	`CS1STATE::TOSTATE[][] = {`
	`{ 0 }, // 0 Place holder`
	`{ 1, 23, 25, 27, 32, 33, 56 }, // 1`
	`{ 12, 23, 27, 54, 60 }, // 2`
	`...`
	`{ 12, 14, 15, 25, 27, 48, 51, 53, 54 }, // 5`
	`{ 1, 2, 6, 7, 9, 11, 15, 27, 36, 37 }, // 6`
	`{ 6, 7, 11, 37 }, // 7`
	`...`
	`{ 2, 25, 27, 40, 50, 56, 57, 59, 61 }, // 59`
	`{ 27, 40, 47, 50, 54, 57 }, // 60`
	`{ 50, 59, 61 }, // 61`
	`{ 5, 27, 40, 47, 50, 56 }, // 62`
	`{ 27 }, // 63`
	`{ 5, 42, 47, 50 } // 64`

TOSTATEPERCENT[][]	Defines the probability (column) that each state present in CS2 (row) moves to the next state, as defined in the corresponding TOSTATE array.
	
	Example:
	`CS2STATE::TOSTATEPERCENT[][] = {`
	`{ 0.0000 }, // 0 Place holder`
	`{ 0.2000, 0.2000, 0.2000, 0.4000 }, // 1`
	`{ 0.0465, 0.6744, 0.1628, 0.0698 }, // 2`
	`{ 1.0000 }, // 3`
	`{ 1.0000 }, // 4`
	`{ 0.7500, 0.2500 }, // 5`
	`{ 1.0000 }, // 6`
	`{ 0.0132, 0.0132, 0.1711, 0.8026 }, // 7`
	`{ 0.0030, 0.0030, 0.0030, 0.1875, 0.8036 }, // 8`
	`{ 1.0000 } // 9`

toStateCount[][] **toStateAllowed[][]**	These two pre-sized tables to hold occurrence counts and transition permissions. These two table ensure that every state (column) transition has been invoked, in the proper proportion. This enforces compliance with the TOSTATEPERCENT table, in case the "random" function is not very random.
TOSTATEIARATE[][]	This array defines the State-to-State Inter-Arrival wait time computation type (LOGNORM, in most cases) and the derived from the IMAP session data samples. As with the TOSTATE and TOSTATEPERCENT, each row represents a specific IMAP4 command and parameter combination. Each column represents the minimum wait time before moving to the corresponding "next" state.

```
Example:
CS4STATE:: TOSTATEIARATE[][]= {
{ "Place holder" }, // 0 Place holder
{ "LOGNORM:349.6307", "LOGNORM: 3386.3247",
"LOGNORM:395.4172", ...  }, // 1
{ "LOGNORM:0.0998" }, // 2
{ "LOGNORM:0.0718" }, // 3
{ "LOGNORM:0.0988" }, // 4
{ "LOGNORM:0.1369", "LOGNORM
{ "LOGNORM:459.2843", "LOGNO
{ "LOGNORM:0.0546", "LOGNORM
{ "LOGNORM:0.0000" }, // 8
{ "LOGNORM:0.0032", "LOGNORM:0027", "LOGNORM:0.0079" },
{ "LOGNORM:428.7473", "LOGNO
{ "LOGNORM:1.4748" }, // 11
{ "LOGNORM:1.1354" }, // 12
{ "LOGNORM:0.0000" }, // 13
{ "LOGNORM:0.0153" }, // 14
{ "LOGNORM:0.1084" }, // 15
{ "LOGNORM:0.0932" }, // 16
...
```

STATEIDTOMETACOMMAND[] The actual State Identification strings to be used for internal to external reports, such as debug statements and results.

```
Example:
CS4STATE::STATEIDTOMETACOMMAND[]={
"Place holer", // 0 Place holder
"APPEND", //1
"CHECK", //2
"CLOSE", //3
"CREATE", //4
"EXPUNGE", //5
"LIST", //6
"LOGIN", //7
"LOGOUT", //8
"NOOP", //9
"SELECT_FOLDER", //10
"SELECT_INBOX", //11
"SELECT_INBOXSENT", //12
"SESSION_START", //13
"SUBSCRIBE_FOLDER", //14
"UID_COPY_NUM_FOLDER", //15
"UID_COPY_RANGE_FOLDER", //16
"UID_FETCH_NUM_BODYALL", //17
...
```

Table 11: IMAP State Transition Tables and Probabilities

6. Benchmark Reportable Parameters:

The following table shows IMAP4 states deemed critical enough to consider the corresponding Quality-of-Service (QoS) value. If any of these states fails the QoS test then the results are deemed *non-compliant*. Non-critical states that fail the QoS criteria will not affect the compliance logic.

Table 12: Required QoS Compliance States

Protocol	State Identification String
SMTP	TCP Connect
	DATA
IMAP	TCP Connect
	APPEND
	EXPUNGE
	FETCH_NUM_BODYALL
	FETCH_RANGE_ENVELOPE_BODYPEEK_HEADERFIELDS_FLAGS_INTERNALDATE_FC822SIZE_UID
	LIST
	LSUB_NULL_WILDCARD
	SELECT_FOLDER
	SELECT_INBOX
	STORE_NUM_SET_FLAGS_DELETED
	UID_STORE_NUM_SET_FLAGS_DELETED
	UID_STORE_NUM_SET_FLAGS_SEEN
	UID_FETCH_RANGE_UID_BODYPEEK_HEADERFIELDS_FLAGS_FRC822SIZE
Overall	Error Rate

7. References:

7.1 Relevant RFCs (see www.ietf.org):

2045 – Part 1: Format of Internet Message Bodies
2046 – Part 2: Media types
2047 – Part 3: Header and Body Extensions for non-ASCII Text, non Textual message parts and multi-part messages
2048 – Part 4: Registrations
2049 – Part 5: Conformance Criteria and Examples
2231 – Extension to specify the language to display the part, parameter values in other (non US-ASCII) character sets, and continuation mechanism for long parameter values.
2646 – Update to RFC 2046 to define variations of supported “Plain/Text” content types to incorporate legacy plain text and flow line control.

7.2 Up-to-date Benchmark Documents

The most up-to-date version of all SPECmail2009 documents can be found on the website: http://www.spec.org/mail2009