EMSD Efficiency

Neda Document Number: 103-101-01.01

Last Updated: Author unspecified

Doc. Revision: source unspecified

http://www.emsd.org/

October 23, 1996

Contents

List of Figures

List of Tables

Abstract

Efficient Mail Submission & Delivery (EMSD) [?] is designed to provide SMTP-over-TCP functionality with reduced overhead via EMSD-over-UDP. Reliability is achieved using the Efficient Short Remote Operations Services 3-way handshake protocol. In this study, we compare the overhead incurred by using EMSD as compared to other Email protocols, and demonstrate EMSD's efficiency advantage. We use a Sniffer to capture the actual packets sent and received from a User Agent (UA), and count the number of bytes in the IP layer and above. These provide a reasonable figure when the UA is sending/receiving messages over an airlink.

Preface

This article was originally published in October 1996. It is being included in the Manifesto, essentially unchanged from its original form.

1 Introduction

We provide here an overview of both SMTP and EMSD, to compare and contrast their features and to lay the groundwork for analysis of the experimental results in Sections **??** and **??**.

1.0.1 SMTP

According to RFC 821[?], the objective of Simple Mail Transfer Protocol (SMTP) is to transfer mail reliably and efficiently. The SMTP design is based on the following model of communication:

As the result of a user mail request, the sender-SMTP establishes a two-way transmission channel to a receiver-SMTP, which may be either the ultimate destination or an intermediate.

Following this, the sender-SMTP sends a MAIL command indicating the sender of the mail. If the receiver-SMTP can accept mail it responds with an OK reply. The sender-SMTP then sends a RCPT command identifying a recipient of the mail. If the receiver-SMTP can accept mail for that recipient it responds with an OK reply; if not, it responds with a reply rejecting that recipient (but not the whole mail transaction). The sender-SMTP and receiver-SMTP may negotiate several recipients. When the recipients have been negotiated, the sender-SMTP sends the mail data, terminating with a special sequence. If the receiver-SMTP successfully processes the mail data it responds with an OK reply. Note that the dialog is purposely lock-step, one-at-a-time.

SMTP provides two mechanisms for the transmission of mail: directly from the sending user's host to the receiving user's host when the two host are connected to the same transport service, or via one or more relay SMTP-servers when the source and destination hosts are not connected to the same transport service. The mail commands and replies have a rigid syntax. Replies also have a numeric code.

Thus it can be seen that for the exchange of any one message with SMTP, a number of transactions must be completed. EMSD attempts to improve efficiency by cutting down on this number and simplifying the process for the case of short messages.

1.1 Efficient Mail Submission & Delivery

The EMSD specifications define the protocols between an EMSD Device and an EMSD Server. EMSD requires ESROS (Efficient Short - Remote Operation Services) [?]. The EMSD-P&FS consist of two independent components: Efficient Mail Submission & Delivery Protocol (EMSD-P) and EMSD Format Standards (EMSD-FS).

EMSD-FS is responsible for defining the format of a limited size interpersonal message. It defines the ”Content” encoding (Header + Body) and the end-to-end envelope. It relies on EMSD-P for the transfer of the content to its recipients.

EMSD-P is responsible for wrapping a limited size message in a point-to-point envelope and submitting or delivering it. EMSD-P performs the envelope encoding and relies on the services of ESROS for transporting the envelope. Some of the services of EMSD-P include message originator authentication and optional message segmentation and re-assembly. The Efficient Mail Submission & Delivery Protocols are designed with three high level goals:

- Define the new ”world” of Efficient Mail Submission & Delivery
- Define a remote operations service that can handle messaging and other standard networking applications
- Make Efficient Mail Submission & Delivery an extension of the existing internetworking world

These goals will prevent, whenever possible, the expense and associated problems of ”re-inventing the wheel.” The EMSD Protocols make heavy use of existing technology:

- RFC-822
- ASN.1
- Basic Encoding Rules
- X.400 and Internet e-mail

These technologies have been thoroughly tested and have proven to be reliable solutions for the problems they address (e.g. message format, reliable message delivery, encoding and compacting). The EMSD Specifications allow for users who enjoy the advantages of this new technology and at the same time want be connected to the rest of the existing messaging world.

