The State of the Email Address

Mike Afergan
MIT CSAIL
afergan@mit.edu

Robert Beverly
MIT CSAIL
rbeverly@mit.edu

ABSTRACT

Electronic mail is an unquestionably vital component of the Internet infrastructure. While the general perception of email is that it "just works," surprisingly little data is available to substantiate this claim. While SMTP is a mature technology of over twenty years, the architecture is increasingly strained by both normal and unsolicited load. In this paper we seek to provide a greater understanding of the behavior of Internet email as a system using active measurement.

In order to survey a significant, diverse, and representative set of Internet SMTP servers, to which we have no administrative access, we develop a testing methodology that provides an email "traceroute" mechanism. Using this mechanism, we measure email loss, latency, and errors over the course of a month to popular, random, and Fortune 500 domains. Our initial results are *quite unexpected* and include *non-trivial loss rates, latencies longer than days, and significant and surprising errors*. While we present plausible explanations for some of these phenomena, there are several that we cannot, as of yet, explain. By better understanding Internet protocols which lack explicit end-to-end connection semantics, our eventual hope is to derive guidelines for designing future networks and more reliable email systems.

1. INTRODUCTION

Store-and-forward communication is an architectural mainstay in designing networks where connectivity and end-to-end paths are transient. A classic historical example is UUCP [3], a protocol for exchanging files and messages when servers communicated by short-lived point-to-point connections. More recently, store-and-forward networks have experienced a renewal of interest as architects address varied challenges such as sensor and delay tolerant networks [4].

The Internet electronic mail architecture and infrastructure is the preeminent example of a large store-and-forward network. Over twenty years since its introduction, the Simple Mail Transfer Protocol (SMTP) [9, 5] is still used to deliver Internet email. Most regard the system as very well-behaved and very reliable, yet there is surprisingly little data to support this claim. This is particularly troubling as we witness the system come under increased strain from load. In this paper we seek to provide a greater understanding of the behavior of Internet email as a system using active measurement. By better understanding Internet protocols which lack explicit end-to-end connection semantics, our eventual hope is to derive guidelines for designing future networks and more reliable email.

In order to survey a significant, diverse, and representative set of Internet SMTP servers, we employ an active measurement probe that sends an email "traceroute." With this mechanism, we measure email loss, latency, and errors for one month to a fixed set of popular, random, and Fortune 500 domains. Our initial expectation was that testing would produce generally uninteresting results: typically low latencies, few errors and near-zero loss. Indeed, a significant fraction of the large and diverse set of domains we survey behave predictably. However, across a non-negligible portion of domains (including both large corporations and small ISPs) we find quite unexpected results. For example, *we witness several domains experiencing periods of bursty loss, emails with latency greater than 10 days, and unresponsive servers.* The importance of email in modern society and business gives our study added relevance and importance. We present our initial results, including aspects that we understand well and those we are continuing to investigate.

The remainder of this paper is organized as follows. Section 2 gives details of our testing methodology. Sections 3, 4, and 5 present analysis of our three primary metrics: loss, latency, and errors respectively. We conclude with a discussion of the results and implications for future work.

2. METHODOLOGY

There are many popular Mail Transfer Agents (MTAs) such as sendmail [1], postfix [2] and qmail [6], each configurable to suit different installations. In addition to software and configuration heterogeneity, individual SMTP servers process vastly different mail loads and have differing network connectivity. To capture this diversity, our goal is to measure email paths, errors, latency and loss to a significant, diverse, and representative set of Internet SMTP servers. However, we do not have email accounts on, or administrative access to, these servers. Thus, we devised an email "traceroute"

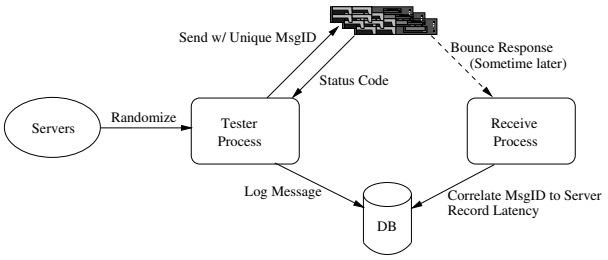


Figure 1: Test Operation. Every 15 minutes, a send process randomizes servers and attempts to deliver emails with invalid recipients to each. These bounce back at a later point. Message IDs in the bounces are correlated in the database.

methodology that relies on *bounce-backs*.

