

Design and Implementation of Highly Scalable E-mail Systems

Brad Knowles	with	Nick Christenson
Systems Architect, Belgacom Skynet SA/NV <i>blk@skynet.be</i>		Senior Software Engineer, Sendmail, Inc. <i>npc@sendmail.com</i>

LISA XIV, 8 Dec 2000

Copyright © 2000 Brad Knowles, all rights reserved

Apologies

- Less “Implementation”
- More “Fundamentals & Architecture”
 - This stuff is hard
 - This stuff is surprisingly hard, even for experienced professionals
- Nick unable to attend

Outline

- Review of Major Publications
- Review of Typical POP3 Implementations
 - Enhancements
- Contrast with IMAP
 - Implications of protocol differences
- Functional Architecture
- Detailed Architecture

Information Sources

- Academia
 - Build vs. Buy
 - Frequently re-invent the wheel
 - Small Scale
 - Occasionally revolutionary
- Commercial
 - Buy vs. Build
 - Time-to-market crucial
 - Large Scale
 - Usually Evolutionary
 - Any revolutions are usually in the area of scaling

Publication Categories

Lists		√
MTAs	√	√
POP3	√	√
IMAP	√	○
Distr.	√	
	Small	Large

Publications Review

- Large Mailing Lists
 - Kolstad97
 - Chalup98
- MTAs
 - Knowles98
 - Christenson99
 - Venema98
 - Golanski2000
- POP3 Mail Systems
 - Grubb96
 - Christenson97
 - Horman99
- IMAP Mail Systems
 - Stevens97
 - Beattie99
- Distributed
 - Yasushi99

Publications Review

- POP3 Mail Systems
 - Grubb96
 - Christenson97
 - Horman99

Review: Grubb96

- Problem
 - NFS mail spool/hub configuration using 7th edition mailbox (mbox) format for ~5000 users could not scale to ~20,000 users

Review: Grubb96

- Solutions
 - Front-end MXes handle incoming communications
 - Back-end servers handle mailboxes
 - Front-ends “trickle” feed via smaller number of cached connections to back-end servers
 - Separate syslog data onto separate disk
 - Tweak kernel, NFS server, & NFS client settings
 - Change client config to use mailhub name based on userid via DNS CNAME records

Review: Grubb96

- Solutions, continued
 - Implement additional mailhubs to serve chunks of user community based on CNAMEs
 - Turn on POP3 & IMAP2bis w/ 7th edition mailbox (mbox) format on each new “post office” server
 - Provide users with POP3/IMAP clients
 - Turn off NFS
 - Convert POP3/IMAP2 mbox → POP3/IMAP4 Cyrus on each “post office” server

Review: Grubb96

- Applicability
 - Does cover entire mail system, not just MTA
 - Doesn't really tell us anything about how POP3/IMAP system is managed
 - Doesn't scale
 - Users have to know too much about post office configuration
 - Requires CNAME RR for each customer
 - Mixes inbound and outbound services on same machines

Review: Christenson97

- Problem
 - Existing information on architecture for robust large-scale mail systems is scarce, doesn't address key issues, and doesn't scale to required levels

Review: Christenson97

- Solutions
 - Front-end MXes handle external communications
 - Secondary MXes do not attempt delivery to back-end, in case there is a problem with deliveries
 - Front-end MXes do not authenticate recipient names
 - All machines are dataless
 - Modify LDA to handle authentication methods, mailbox formats and quotas

Review: Christenson97

- Solutions, continued
 - Back-end servers **do** accept outgoing SMTP mail
 - Do not do local delivery, pass to inbound MXes instead
 - POP3 code must also be modified to know about authentication methods and mailbox format
 - All data (mailboxes and *sendmail* mqueues) stored on NetApp NFS servers
 - Mail spool directories hashed and split across multiple NFS servers

Review: Christenson97

- Solutions, continued
 - Dynamically balance mailboxes or expand capacity
 - Both POP3 daemon and LDA know about “old” vs. “current” mailbox location
 - POP3 daemon moves mailbox if necessary
 - POP3 & LDA modified to use database for user authentication, avoiding use of `/etc/passwd`
 - Cluster & fail-over for user authentication database