2 Study Overview

We have chosen to compare the efficiency of using EMSD to the efficiency obtained by other submission and delivery protocols in this study. While it is debatable whether EMSD can be placed at the same level as the test protocols, we nonetheless feel that a study such as this is quite useful and provides a common denominator to discuss various aspects of EMSD performance.

The experiments cover both submission (from a mobile unit) and delivery (to a mobile unit). Under submission we looked at comparing EMSD and SMTP. The delivery experiments tested EMSD vs. SMTP, POP, and IMAP. In all cases a single uniform test message was relayed between two devices (a recipient or sender, and a mail server) and the data traffic recorded. Although you cannot compare EMSD directly to any one messaging protocol, because each protocol is designed to perform a specific function, you can compare the results obtained by each messaging solution. The following table illustrates the functions supported by each protocol. Note that EMSD is the most like SMTP.

3 Submission

Please refer to Figure 1 below, which shows the setup for the following two submission experiments in detail. The experimental setup involves:

At Site One:

Protocol	Submission	Delivery	Relay	Retrieval	Mailbox Access	Mailbox Sync
SMTP	X	X	X	X		
IMAP				X	X	X
POP				X		
EMSD	X	X		X		

Table 1: Messaging Protocols

- A ”sender”: Toshiba Laptop running Windows 3.1 and Chameleon Winsock TCP/IP stack from NetManage
- A ”receiver”: Sun Sparc running Solaris 2.4
- A ”mail server”: Sun Sparc running Solaris 2.4
- A sniffer that monitors packet movement at the juncture of the above three devices, recording two-way traffic

At Site Two:

- A Message Test Center: Sun Sparc running Solaris 2.4.

The two setups are linked to each other over a number of routers across the Internet.

In both cases below, we are interested exclusively in analyzing the recorded data between the sender laptop and the Unix mail server (in the case of SMTP submission), or the EMSD Message Test Center (in the case of EMSD submission). Thus the ”receiver” shown below, although necessary to submit the message, does not enter into our study picture directly.

3.1 SMTP Submission from PC to Unix

3.1.1 Message Submission Process

Message was submitted from the Laptop to the Unix mail server. To submit a message from the laptop, Netscape’s Mail User Agent on Windows 3.1 was utilized. From the file menu on Netscape, ”New Mail Message” was selected, popping up a mail window. The message was typed in, a recipient (the ”receiver” in Figure 1) was specified, and ”Send” was then clicked. The sniffer recorded the exchange of data between the sender and the mail server that was ¡nwestmail.nwest.airdata.com¿ implementing sendmail.

3.1.2 Protocol Trace

The following is the protocol trace recorded by the sniffer. After TCP synchronization and acknowledgment, a virtual circuit is established between the sender’s Netscape Mail User Agent and sendmail on the mail server, and data is exchanged after specifying the sender and recipient addresses.

```
IP_PDU      MailServer             UA   DATA   TCP     IP
--------------------------------------------------------------------
1  TCP   .<------- TCP SYN -------- .     0     24     44
2  TCP   . ------- TCP SYN ack ---->.     0     24     44
```

```
3  TCP   .<------- Push ACK ------- .     0     20     40  (128)
4  SMTP  . ----220 server ready --->.   116    136    156
5  TCP   .<------- Push ACK ------- .     0     20     40  (196)
6  SMTP  .<------- HELO <client>--- .    36     56     76
7  SMTP  . ----250 server Hello --->.   111    131    151
8  TCP   .<------- Push ACK ------- .     0     20     40  (267)

9  SMTP  .<--MAIL FROM:<sender> --- .    32     52     72
10 SMTP  . ----250 ... Sender ok--->.    39     59     79
11 TCP   .<------- Push ACK ------- .     0     20     40  (191)
12 SMTP  .<--RCPT TO:<rcpt>-------- .    33     53     73
13 SMTP  .<----250...Recipient ok-- .    45     65     85
14 TCP   .<------- Push ACK ------- .     0     20     40  (198)

15 SMTP  .<------ "DATA" ---------- .     6     26     46
16 TCP   . ------- ACK ------------>.     0     20     40   (86)
17 SMTP  . ----354..end with "."--->.    50     70     90
18 TCP   .<------- Push ACK ------- .     0     20     40  (130)
19 SMTP  .<--Mail header+body ----- .   437    457    477
20 SMTP  .<------ . --------------- .     5     25     45
21 TCP   . ------- ACK ------------>.     0     20     40  (562)

22 SMTP  . ------- 250 Ok --------->.     8     28     48
23 TCP   .<------- Push ACK ------- .     0     20     40
24 TCP   .<------- Push Reset ----- .     0     20     40  (128)
----------------------------------------------------------------------
```

