Tuning Sendmail for
Large Mailing Lists

NLUUG
Antwerp, Belgium

February 27, 1997

Rob Kolstad, Ph.D.
<kolstad@bsdi.com>

Overview

- The problem
- Goals
- Initial state
- What we did (with graphs)
- Where we ended up

The Problem

- Big mailing lists increasingly popular
  - News less useful
  - Push vs. pull

- Big mailing lists are big
  - 500-15,000 recipients
  - 1-100+ messages per day

- Message delivery slow and resource intensive

Some Goals for Mailing Lists

- Fast delivery (low latency)
- Reasonable consumption of resources
- Existing tools
- Monitor results
- Easy to administer
Initial Situation

- inet-access@earth.com mailing list
- 1,500 recipients
- 40-110 messages per day
- Delivery times > 5 hours
- System load high, paging high

Initial Data Gathering

- 50,000 messages in mail queue
- > 100 sendmail processes
- Each process > 2.5 MB
- Load average high; CPU not fully used
- Disks medium busy
- Queued deliveries often catch up overnight

Zeroth Trial

- Tony Sanders sorted outgoing mail queue
- Gathered ‘delivery stats’ nightly
- Gave priority to those who responded quickly
- Great for those at front of queue
  - But still slow for those at end

First Suggestion

- Split list into 75 lists of 20 each
- Completely overruled by sysadmins
  - Not enough RAM
  - Too many processes
- Kolstad unconvinced
Second Try, First Suggestion

- Split list into four lists of 375 each
- Sysadmins not pleased
- Kolstad wins these arguments; he’s the president

Results of First Suggestion

- Primitive analysis tools show 4x improvement in throughput
  - Looks like mail is delayed < 2 hours
  - Looks like delivery rate is increased
- Lots-o-processes
  - Lots of RAM use
  - Disks OK
- Hard to measure throughput cuz mail queue pieces only updated every ten deliveries
- So, we changed the update rate to update on every delivery
  - Improved monitoring
  - HUGE mistake

Next Suggestion

- Trend toward more lists of fewer elements
- Construct more monitoring tools
  - Instantaneous rates
  - Summary pages
- Watch processes, disk I/O, network I/O

Results

- Ever fewer elements → ever faster deliveries
- Single recipient sendmail processes never use much RAM (!)
- Disk I/O continuing to increase
- Network I/O very low
- Delivery times now below one hour (yay)
- Queue sizes now considerably reduced
Head Scratching

- RAM usage now low (!?)
- Where is/are the bottleneck(s)?
  - Disks are getting busy
  - CPU isn't that busy
  - Load average isn't getting worse
  - Network isn't the problem
  - Something else???
- What is the maximum throughput for mail delivery?

Time to Build Monitoring Tools

- mailqq (slow! but easier way of seeing queues):
  
```bash
#!/bin/bash
mailq | perl -e '$junk = <STDIN>; $junk = <STDIN>; $i = 0;
print "##", $i = 0;
while (<STDIN>) {
  if (!/\t/) { $i++; next unless eof(STDIN); }
  if ($line || eof(STDIN)) {
    printf("%5d %s\n", $i, $line);
    $sum += $i; $i = 0;
  } chop($line = $_);
} printf "%5d TOTAL\n", $sum;
```

Batchstat Output

Mail Delivery Performance

- Look at those spikes!

Realstat Output During Slow Period

<table>
<thead>
<tr>
<th>Time</th>
<th>Value_1</th>
<th>Value_2</th>
<th>Value_3</th>
</tr>
</thead>
<tbody>
<tr>
<td>9:00:00</td>
<td>480/2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9:00:15</td>
<td>960/4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9:00:30</td>
<td>960/4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9:00:45</td>
<td>2160/9</td>
<td></td>
<td>*</td>
</tr>
<tr>
<td>9:01:00</td>
<td>1920/8</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>9:01:15</td>
<td>720/3</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>9:01:30</td>
<td>2400/10</td>
<td>**</td>
<td>**</td>
</tr>
<tr>
<td>9:01:45</td>
<td>720/3</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>9:02:00</td>
<td>1440/6</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>9:02:15</td>
<td>960/4</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>9:02:30</td>
<td>960/4</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>9:02:45</td>
<td>1440/6</td>
<td>*</td>
<td>*</td>
</tr>
</tbody>
</table>
Mailstat (Slow Day)

```plaintext
mailstat: Sat Feb 22 09:05:40 MST 1997
failed deliveries
MMM DD HH mhosts/recipt mhosts/recipt
--- --- --------------- ---------------
Feb 22 02 433/ 612 1111/ 1111
Feb 22 03 495/ 696 1298/ 1298
Feb 22 04 431/ 615 1137/ 1137
Feb 22 05 421/ 610 810/ 810
Feb 22 06 422/ 606 717/ 717
Feb 22 07 411/ 587 931/ 931
Feb 22 08 427/ 616 1039/ 1039
Feb 22 09 22/ 22 105/ 105
--- --- --------------- ---------------
Totals 3062/ 4364 7148/ 7148
```

Mailstat (Few Deliveries, Many failures)

```plaintext
mailstat: Sat Feb 22 09:03:49 MST 1997
failed deliveries
MMM DD HH mhosts/recipt mhosts/recipt
--- --- --------------- ---------------
Feb 22 03 18811/ 18811 522/ 522
Feb 22 04 27065/ 27065 574/ 574
Feb 22 05 29342/ 29346 1738/ 1887
Feb 22 06 29973/ 29978 8/ 8
Feb 22 07 26668/ 26675 1556/ 1690
Feb 22 08 11768/ 11787 1347/ 1464
Feb 22 09 1893/ 1896 566/ 606
--- --- --------------- ---------------
Totals 145520/145558 6311/ 6751
```

What Is Max Speed?