Traditional email etiquette calls for servers to inform users of errors, for instance unknown recipient or undeliverable mail notifications. These descriptive errors, known as bounce-backs, are returned to the originator of the message. Unfortunately, issues such as load or dictionary attacks lead some administrators to configure their servers to silently discard badly addressed email. Indeed, we find that only approximately 25% of the domains tested in this survey reply with bounce-backs. Despite this low return rate, we carefully select a large number of domains for testing in order to provide the most representative cross-section of Internet SMTP servers possible.

The testing system is depicted in Fig. 1. Each email is sent to a randomly selected (and with very high probability invalid) unique recipient. When and if the bounce returns, this unique address allows us to disambiguate received bounces and later to calculate the appropriate statistics. All data is stored in a database to facilitate easy calculations.

For the month of September 2004, we ran traces every 15 minutes to the 1,468 servers of 571 domains summarized in Table 1. For pragmatic reasons, we limit ourselves in this study to performing measurements from a single vantage point. The first set of domains include the members of the Fortune 500. Their servers are likely indicative of robust, fault-tolerant SMTP systems. The second set of domains are randomly chosen. To generate these random domains, we select random legitimate IP addresses from a routing table and perform a reverse DNS [7] lookup. We truncate DNS responses to the organization's root name to form the list. A third set of domains, "Top Bits" are those that source a large amount of web traffic as seen by examining the logs of a cache in an ISP.

Testing these domains scientifically requires removing various levels of non-determinism. These arise from message routing, load balancing and redundancy mechanisms built into DNS [8]:

- MX Records: Mail exchanger (MX) records map a domain name to a set of mail servers. Each server in an MX record has a corresponding preference value. SMTP servers attempt delivery to the most preferred server. If the message cannot be sent, the remaining servers are tried in order of their preference. The preferences allow administrators to configure primary and backup servers. In addition, two servers may have the same preference value in order to load balance. We term servers with the highest preference as 'primary' and all others as 'secondary.'[1]

- A Records: Each server named in an MX record has one or more corresponding address (A) records. The address record may be a single IP address or multiple addresses, again for load balancing or multi-homed hosts.

We introduce a pre-processing step to remove this non-determinism. Each domain is resolved into the complete set of MX records, inclusive of all servers regardless of their preference value. Each MX record is further resolved into the corresponding set of IP addresses. Thus, the atomic unit of testing is the IP address of a server supporting a domain rather than the domain itself.

A round of testing consists of sending a message to every server found in the pre-processing step. Rounds occur every 15 minutes; our data set includes 2880 rounds of traces over the month of September, 2004. Our system chooses, per server and round, a unique random 10 character alphanumeric string as the recipient of the message. It then connects to that IP address and sends the message. The message body is always the same and designed to be innocuous to pass any inbound filtering. The message explains the study and provides an opt-out link.[2] For example, consider a domain `example.com` which has primary and secondary mail servers with IP addresses `1.2.3.4` and `5.6.7.8` respectively. The tester will connect to `1.2.3.4` and send its message addressed to `randstring1@example.com`. It then connects to `5.6.7.8` and sends the message addressed to `randstring2@example.com`. Upon receipt of bounce-backs, the tester can easily determine which bounce corresponds to which originating message based on the unique random string. We performed extensive testing to ensure the testing system could handle a receive load an order of magnitude higher than expected without introducing bias. A round of testing can be summarized as:

[1]MX preference values provide nothing more than an ordered list, thus our separation between primary and secondary servers is arbitrary. We make the distinction in order to clarify the discussion when testing servers that normally would not have received an email (because another MX server with a lower preference value is accepting email).
[2]Over the course of the study six domains opted-out. That system administrators monitor sources of invalid email this closely shows the perceived importance of email.