3.1.3 Measurement Results

```
Total IP Packet bytes: 1886
Message Length (header + body): 437
Total Overhead (TCP header + IP header): 1449
3.1.4 Message as Received
Message-ID: <32249501.46FD@airdata.com>
Date: Wed, 28 Aug 1996 11:50:41 -0700
From: Jia-bing Cheng <jcheng@airdata.com>
Organization: AT&T Wireless Services
X-Mailer: Mozilla 2.0 (Win16; U)
MIME-Version: 1.0
To: j.cheng@pocketnet.net
Subject: test3
X-URL: file:///c:/netscape/jbc.htm
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit

123456789012345678901234567890
```

```
12345678901234567890
1234567890
```

3.2 EMSD Submission from PC to Unix

3.2.1 Message Submission Process

The message was submitted from the laptop using Neda's EMSD Mail User Agent version 0.9 on Windows 3.1, to Neda's EMSD Message Test Center. EMSD-Pine version 3.91 was used to submit the message from the laptop. After invoking Pine, and typing "C", a new message was composed and then sent via ¡CTRL X¿. A direct connection was then established between the EMSD Mail User Agent on the laptop and the EMSD Message Test Center, and the message was relayed. The sniffer recorded exchange of data between the sender and Neda's EMSD Message Test Center which was ¡emsd.neda.com¿ implementing ESROS.

3.2.2 Protocol Trace

The following is the protocol trace recorded by the sniffer. As compared to the SMTP protocol trace in section 3.1.2, you can see the exchange is quite brief.

```
IP_PDU  MailServer            UA   DATA   UDP     IP
----------------------------------------------------------------
1  UDP   .<--Invoke header+body --- .   206    214    234
2  UDP   . -----Response ---------->.    15     23     43
3  UDP   .<------- Ack ------------ .     2     10     30
----------------------------------------------------------------
```

3.2.3 Measurement Results

Total IP Packet bytes: 307

Message Length (header + body): 206

Total Overhead (EMSD header + UDP header + IP header): 101

Total IP bytes in the case of EMSD submission as compared to SMTP submission is down by a factor of 6; the header count is down by a factor of 2.6; and total overhead is down by a factor of 14, representing major savings in data traffic.

3.2.4 Message as Received

The # text below is provided as comments and does not appear in the received message.

```
P.!.0. ... 0..0z@."333. 333"<test1@emsd. neda.com>          # FROM:
010/@)Jia-bing-pn Cheng<j.cheng@pocketnet.net>             # RCPT:
......test4                                                # Subject:
....text/plain; charset=us-ascii                           # content-type
0C.A                                                       #

123456789012345678901234567890.                            # BODY:
12345679801234567890.
1234567890
```

4 Delivery

Similar to the submission experiments above, we also conducted analogous delivery tests. The first experiment on SMTP delivery is essentially the complement of the SMTP submission experiment described above, and uses the same setup as in Figure 1. The second and third delivery experiments are with POP and IMAP servers and are described in their corresponding sections below. The final experiment is on EMSD delivery and also uses the same setup as in Figure 1. We then compare the performance of EMSD delivery versus the other three delivery methods.