Review: Christenson97

- Solutions, continued
 - NFS file locking doesn't work reliably
 - Replace w/ lockfiles on separate shared NFS server
 - Uses semantics of `open()` system call with exclusive write
 - Lock system needs to be replaced to scale further
 - Custom clustered servers w/ shared RAID & unbuffered writes
 - Query different lock servers for different ranges of mailbox names

Review: Christenson97

- Applicability
 - Does cover entire mail system in centralized fashion
 - NFS servers are SPOFs
 - UDP & RPC are major security hazards
 - Customized code is expensive to maintain
 - Specific to POP3, does not cover IMAP

Review: Horman99

- Goal
 - Define architecture to scale mail systems transparently to multiple servers

Review: Horman99

- Solutions
 - Multiplex SMTP
 - Single layer
 - If recipient not local, must forward to correct server
 - With growth, amount of forwarding approaches 100%
 - Dual layer
 - No local recipients on front-end servers
 - Must always forward to correct back-end server
 - Add layer 4 load-balancing switches to hide number of machines accepting SMTP connections

Review: Horman99

- Solutions, continued
 - Multiplex POP3 & IMAP
 - Single layer
 - Must handle local connections
 - Must also proxy for remote connections
 - Dual layer
 - Dedicated content-free proxies

Review: Horman99

- Solutions, continued
 - Mailbox migration
 - Calculate metric for each server over reasonable time
 - Migrate only if a server deviates significantly from avg.
 - Order users by decayed metric cost
 - How long are migrations remembered?
 - How long since this mailbox migrated?
 - Generate user list probabilistically

Review: Horman99

- Solutions, continued
 - Mailbox migration, continued
 - Move from most heavily loaded server to least heavily loaded server(s)
 - Move only if result would not push recipient over average
 - Continue with next most heavily loaded server until no more migrations are possible

Review: Horman99

- Applicability
 - Covers only POP3 and not IMAP
 - Proper load balancing requires programming for peaks, not long-term averages
 - Focuses exclusively on “free” or “cheap” solutions
 - Too much time/space spent on less important issues
 - Not enough detail provided where needed

Publications Review

- IMAP Mail Systems
 - Stevens97
 - Beattie99

Review: Stevens97

- Statistics
 - 60,000 accounts
 - 4,000 peak concurrent logins
 - 1.4 million logins per month
 - 500,000 messages/day
 - 1,083 peak messages/minute
 - 65,000 peak messages/hour

Review: Beattie99

- Goals
 - Implement and document replacement mail system for ~30,000 users
 - Reliable
 - Secure
 - IMAP & SMTP
 - Web interface available
 - Quotas

Review: Beattie99

- Solutions
 - Mix & match software on cluster of commodity computers running Unix-like OS
 - UW imapd
 - Exim
 - Apache/mod_perl
 - WING (Web IMAP/NNTP Gateway)
 - PostgreSQL
 - BIND
 - Custom account & cluster management tools

Review: Beattie99

- Solutions, continued
 - Two front-end servers are firewalls & nameservers
 - Configured for fail-over
 - IMAP servers hold all per-user filestore
 - IMAP, POP3, & SMTP (public)
 - NFS export to other nodes (private)
 - Vacation messages
 - Forward files
 - Personal home page links

Review: Beattie99

- Solutions, continued
 - WING servers hold only temporary data
 - HTTP (public)
 - IMAP & NFS to IMAP/NFS servers (private)
 - SQL to front-end/firewall servers (private)
 - Each user has DNS entry
 - `username.herald.ox.ac.uk`
 - CNAME alias to home IMAP node

Review: Beattie99

- Solutions, continued
 - Front-end machines are
 - Cluster nameservers
 - SMTP & HTTP login gateways
 - DBMS servers for all user config data
 - Generate mailer tables and push to other nodes

Review: Beattie99

- Solutions, continued
 - Security
 - Front-ends are firewalls
 - IMAP & WING servers trust front-ends 100%
 - IMAP servers export `~foo/wing` directory owned by `httpd` for each user *foo*
 - Automap games to handle mounts
 - Break-in on WING servers allows modification of forward files, vacation messages, & personal links but **NOT** mail