Table 1: Domain and Server Counts in Survey

Category	Domains	Primary Servers	Total Servers
Fortune 500	282	486	735
Random	216	309	436
Top Bits	73	212	297

One Round of Tester Execution

1. Randomize order of the entire list of MX servers.
2. Query DNS for each domain's primary MX server(s). This step allows us to characterize behavior inclusive of servers that would normally receive email during the round.
3. Generate a unique 10 alphanumeric long recipient address per server IP
4. Attempt delivery of each test message.
5. Record message timestamp and delivery status code in database.

To assist our analysis, our system also records other properties when sending the message. One property we record is whether or not the mail server was a primary or secondary, as specified in the MX records (step 2). Unless otherwise mentioned, our analysis looks only at primary servers, i.e. those which operational MTAs will try first. By querying DNS for the entire list of primary MX servers at the time of the testing round, we can characterize the behavior of the domain as experienced by a proper MTA. (Recall that in normal operation, an MTA will not send email to a server with a higher preference until it has failed delivery to all servers with lower preferences.) We record the return codes of each email. Typically this is the code corresponding to successful delivery, however there are various modes of failure that can occur, both in interacting with the server and in attempting to reach the server. In Section 5, we detail error modes we encountered.

The raw anonymized data collected in this study is publicly available from: `http://ana.lcs.mit.edu/emailtester/`.

Using bounces is a novel and unique way to test a wide variety of domains and gain insight into the behavior of Internet email as a system using active measurement. However, it has a potential limitation of which we are cognizant: whether the behavior seen through bounces, by definition errant mail, is representative of real email. For example, a server could handle bounces differently than normal emails at different periods of time. Depending on load, bounces might be placed in a different queue or simply dropped.

While there is concern over the validity of the bounce methodology, we find no direct evidence to show that it is invalid. We are unaware of any system or software that handles bounces differently. Further, we are unable to find direct evidence in our data to support the theory that bounces are handled differently. For example, we examine the pattern of loss to see if it is correlated with peak traffic hours, perhaps indicating a lower priority for bounces when systems are under load. We do not see this. We also examine greeting banners (the `HELO` message) to determine if the single IP of a mail exchanger actually represents multiple virtual machines with different configurations. While we find different banners for the same IP address, the existence of such an architecture does not correlate with or seem to explain the losses.

Table 2: Per-Category Bounce Response Summary Statistics

Category	Domains Ever Responding (%)	Servers Ever Responding
Fortune 500	79 (28%)	515
Random	118 (55%)	203
Top Bits	29 (40%)	68

Thus, while our system is not perfect, we hope it represents a step forward in understanding a large, complex system such as email. More importantly, while we present plausible explanations for some of the phenomenon we see, there are several that we cannot, as of yet, explain. The remainder of this paper presents and analyzes our testing results.

3. LOSS

Loss rate is perhaps the most interesting and confusing metric that we consider. Prior to testing, our initial belief was that servers would either always or never return a bounce. We therefore expected that over time a clear division between the two groups would emerge. Focusing on the group of servers that always respond, we then hoped to quantify loss rates by looking at the (presumably rare) exceptions. In this section we show that while many domains behave in this fashion, a non-negligible portion exhibit surprising results.

We define a lost email to be one where the email is sent to a server and the server returns a succession of successful status codes during delivery, but no bounce for that message ID is ever received. We ran our analysis a sufficiently long time after sending the last email in September; all emails had 21-51 days to bounce and be received.

Table 2 summarizes those servers that *ever* responded with a bounce. In sum, only 25% of the domains ever bounced an email addressed to an invalid recipient.[3] As discussed, a primary reason this rate is so low is due to spam concerns.[4] However, in conjunction with our large number of carefully selected domains, this bounce rate still provides us with a representative sample of Internet SMTP servers to measure.