4.1 SMTP Delivery from Unix to Unix

Please refer to Figure 1 above for this experiment.

4.1.1 Message Delivery Process

The message was delivered to the Unix receiver from the Unix mail server. Both were implementing sendmail and the message was delivered via standard SMTP. The sniffer recorded the exchange of data between the recipient and the mail server, which was ¡nwestmail.nwest.airdata.com¿.

4.1.2 Protocol Trace

The following is the protocol trace recorded by the sniffer. After TCP synchronization and acknowledgment, a virtual circuit is established between the recipient's Mail User Agent and sendmail on the mail server, and data is exchanged after specifying the sender and recipient addresses.

```
IP_PDU  MailServer           bluejeans   DATA   TCP     IP
--------------------------------------------------------------------
1  TCP   .<------- TCP SYN -------- .     0     20     40
2  TCP   . ------- TCP SYN ack ---->.     0     20     40
3  TCP   .<------- Push ACK ------- .     0     20     40

4  SMTP  . ----220 server ready --->.   116    136    156
5  SMTP  .<------- HELO <client>--- .    16     36     56
6  SMTP  . ----250 server Hello --->.    95    115    135
7  SMTP  .<--MAIL FROM:<sender> --- .    29     49     69
8  SMTP  . ----250 ... Sender ok--->.    39     59     79
9  SMTP  .<--RCPT TO:<rcpt>-------- .    44     64     84
10 SMTP  .<----250...Recipient ok-- .    57     77     97

11 SMTP  .<------ "DATA" ---------- .     6     26     46
12 TCP   . ------- ACK ------------>.     0     20     40
13 SMTP  . ----354..end with "."--->.    50     70     90
14 SMTP  .<--Mail header+body ----- .   301    321    341
15 TCP   . -------- ACK ----------->.     0     20     40
16 SMTP  .<------ . --------------- .     5     25     45
17 TCP   . ------- ACK ------------>.     0     20     40
18 SMTP  . ------- 250 Ok --------->.     8     28     48
```

```
19 SMTP  .<------- QUIT ----------- .     6     26     46
20 SMTP  .--221 closing connection->.    46     66     86
21 TCP   .<------- FIN ACK -------- .     0     20     40
22 TCP   . -------- ACK ----------->.     0     20     40
23 TCP   . ------- FIN ACK -------->.     0     20     40
24 TCP   .<-------  ACK ----------- .     0     20     40
```

4.1.3 Measurement Results

Total IP Packet bytes: 1778
  Message Length (header + body): 301
  Total Overhead (TCP header + IP header): 1477

4.1.4 Message as Received

```
Received: by bluejeans. (SMI-8.6/SMI-SVR4)
<09>id PAA24890; Fri, 13 Sep 1996 15:34:53 -0700
Date: Fri, 13 Sep 1996 15:34:53 -0700
From: jcheng@bluejeans
Message-Id: <199609132234.PAA24890@bluejeans.>
To: dnakano@griffey.nwest.airdata.com
Subject: test1

1234567890 1234567890 1234567890
1234567890 1234567890
1234567890
```

4.2 Message Delivery via POP Mailbox

Please refer to Figure 2 below, which shows the setup for the following two delivery experiments in detail. The experimental setup at Neda Communications involves the following:

- A POP Server: Sun Sparc running Solaris 2.4.
- An IMAP Server: Sun Sparc running Solaris 2.4.
- A ”receiver”: IBM Thinkpad Laptop running Microsoft TCP/IP on Windows 95
- A sniffer that monitors packet movement at the juncture of the above three devices, recording two-way traffic

4.2.1 Message Delivery Process

The message was delivered to the laptop from the POP server. After invoking Microsoft’s Internet Explorer 3.0 on the laptop and bringing up MS Internet Mail, the new message was automatically retrieved from the POP server. The sniffer recorded traffic data between the POP server and the recipient.