Review: Beattie99

- Solutions, continued
 - Failure analysis
 - IMAP
 - Mail stored on RAID5
 - » Immune to single disk failure
 - » If node dies, all users on that node lose access
 - WING
 - Current sessions die
 - 1/n login attempts fail until server manually removed from lists
 - Switch = SPOF

Review: Beattie99

- Solutions, continued
 - Failure analysis
 - Front-end
 - DNS continues
 - IP traffic dropped but can reconnect
 - SQL failover currently manual
 - » Lose config changes since last sync
 - Changes
 - Added outbound mail relay servers to speed up acceptance of mail from dumb clients

Review: Beattie99

- Statistics
 - Recent average week
 - 2 IMAP servers, 2 WING servers
 - 82,000 total connections to IMAP servers
 - 113,000 mail deliveries by IMAP servers
 - 95,000 local
 - 18,000 outgoing
 - 26,000 outgoing messages from WING
 - 66,000 IMAP sessions (including 38,000 WING)
 - 120,000 POP3 sessions

Review: Beattie99

- Applicability
 - Very small scale
 - We have ~7.5x their # of users
 - We do ~38x their number of inbound mail messages
 - We do ~35x their number of local mail deliveries
 - We do ~64x their number of outbound mail messages
 - We don't know how many more POP3 sessions we do
 - Too expensive too track

Review: Beattie99

- Applicability
 - Not scalable, not enough functional decomposition of services
 - Front-end/firewall/nameserver/user meta-data server doing way too much
 - IMAP servers should not be used as outbound mail relays
 - IMAP servers should not be used as NFS servers

Publications Review

- Distributed
 - Yasushi99

Review: Yasushi99

- Goals
 - Build and describe distributed, replicated, clustered, automatically load-balanced, functionally homogenous mail system

Review: Yasushi99

- Solutions
 - Use commodity hardware and OS
 - Write all custom application code
 - Mailboxes fragmented at message level
 - Replicated across two servers
 - Distributed across as many as four servers
 - All servers run all protocols
 - SMTP in & out, POP3, IMAP, User metadata database

Review: Yasushi99

- Solutions, continued
 - Soft limit of four distributed servers can be exceeded if one or more nodes is down
 - Some affinity of distributed servers is maintained to reduce latency
 - Automatically discover new resources
 - Detect and route around failures automatically
 - Balance cluster automatically across all nodes

Review: Yasushi99

- Solutions, continued
 - Claims to be lock-free because POP3 and IMAP require only convergence to consistency over time
 - “Load” defined as boolean + integer
 - Disk full or not?
 - Total number of outstanding potential I/O requests
 - Node with full disk is always considered to be “very loaded”
 - Used only for reading and deleting mail

Review: Yasushi99

- Solutions, continued
 - Testing methodology
 - Avg. msg size 4.7KB w/ fat tail to 1MB
 - SMTP traffic = 90% of load
 - POP3 traffic = 10% of load
 - Compare against *sendmail* 8.9.3 + *ids-popd*-0.23
 - Custom load-generation programs
 - POP3 test program collects and deletes all mail for user
 - Linux async writes are used

Review: Yasushi99

- Solutions, continued
 - Testing results
 - One node w/ no replication and one IDE disk could handle ~23 msgs/sec.
 - Adding two SCSI disks to single node, it could handle ~105 msgs/sec.
 - Two nodes w/ one IDE and two SCSI disks each could handle ~38 msgs/sec. w/ replication, ~48 msgs/sec. w/ simulated NVRAM for coordinator log

Review: Yasushi99

- Solutions, continued
 - Testing implications
 - @ ~105 msgs/sec. per node, ~62 nodes could saturate 1Gbps network, w/ ~562 million msgs/day
 - ~6500 msgs/sec. aggregate
 - With replication, this drops to ~5200 msgs/sec. aggregate, and ~450 million msgs/day on ~108 NVRAM nodes or ~137 non-NVRAM nodes