[3]While only 25% of the domains responded, approximately 60% of the servers responded. This suggests that domains with more servers are more likely to respond, a phenomenon we do not fully understand.

[4]Even if this behavior is intentional, we note that operating with silent failures is unfortunate as bouncing mistakenly addressed emails is valuable for both senders and receivers.

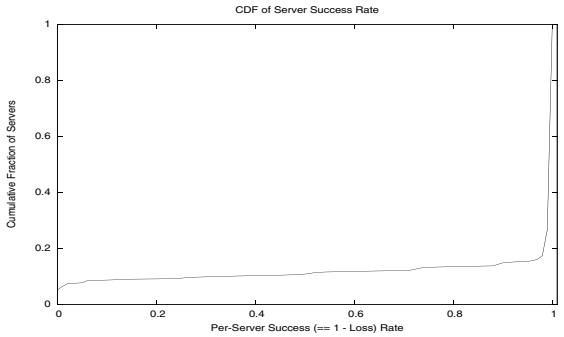


Figure 2: Success rate per server CDF inclusive of primary servers that returned at least one bounce. Most surprising is the 15% of servers that experienced loss rates between zero and one.

Unexpectedly, within the set of servers responding, there is a large variance in the response rates. We calculate success rate simply as the ratio between the number of bounces received from a server to emails successfully delivered to that server:

$$\text{Success Rate} = \frac{\text{Received Bounces}}{\text{Successfully Sent Emails}}$$

where an email is considered successfully sent only if the server returns the proper sequence of successful status codes during delivery. Fig. 2 presents the cumulative distribution of per-server success rates for all servers where: i) the server is advertised by DNS as a primary mail exchanger, ii) at least 500 emails were successfully delivered to this server, and iii) the server returned at least one bounce. We break these servers into classes, some of which are easily explained and some of which are quite surprising.

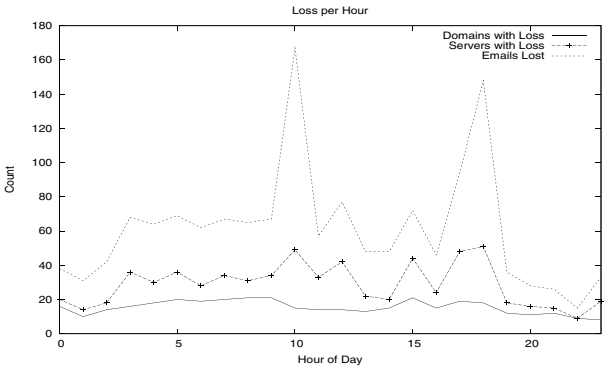
- *Servers that Always Respond.* This represents approximately 73% of all servers. As discussed, this category is expected and not surprising.
- *Servers that Rarely Respond.* Approximately 4% of servers responded with bounces to more than 0%, but less than 0.01% of emails. For example, some servers respond to our succession of invalid emails with a single bounce.
- *Servers with Slight Loss.* While we predicted the existence of this category, *it is much larger than we expected*. Overall, 6% of servers fell into this class, as seen in Table 3. Further analysis is even more perplexing. For example, consider only the set of servers that responded to greater than 99.9% of emails and which responded to 100% of emails for at least 3 full consecutive days. From this behavior we assume that the *intended behavior* for these servers is to always respond with a bounce. However, within this class of servers we still see loss. Specifically, 125 of 486,177 emails (0.03%) were lost in this class. Not only is this surprising, but also likely represents *incorrect, anomalous, and/or undesirable* behavior.
- *Servers with Moderate Loss.* Seeing a loss rate in the range (0.1,5%] for any server is unexpected. Therefore the degree to which it is present in our results is quite shocking. Digging deeper into this loss reveals no clear explanation. Some servers exhibit loss patterns that do not seem correlated with any other property. In other cases, we observe that all servers for a domain will suffer from significant and correlated loss. This latter case suggests some system-wide change, but the effect on bounces and what it implies about standard email is not yet understood.
- *Servers Exhibiting Persistent Loss.* The existence and size of this category of servers is also both unexpected and as of yet unexplained. We see that there are 60 servers with between 5% and 99.9% loss during the experiment. In fact some had close to 50% loss. *We do not understand how or why this is happening.* Since the email body and process is identical for each email, these servers seem to exhibit non-deterministic behavior. The only correlation we find thus far is that often these anomalous servers belong to the same domain. Manual inspection and interaction with the mail servers suggests that the behavior for some servers may be correlated to combined email body and recipient headers, perhaps suggesting a spam filter. Nonetheless, such a highly non-deterministic and sensitive filter is surprising since it it likely undesirable. As discussed previously, a second possibility is servers deprioritizing bounces during periods of load although subsequent analysis of loss periodicity revealed no correlation.