4.2.2 Protocol Trace

```
        (arash)                (vader)
IP_PDU  Mailbox                 Client DATA   TCP     IP
---------------------------------------------------------------
1  DNS   *<-- DNS Query ----------- .                     (dns)
2  DNS   * -- DNS Reponse---------->.

3  TCP   .<-- SYN req-------------- .     0    24     44 (conn)
4  TCP   . ---SYN ACK ------------->.     0    24     44
5  TCP   .<-- ACK ----------------- .     0    20     40

6  TCP   . ---POP3 server OK ------>.   117   137    157 (auth)
7  TCP   .<-- ACK ----------------- .     0    20     40
8  TCP   .<-- AUTH twinkie -------- .    14    34     54
9  TCP   . ---unknown command ----->.    45    65     85
10 TCP   .<-- USER test-1 --------- .    13    33     53
11 TCP   . ---user acpt,password? ->.    41    61     81
12 TCP   .<-- PASS ****** --------- .    13    33     53
13 TCP   . ---+OK ----------------->.     0    20     40

14 TCP   . ---+OK mbox open 1 msg-->.    30    50     70 (trans)
15 TCP   .<-- STAT ---------------- .     6    26     46
16 TCP   . ---+OK 1 542------------>.    11    31     51
17 TCP   .<-- UIDL 1 -------------- .     8    28     48
18 TCP   . ---unknown command ----->.    43    63     83
19 TCP   .<-- TOP 1 0 ------------- .     9    29     49
20 TCP   . ---+OK Top of msg ------>.   503   523    543 (_header)
21 TCP   .<-- LIST ---------------- .     6    26     46
22 TCP   . ---+OK scan listing----->.    44    64     84
23 TCP   .<-- RETR 1 -------------- .     8    28     48
24 TCP   . ---+OK 542  msg body---->.   561   581    601 (_body)
25 TCP   .<-- DELE 1 -------------- .     8    28     48
26 TCP   . ---+OK msg deleted ----->.    21    41     61
27 TCP   .<-- ACK ----------------- .     0    20     40

28 TCP   .<-- QUIT ---------------- .     6    26     46
29 TCP   . ---+OK ----------------->.     6    26     46
30 TCP   . ---Sayonara ------------>.    14    34     54
31 TCP   .<-- FIN ACK ------------- .     0    20     40
32 TCP   . ---+OK sa--------------->.     6    26     46
33 TCP   .<-- FIN ACK ------------- .     0    20     40
34 TCP   . ---ACK ----------------->.     0    20     40
---------------------------------------------------------------
```

4.2.3 Measurement Results

Total IP Packet bytes: 2731
   Message Length (header+body): 561
   Total Overhead: 2170

4.2.4 Message as Received

```
+OK 542 octets..
Return-Path: <muratd@neda.com>..
Received: from vader.neda.com by arash.neda.com (5.0/SMI-SVR4)...
  id AA04601; Wed, 18 Sep 1996 16:35:39 +0800..
Date: Wed, 18 Sep 1996 16:35:11 -0700 ()..
From: Murat Divringi <muratd@neda.com>..
To: test-1@arash.neda.com..
Subject: test6..
Message-Id: <Pine.WNT.3.95.960918163418.-158025A-100000@vader.neda.com>..
X-X-Sender: muratd@zahak.neda.com..
Mime-Version: 1.0..
Content-Type: TEXT/PLAIN; charset=US-ASCII..
Content-Length: 66..
Status:   ..
..
0123456789012345678901234567890 ..
01234567890123456789 ..
0123456789 ..
..
```

4.3 Message Delivery via IMAP Mailbox

Please refer to Figure 2 above for the experimental setup.

4.3.1 Message Delivery Process

Message was delivered to the laptop from the IMAP server. After invoking PC-Pine, the new message was automatically retrieved from the IMAP server. The sniffer recorded traffic data between the IMAP server and the recipient.