- Wrote `mailtest.c`
  - Opened mail port
  - Sent shortest possible message
  - Completed
- On PPro/200: ~30 ms → 120000 messages/hour
- Only bottlenecks were CPU and network

Further Analysis

- Continued to increase parallelism
  - Went to 100 lists of 15-20 people each
  - Decreased delivery time
  - Machine very busy
  - Disks incredibly busy
- Stupid decision was to update stats for ‘every delivery’
  - Great for real-time understanding of deliveries
  - `/var/log/maillog vs. mailq`
- Synchronous disk operations were destroying performance
- → So, went back to ‘update every 10 deliveries’
Performance Increased Dramatically
Mail Delivery Performance

Notes about Performance
- Note that peaks are very high
  - Even averaged, they are closing in on theoretical maximum
- Performance depends on other things, too
  - Number of hosts unavailable
  - Number of messages available to deliver (!)

Next Analysis
- Why is mail queue so long?
  - [...]  
    nrecipients  length  date      sender
    3  BAA08598  1554  Sat Feb 22 01:02  <inet-access@earth.com>
    3  BAA08677  1017  Sat Feb 22 01:29  <inet-access@earth.com>
    3  FAA10201  1438  Sat Feb 22 05:24  <inet-access@earth.com>
    3  FAA10208  1438  Sat Feb 22 05:24  <inet-access@earth.com>
    3  HAA10369  1527  Sat Feb 22 07:46  <inet-access@earth.com>
    3  IAA10524* 423  Sat Feb 22 08:52  <inet-access@earth.com>
    4  HAA10371  1527  Sat Feb 22 07:46  <inet-access@earth.com>
    4  HAA10383  1527  Sat Feb 22 07:46  <inet-access@earth.com>
    4  IAA10544  423  Sat Feb 22 08:53  <inet-access@earth.com>
    4  IAA10556  423  Sat Feb 22 08:53  <inet-access@earth.com>
    5  IAA10041* 423  Sat Feb 22 08:53  <inet-access@earth.com>

- Each message has a few ‘stragglers’

What About Those Stragglers?
- 33  nathan@netrail.net
- 46  davek@melita.com
- 82  todd@acc.com
- 127  glennh@netstation.net
- 127  ispmail@zhi.dialup.access.net
- 127  jnussbaum@americandata.net
- 127  kevin@rrt.com
- 127  mp3@cyber-gate.com
- 127  nevin@shadowave.com
- 127  tcosta@biznm.com
- 164  rdavis@masschaos.de.convex.com
- 200  whenpigsfly@worldsrv.net
- 388  cbrown@matnet.com
- 559  robert_thompson_at_usr-cssl@robogate2.usr.com
- 593  berney.ortiz@mailserver.oig.state.pa.us
- 595  list.inet-access@optimum.net

- Small number of recipients never accepting mail!
What Happens When Recipient Unavailable?

- Sendmail tries to connect host
  - Maybe succeeds (but host turns out to be slow or net is losing lots-o-packets)
  - Maybe not
- Each step in the protocol from connection through completion has long time-out
  - Like as much as 300 seconds
  - So that particular sendmail process idles for five minutes
  - Reduces throughput
- Amazing fact: at any point in time, 1-3% of recipients are unavailable
  - And these are ISPs!
  - But they can’t control many kinds of outages

Next Step

- Reduce timeouts for initial contact/mail transmission
  - Configurable in sendmail
  - Reduced them 5x
- Speeds up initial mail delivery
  - Of course, some messages never delivered
  - So a second sendmail.cf file with slower timeouts was created; ‘reaper’ process runs 3x/hour

High Points of Modifications

- Lots of parallelism (100 processes in parallel)
- Reduce impact of unavailable recipients
- Continued effects of stragglers
  - Mailq commands run really slowly
  - Sendmail running entire queue is very slow
  - Needed to pay even more attention to stragglers to reduce queue search time

Reducing Queue Search Time

- Can’t remove people from list for one bounce – or even two days of bounces
- Created 10 more queues to run separately
  - Jobs moved from queue to queue when older than specified amount
  - Ever more ‘reaper’ processes running those (presumedly smaller) queues
- Just never seemed to help :(
Yet To Try

- Coalesce stragglers into single message/recipient-list pair
- Just delete notes older than, say, 24 hours from queue (they can always look at an archive)
- Rewrite sendmail for:
  - High constant number of transmission processes
  - Use of extended SMTP to send multiple messages once a machine is up
- But, to be fair, already fairly close to achieving highest possible bandwidth given speed of network connections
- Easy to use multiple CPUs to deliver more messages in parallel
  - And still not pushing T-1 speeds to limit yet (!)

Conclusion

- Not difficult to reduce latency dramatically
- Kolstad has script to insert in /etc/aliases to break message into parts (kolstad@bsdi.com for details – it's only 99% great)
- 95% mail delivery time was reduced from 5 hours to 3.5 minutes
- Unavailability of recipients still the biggest problem
- Probably about as good as can be done without redoing sendmail
  - But Russ Cox is writing a mail delivery agent
  - And has these results :)