Table 3: Fraction of Servers vs. Success Rate per Category

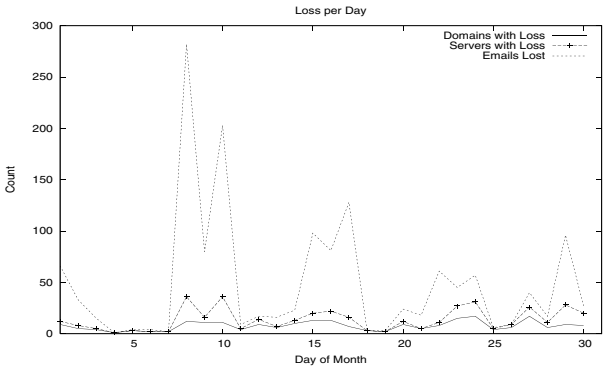
Success Rate(%)	F500(%)	TopBits(%)	Rand(%)	All(%)
100	36	16	53	38
≥ 99.9 & < 100	7	3	5	6
≥ 95.0 & < 99.9	13	26	5	12
> 0.01 & < 95.0	8	0	3	6
> 0 & ≤ 0.01	5	0	2	4
0	31	55	32	34

Table 3 examines the distribution of servers among the success rate groupings per category. Several items bear notice in this table. First, the Random servers are far more likely to have a perfect response rate than corporate servers in the Fortune 500 set. Second, corporate servers are more likely to have significant loss. Both observations could be due to load or perhaps the complexity of large installations.

Given the observed periods of sustained loss, we examine loss as a function of time. Fig. 3(a) plots loss per hour. Interestingly, loss spikes at 10am and 6pm Eastern time. In Fig. 3(b), we plot loss per day in an attempt to find relationships between emails lost, servers with loss, and domains with loss. The large spikes that we observe could, for example, be indicative of measurement error. If measurement error were the cause, we'd expect this loss to be distributed over multiple servers and domains. However, the graph shows that days of high loss in messages do not exhibit



(a) Loss Per Hour: The rate of loss spikes at 10am and 3pm Eastern



(b) Loss Per Day: Significant increases in loss are concentrated in one or more domains having loss across multiple servers.

Figure 3: Periodicity of Bounce-Back Loss Rates

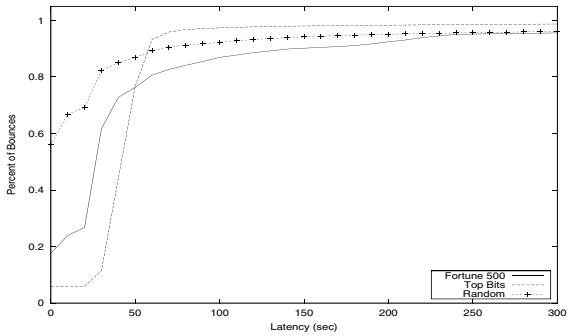


Figure 4: Cumulative Distribution of Bounce Response Latency

a significantly larger number of domains or servers with loss. Instead, deeper inspection reveals that most spikes (particularly larger ones) are explained by one or more domains experiencing significant loss across multiple servers. The cause of these periods is again not something we understand. However, the fact that the change is both sudden and moreover *temporary* suggests that it is not intended behavior.