4.3.2 Protocol Trace

```
        (zahak)         (198.62.92.35)
IP_PDU  Mailbox                 Client DATA   TCP     IP
---------------------------------------------------------------
1  DNS   *<-- DNS Query ----------- .                     (dns)
2  DNS   * -- DNS Reponse---------->.

3  TCP   .<-- SYN req-------------- .     0    24     44  (conn)
4  TCP   . ---SYN ACK ------------->.     0    24     44
```

```
5  TCP   .<-- ACK ----------------- .     0    20     40

6  TCP   . ---IMAP2 server OK ----->.    78    98    118 (auth)
7  TCP   .<-- ACK ----------------- .     0    20     40
8  TCP   .<-- LOGIN test-1 **** --- .    28    48     68
9  TCP   . ---ACK ----------------->.     0    20     40
10 TCP   . ---LOGIN completed ----->.    27    47     67

11 TCP   .<-- A001 SELECT INBOX --- .    21    41     61
12 TCP   . ---ACK ----------------->.     0    20     40
13 TCP   . ---A001 cmp 1 EXISTS --->.   110   130    150
14 TCP   .<-- A002 NOOP ----------- .    13    33     53
15 TCP   . -- A002 NOOP cmp ------->.    26    46     66
16 TCP   .<-- A003 FETCH 1:1 ALL -- .    22    42     62
17 TCP   . -- A003 FETCH evlp cmp-->.   364   384    404 (())
18 TCP   .<-- A004 NOOP ----------- .    13    33     53
19 TCP   . -- A004 NOOP cmp ------->.    26    46     66
20 TCP   .<-- ACK ----------------- .     0    20     40
21 TCP   .<-- A005 FETCH 1:1 FULL-- .    23    43     63
22 TCP   . -- A005 FETCH 1:1 cmp--->.   431   451    471 (())
23 TCP   .<-- A006 FETCH 1 RFC822hdr.    30    50     70
24 TCP   . -- A006 FETCH 1 cmp hdr->.   708   728    748 (_header)
25 TCP   .<-- A007 FETCH 1 body-----.    24    44     64
26 TCP   . -- A007 FETCH 1 cmp body>.   125   145    165 (_body)
27 TCP   .<-- ACK ----------------- .     0    20     40
28 TCP   .<-- A008 SEARCH DELETED --.    23    43     63
29 TCP   . -- A008 SEARCH cmp ----->.    38    58     78

30 TCP   .<-- A009 LOGOUT --------- .    15    35     55
31 TCP   . ---ACK ----------------->.     0    20     40
32 TCP   . -- A009 LOGOUT cmp ----->.    80   100    120
33 TCP   .<-- FIN ACK ------------- .     0    20     40
34 TCP   . ---ACK ----------------->.     0    20     40
35 TCP   . -- FIN ACK ------------->.     0    20     40
36 TCP   .<---ACK ----------------- .     0    20     40
```

4.3.3 Measurement Results

Total IP Packet bytes: 3593
  Message Length (header+body): 833
  Total Overhead: 2760

4.3.4 Message as Received

```
* 1 FETCH(RFC822.HEADER {646}..
Received: from arash.neda.com (arash [198.62.92.10]) by zahak.neda.com
```

```
  (8.6.10/8.6.10) with SMTP id QAA16710 for <test-1@zahak>;
  Wed, 18 Sep 1996 16:42:24-0700..
Received: from vader.neda.com by arash.neda.com (5.0/SMI-SVR4)...
  id AA04617; Wed, 18 Sep1996 16:41:42 +0800..
Message-Id: <9609182341.AA04617@arash.neda.com>..
From: "test-1" <test-1@neda.com>..
To: <test-1@zahak.neda.com>..
Subject: test6..
Date: Wed,18 Sep 1996 16:41:13 -0700..
X-Msmail-Priority: Normal..
X-Priority: 3..
X-Mailer: Microsoft Internet Mail 4.70.1155..
Mime-Version: 1.0..
Content-Transfer-Encoding: 7bit..
Content-Type: text/plain; charset=ISO-8859-1..
Content-Length: 66....)..
A00006 OK FETCH completed..
* 1 FETCH (BODY[1] {70}..
012345678901234567890123456789 ..
01234567890123456789 ..
0123456789..
..
)..
A00007 OK FETCH completed..
```

4.4 EMSD Delivery from Unix to PC

Please refer to Figure 2 above for this experiment.