Review: Yasushi99

- Applicability
 - Throws out all previous application work
 - 100% new, untrusted code
 - Can't list 100 IP addresses in DNS for POP services
 - Won't fit into 512 byte UDP packets
 - Can't list 100 IP addresses in DNS for MX services
 - Forced to use proxy front-ends or L4 load-balancing switches to hide the number of servers

Review: Yasushi99

- Applicability
 - Microsoft OSes only ever use the first IP address, then cache forever (until reboot)
 - Forced to use L4 load balancing switches
 - Must be set up in HA/failover mode
 - May have application proxies behind them
 - Some SMTP MTA or resolver implementations are equally dain-bramaged
 - L4 load-balancing switches in front of MXes

Review: Yasushi99

- Applicability
 - Can't get around DNS UDP packet size restrictions with multiple IP addresses per name
 - If connection refused, skip to next name
 - If connection timed-out, go to next IP address for same name
 - At ~2 min. TCP timeout per IP address, 45 IP addresses = 90 minutes to timeout
 - If you have a queue runner fired off every 60 minutes, you ultimately wind up with all memory taken up and no mail flow

Review: Yasushi99

- Applicability
 - Did not use standard benchmarking tools
 - May or may not be valid to create own tools, but needs justification
 - Fundamentally, locking **IS** required
 - Users simply will not accept messages appearing and disappearing and reappearing again
 - Requires serialization which violates most basic principles espoused

Review: Yasushi99

- Applicability
 - Did not test suitable array of MTAs, POP3 daemons, message size and arrival distributions, mailbox sizes, etc...
 - Did not even prove special case, much less general case
 - Anybody can select bad special case and demonstrate superiority
 - To claim general superiority, you must test across a much broader array of variables

Review: Yasushi99

- Applicability
 - IMAP implementation is only a subset — does not include shared folders
 - Perhaps possible in small academic environment
 - Simply not acceptable in large commercial environment
 - SMTP server holds sender open while all writes are completed
 - Violation of RFC 1123, section 5.3.2?
 - All other MTAs accept first, then deliver in background

Review: Yasushi99

- Applicability
 - Each server must implement all protocols
 - Doesn't allow for scaling of each part independently
 - Load discovery protocol is broadcast-based
 - Uses Linux async writes
 - Violation of RFC 1123, section 5.3.3
 - Replication already used to address lower reliability of commodity hardware, OS, and custom application code

Review: Yasushi99

- Applicability
 - Peak sustained rates do not scale linearly
 - Msgs/sec. \rightarrow msgs/min. \rightarrow msgs/hr. \rightarrow msgs/day
 - Msgs/hr. * 10 = \sim msgs/day

Review: Yasushi99

- Applicability
 - Good things
 - Splitting mailboxes at message level
 - Replicate messages to at least two servers
 - Distribute messages across up to four servers
 - Dynamically distribute messages to least loaded servers
 - Calculate “load” based primarily on current and potential disk I/O operations

Skynet Statistics

- POP3 Mail Server
 - 285,000 Accounts
 - 225,000 Mailbox files
 - 600,000 Aliases
 - 6800 Domains
 - 150 GB Total mailbox storage
 - 1 GB Overhead

Skynet Statistics

- POP3 Mailbox Sizes
 - 80,000 Empty
 - 690 KB Average
 - 9282 bytes Median (50th percentile)
 - 1.1 MB 90th percentile
 - 3.35 MB 95th percentile
 - 12 MB 99th percentile
 - 42.1 MB 99.9th percentile

Skynet Statistics

- POP3 Connections
 - 100 peak connections/attempts per second
 - 2300 peak connections/attempts per minute
 - 105,000 peak connections/attempts per hour
 - ??? peak connections per day?
 - 13.14 second typical daily average connection time
 - 300 Max total simultaneous connections allowed

Skynet Statistics

Millisecond response times (14 day sample)

Protocol	Min	Avg.	Max
SMTP	33	672	3600
POP3	28	185	949

Skynet Statistics

- Typical messages per day
 - 450,000 inbound SMTP
 - 450,000 POP3 mailbox deliveries
 - 200,000 webmail/freemail
 - 40,000 business SMTP
 - 400,000 outbound SMTP