4. LATENCY

The next metric we examine is latency. The time between transmission of a message and receipt of the corresponding bounce is recorded as the latency of that message. Figure 4 plots the cumulative fraction of messages versus latency for each of the three domain classes. While more than 90% of the bounces are received within five minutes of their transmission, we see that the distribution is strongly heavy tailed. While we did expect outliers, we find it quite surprising that we received 295 (0.035%) of the bounces more than 24 hours after sending the initial email. In fact, one bounce arrived 30 days after the initial email was sent!

To better understand the cause of bounces with large latencies, we systematically examined the headers returned by the ten slowest emails in our study. Because messages are timestamped by each server along the delivery path, the headers provide insight into where delay occurred.[5] These ten correspond to five distinct domains, four of which are corporate entities and one a university. In detail, we found:

- The largest delay occurred within a corporation's email infrastructure. The message was received by the first-hop SMTP server which generated a bounce. The bounce traversed two internal SMTP hops where it then sat for approximately 34 days before being returned to our test system.[6]
- The next four emails with the largest delays were sent to the same commercial domain within an hour and fifteen minute period. These messages were not processed for 21 days and then bounces were returned in immediate succession. The bounces traversed four internal SMTP hops with no delay. We can only suspect that some single event at this time caused all four mails to be delayed in a queue for the entire 21 days.
- One corporate domain outsources their email service to a major ISP. Between the first and second SMTP hops, the email was delayed for approximately 18 days. Once the mail was eventually delivered, the bounce was returned immediately.
- Three emails were received by the final SMTP host where they were delayed for two days before generating a bounce. The bounce then took two weeks to be returned to our tester.

[5]However, we often see servers set with the incorrect time adding erroneous timestamps in the header.

[6]While 34 days is a very long period of time, one author recently experienced a delay of over 42 days with one of his personal emails. The email was sent from the MIT mail system (which has no relationship to the testing system) and was delayed for over 42 days in reaching the mail system of another large Boston-area academic institution.

- The bounce with the tenth highest delay was returned from a university. The message arrived at the first hop SMTP server which subsequently took 11 days to return a bounce.

A final interesting feature of the latency data is the qualitative difference between the classes of domains. The majority of bounces for Fortune 500 and Top Bits take at least 30 seconds whereas the tester receives approximately 60% of the bounces from random domains in near-zero time. This behavior is not surprising as the Fortune 500 domains are likely handling significant load, and performing more spam filtering.

5. ERRORS

The final result we present is errors our tester experienced attempting to deliver email. In contrast to the loss and latency results, errors sending email are unambiguously indicative of true errors since they exactly model the behavior of delivering a mail to a valid email address. From the perspective of a domain, component failures (e.g. an individual mail exchanger) may not be harmful since the system gracefully handles the problem. SMTP servers that cannot connect will retry or try a secondary. Nonetheless, we find significant and surprising errors over the course of our survey.

As explained, the tester attempts to deliver each message only once and logs a status code. If the tester is able to connect to the server, but delivery is unsuccessful, the status code is the error response code from the server. SMTP response codes are detailed in [9] and [10]. However, if the tester is unable to interact with a given mail server, the status code is generated by our system.

There are four scenarios where the tester is unable to interact with a server and delivery email. In the first, the tester successfully establishes a TCP connection, but the remote server is silent for more than 30 seconds. We term this case 'No HELLO' as the server does not respond with any welcome banner. The second case is a server which responds to the hello handshake, but then goes silent for 30 seconds after attempting to send mail. These we call 'No OK.' If the tester is unable to establish a TCP connection, it tries pinging the server. Thus, the last two error modes are no connection and pingable or no connection and not pingable.