4.4.1 Message Delivery Process

The message was delivered to the laptop, running Neda's EMSD Mail User Agent version 0.9 on Windows 3.1, from Neda's EMSD Message Test Center. The sniffer recorded exchange of data between the recipient and Neda's EMSD Message Test Center which was ¡emsd.neda.com¿ implementing ESROS.

4.4.2 Protocol Trace

```
IP_PDU   UA               MailServer   DATA   UDP     IP
---------------------------------------------------------------
1  UDP   .<--Invoke header+body --- .   299    307    327
2  UDP   . -----Response ---------->.     2     10     30
3  UDP   .<------- Ack ------------ .     2     10     30
---------------------------------------------------------------
```

4.4.3 Measurement Results

Total IP Packet bytes: 387

	EMSD	SMTP
Total number of IP packets	3	24
Total IP bytes	307	1886
Total MSG length (mail hdr+ mail body)	206	437
Total overhead	101	1449

Table 2: Comparison of Submission Traffic overhead for EMSD and SMTP

Message Length (header+body): 299

Total Overhead: 88

Comparing EMSD delivery with SMTP delivery we see that total IP packets in the case of EMSD delivery is down by a factor of 4.6, and total overhead is down by a factor of 16.8.

In the case of POP retrieval, total IP bytes in the case of EMSD delivery is down by a factor of 7, and total overhead is down by a factor of 24.7.

Finally for IMAP delivery, total IP packets in the case of EMSD delivery is down by a factor of 9.3, and total overhead is down by a factor of 31.4.

4.4.4 Message as Received and Decoded

```
From jcheng@airdata.com Tue Oct 01 16:16:31 1996
Date: 14 Sep 96 05:48:55 GMT
From: jcheng@airdata.com
Subject: TEST Subject
To: 333.333@
Message-ID: <199609132148.OAA24774@bluejeans.>
Content-Length: 66
X-Homepage: Visit our home page at http://www.airdata.com/
     id OAA24774; Fri, 13 Sep 1996 14:48:55 -0700

1234567890 1234567890 1234567890
1234567890 1234567890
1234567890
```

5 Results Summary

The following paragraphs summarize the results obtained above. Results indicate that EMSD compares very favorably to other message transfer mechanisms.

6 Conclusion

Results of the experiments show the dramatic efficiency gain of EMSD over all the other protocols under test.

	EMSD	SMTP	IMAP	POP
Total number of IP packets	3	24	36	34
Total IP bytes	387	1778	3593	2731
Total MSG length (mail hdr+ mail body)	299	301	833	561
Total overhead	88	1477	2760	2170

Table 3: Comparison of Delivery Traffic Overhead for EMSD, SMTP, IMAP and POP

However, it should be noted that EMSD was specifically designed for efficient short messaging in the context of mobile wireless devices, and thus from inception was meant to be more efficient than protocols designed to handle a wider variety of messages. Deployment and use of EMSD similarly is geared towards messaging scenarios that are a subset of the current global messaging world, such as palmtop devices exchanging messages over an airlink. At the other extreme, in a traditional office environment, concerns like efficient use of communications infrastructure and maximizing the battery life of the devices do not necessarily apply to tethered devices plugged to a standard wall outlet and a high speed (non-air) networking infrastructure.

Within its own domain, EMSD does its job efficiently and admirably and as is clear from the results of this study, is a highly competitive alternative to other messaging protocols.

7 Acknowledgments

This study was performed with the support and assistance of AT&T Wireless Services.