Skynet Statistics

- Peak messages per hour
 - 48,000 inbound SMTP
 - 42,000 outbound SMTP

Skynet Statistics

- Typical message volume per day
 - 48 GB inbound
 - 25 GB POP3
 - 18 GB webmail
 - 4.5 GB business
 - 48 GB outbound

Skynet Statistics

- Average message sizes
 - 110 KB inbound
 - 60 KB POP3
 - 100 KB webmail
 - 120 KB business
 - 120 KB outbound

Protocol Implementation Analysis

- POP3
 - Typical implementation
 - Qpopper “Server Mode”
 - Indexed Mailbox
 - Login Frequency Limitation
 - Mailbox Directory
- IMAP Differences & Implications

Analysis: Typical POP3

- User login
- Lock mailbox
- Create temp file
- Copy mailbox to temp file
- Truncate mailbox
- Unlock mailbox
- Operate on temp file
 - New messages may come in to mailbox

Analysis: Typical POP3

- User logout
- If any messages are being retained
 - Re-lock mailbox
 - If mailbox not empty
 - Append new messages to temp file
 - Truncate mailbox
 - Merge retained temp file contents onto mailbox
 - Unlock mailbox
- Delete temp file

Analysis: Qpopper “Server Mode”

- User login
- Lock mailbox
- Operate on mailbox
 - New mail messages wait to be added to mailbox
- User logout

Analysis: Qpopper “Server Mode”

- Are messages being retained?
 - Yes
 - Create temp file
 - Merge retained contents of mailbox onto temp file
 - Move temp file to mailbox
 - No
 - Truncate mailbox
- Unlock mailbox

Analysis: Qpopper “Server Mode”

- Improvements
 - Big “win” if no mail is left on server
 - Virtually all synchronous meta-data operations eliminated
 - No “loss” if mail is left on server
- Issues
 - Still have to scan entire mailbox every time user logs in, even if only to tell them they don’t have any new messages

Analysis: Indexed Mailbox

- User login
- Lock index
- Stat index & mailbox
- If index newer, all questions can be answered from index
 - Only need to lock mailbox if messages are deleted

Analysis: Indexed Mailbox

- If mailbox newer
 - Lock mailbox
 - `lseek()` to last position specified by index, then scan and update index
- Otherwise, like Qpopper “Server Mode”

Analysis: Indexed Mailbox

- Improvements
 - Each message read from mailbox is handled by `lseek ()` and large-size `read ()`
 - Greatly increases use of read-ahead cache
 - Assumes that LDA appends only
 - Assumes that LDA & POP3 server are only methods of reading or writing mailboxes

Analysis: Indexed Mailbox

- Problem
 - Still have to update mailbox if messages are retained and message status changes
- Solution
 - In index, separately store header and body start+offset info
 - Store message status in index
 - Generate message status header info on-the-fly

Analysis: Indexed Mailbox + status

- Results
 - Twice as many read operations
 - Fewer write operations
 - More complex POP3 server
 - Probably a big win for leave-on-server

Analysis: Limiting User Login

- Problem
 - Some clients still login too frequently to check their mail
- Solution
 - Require that at least X minutes elapse before you allow updating of index
 - Tune X for pain threshold of your users

Analysis: Mailbox Directory

- Some POP3 implementations create a directory that comprises the mailbox, and store one message per file
 - Trades smaller number of larger I/O operations for much larger number of smaller I/O operations
 - Avoids mailbox locking issues
 - Creates message locking issues

Analysis: Mailbox Directory

- Problems
 - The I/O operations it creates in trade are all synchronous meta-data operations
 - The most expensive kind
 - The type we most want to eliminate, reduce, or optimize
 - May need to implement directory hashing within mailbox to avoid excessively large directories

Analysis: Mailbox Directory

- Problems
 - Typically has to scan entire directory tree to build mailbox status
 - Must know size of each message
 - Must `stat ()` each file or have file size encoded in file name
 - Must know UIDL value for each message
 - Must open and read each file
 - Can solve these problems by using index
 - Still doesn't eliminate sync. meta-data updates