Table 4 summarizes the errors encountered interacting with primary servers to which we sent at least 500 other emails successfully (status code 250). As such, we consider these servers generally well-behaved. There are several interesting observations:

- The leading error (both in absolute terms and as a percentage of a category) is "Mailbox unavailable", represented by 550 and 553 error codes. While such errors are not surprising, they are unexpected among the set of servers we are considering, which behaved differently (by returning status code 250) more than 500 times. It is also interesting to see the 550/553 error code only among random domains. These errors are from 5 different domains.

Table 4: Error Statistics, Percent of Total Messages Sent

Error Type	Fortune 500	Random	Top Bits
Local Error (451)	0	0.01	0
Bad Command (503)	0	0.14	0
Bad Mailbox (550+553)	0	5.6	0
No HELLO	0	0.006	0.011
No OK	0.15	0.42	1.7
No Conn and No Ping	0.34	2.22	2.10
No Conn and Ping	0.06	0.14	0.12

- The next largest error is unreachable mail servers. Again, we only consider servers that are generally reachable and also still listed as a primary MX. We see that 0.34-2.22% of messages are sent to primary MXes that were completely unreachable.

- Another interesting pathology is servers that responded to ping but not SMTP. This represents 0.06-0.14% of messages.

- 426 primary servers across 202 domains always encountered an error. [7]

6. CONCLUSIONS AND FUTURE WORK

The SMTP architecture and infrastructure is an integral part of today's Internet and society. In this work, we developed a methodology that gives us insight into the performance of a widely diverse and representative set of approximately 600 email domains. Over the month of September 2004, we analyzed three metrics: loss, latency and errors. Our initial expectation was that testing would produce uninteresting results. While the majority of servers behave as expected, a non-negligible number had either persistent loss or periods of bursty loss. Perhaps more interestingly, many servers exhibit non-deterministic behavior which we have yet to fully explain. Across the sets of domains we consider, we see a qualitative difference between the latency and loss behaviors. Our latency results show a heavy-tailed distribution with outliers that include multiple-day delays. Finally, we analyzed the errors our tester encountered including server and connection problems.

This initial survey suggests many avenues for further research. We would like to investigate the sources of non-deterministic behavior and validate our testing methodology across a large set of servers. As alluded to in the analysis of Section 4, our system maintains the SMTP headers which allows us to reconstruct the message path. Further analysis here includes further examining latency as well as other metrics such as path stability in the SMTP system. Furthermore, we are examining other measurement techniques (passive and active) with which we may be able to compare the results of this study.

[7]These are not reflected in the table since only servers which successfully received more than 500 emails are included.

Acknowledgments

The authors would like to thank the members of the MIT Advanced Network Architecture group, especially Steve Bauer, Karen Sollins and John Wroclawski. We also thank the reviewers and editors for their helpful comments and suggestions.

7. REFERENCES

[1] B. Costales and E. Allman. *Sendmail*. O'Reilly, 3 edition, 2002.

[2] K. D. Dent. *Postfix: The Definitive Guide*. O'Reilly, 2003.

[3] M. R. Horton. *UUCP Mail Interchange Format Standard*. Internet Engineering Task Force, Feb 1986. RFC976.

[4] S. Jain, K. Fall, and R. Patra. Routing in a delay tolerant network. In *Proceedings of ACM SIGCOMM*, 2004.

[5] J. Klensin. *Simple Mail Transfer Protocol*. Internet Engineering Task Force, Apr. 2001. RFC 2821.

[6] J. R. Levine. *qmail*. O'Reilly, 2004.

[7] P. V. Mockapetris. Domain names - concepts and facilities. Request for Comments 1034, Internet Engineering Task Force, Nov. 1987.

[8] C. Partridge. *Mail routing and the domain system*. Internet Engineering Task Force, Jan. 1986. RFC 974.

[9] J. B. Postel. *Simple Mail Transfer Protocol*. Internet Engineering Task Force, August 1982. RFC 821.

[10] G. Vaudreuil. *Enhanced Mail System Status Codes*. Internet Engineering Task Force, Jan. 2003. RFC 3463.