8 Appendix: DLC, IP, TCP, UDP Headers

```
DLC_header:     14 bytes for Ethernet
    Destination MAC:    6 bytes
    Source MAC:         6 bytes
    Eithertype:         2 bytes

IP_header:      20 bytes
    Version+hdr_length: 1 byte
    Type:               1 byte
    Total length:       2 bytes
    Identification:     2 bytes
    Flag+Offset:        2 bytes
    Time to live:       1 byte
    Protocol:           1 byte
    CheckSum:           2 bytes
    Source IPaddr:      4 bytes
    Destination IPaddr: 4 bytes

TCP_header:     20 bytes
```

```
    Source Port:        2 bytes
    Destination Port:   2 bytes
    Sequence Number:    4 bytes
    Acknowledge Number: 4 bytes
    Data offset:        1 byte
    Flag:               1 byte
    Window:             2 bytes
    CheckSum:           2 bytes
    Option:             2 bytes

UDP_header:     8 bytes
    Source Port:        2 bytes
    Destination Port:   2 bytes
    Length:             2 bytes
    CheckSum:           2 bytes
Sun Sparc running
Solaris 2.4
IBM Thinkpad:
MS TCP/IP on Windows 95
```

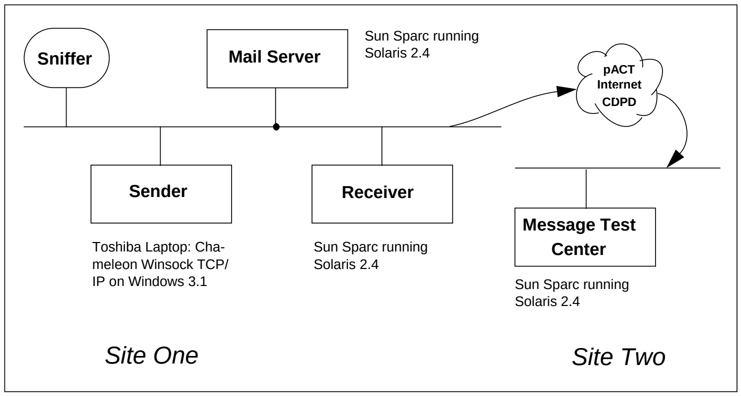

Figure 1: Experimental Setup for Submission

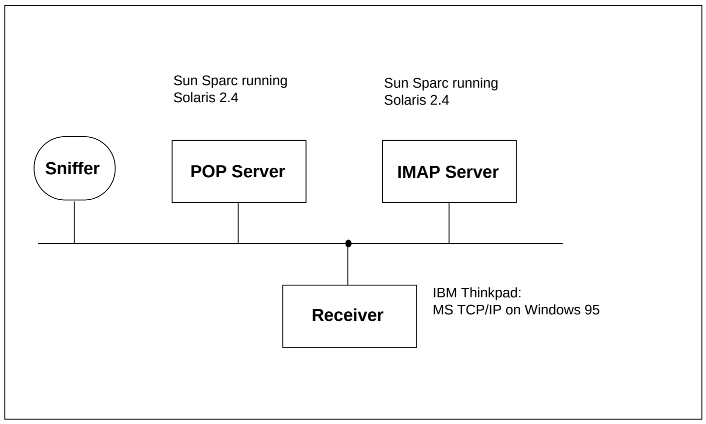

Figure 2: Experimental Setup for Delivery

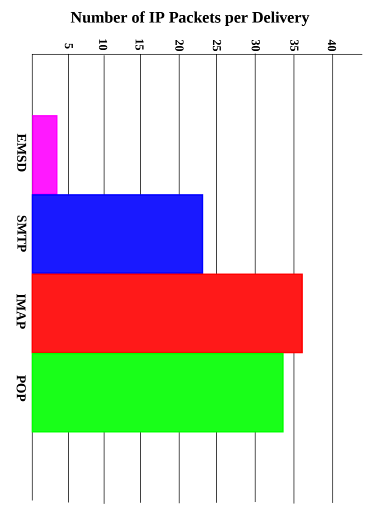

Figure 3: Packets Per Delivery