Analysis: Mailbox Directory

- Claim
 - More NFS-friendly
 - Avoids mailbox locking
 - Mechanism for creating filenames sufficiently unique to virtually eliminate collisions on files
 - Uses “create w/ exclusive ownership” semantics to detect

Analysis: Mailbox Directory

- Reality
 - Christenson97 shows that 7th edition mailbox (mbox) format can also be made NFS-friendly, using same trick
 - Still have issues with sync. meta-data updates
 - Now problem for NFS server vendor?
 - Does not solve locking problems with message changes, moves, or deletions
 - Mailbox locking not really a problem

Implications

- POP3
 - Only one reader process at a time
 - Can safely lock entire mailbox
 - Only one writer process at a time
 - Can safely lock entire mailbox
 - Long-term mail storage is local to user
 - Large sites may not allow “leave on server”
 - Otherwise mitigated by quota or expiration mechanisms

Implications

- IMAP
 - There will be more than one simultaneous reader and/or writer process
 - Cannot lock entire mailbox
 - Must lock at message level or below
 - Long-term mail storage is centralized
 - Only cached locally

Implications

- Solutions
 - Easiest way to deal with message locking is to avoid 7th edition mailbox (mbox) format
 - Use mailbox directory instead, but can use folders
 - One message per file
 - Some typical POP3 enhancements not applicable
 - However, so long as lock mechanism is shared by LDA & IMAP server, can avoid file locking and use database instead

Scaling Growth

- Problem
 - Number of users is increasing
 - Number of messages sent/received per user is increasing
 - Average size of messages is increasing
 - Length of retention of messages increasing
 - Due to centralized storage of mailboxes

Scaling Growth

- Result
 - Disk storage requirements increasing exponentially
 - Number of I/O operations increasing exponentially

Scaling Growth

- Mitigating Factors
 - Disk storage space increasing exponentially
- Complications
 - Disk rotational speed increasing
 - But not increasing very fast
 - Track-to-track latencies improving
 - But not improving very quickly

Scaling Growth

- Result
 - Disk storage requirements still increasing
 - Not quite as bad
 - Number of I/O operations increasing exponentially
 - Our main killer before
 - Will become bigger and bigger bottleneck

Scaling: Future Improvements

- Single Instance Message Store
 - If storing message per file, store message only once per machine and hard link other recipients to same file
 - Reduces I/O bandwidth requirements
 - Doesn't reduce sync. meta-data updates since linking to an existing inode requires just as much directory update work as creating new file

Scaling: Future Improvements

- Multi-session Single Instance Message Store
 - Generate MD5 or SHA-1 hash of message
 - Already in system?
 - Yes
 - Compare binary files, store if different, link otherwise
 - No
 - Store
 - Further reduces disk storage capacity issues
 - **Increases** synchronous meta-data I/O

Scaling: Future Improvements

- Multi-session Single Instance in Bodypart Store
 - Recursively parse MIME message structure, store bodypart-per-file
 - For attachments, insensitive to trivial changes in body
 - Allows you to replace base64 or quoted-printable with binary
 - Allows you to “invisibly” compress data
 - Further reduces disk storage requirements
 - Still doesn’t address issues of sync. meta-data updates

Scaling: Future Improvements

- Use Database for Everything
 - Eliminates sync. meta-data I/O problems
- Problem
 - No database handles BLOBs properly
 - Large scale database reliability problems?

Scaling: Future Improvements

- Use Message “heap”
 - Use INN timecaf/timehash-style files instead of message-per-file
 - New message comes in
 - Append to one of small number of large files
 - Update database index
 - Message is deleted
 - Mark space as available
 - Reclaim empty space at time of reduced load

Scaling: Future Improvements

- Message “heap”, continued
 - Virtually eliminates all sync. meta-data updates
 - Could potentially be combined with previous single-instance-store ideas
 - Probably not worth it
 - Does increase maintenance overhead

Scaling: Future Improvements

- From Yasushi99
 - Break mailboxes into component messages
 - Replicate messages to at least two servers
 - Distribute messages across four or fewer servers
 - Doesn't help address either disk storage or sync. meta-data issues
 - Does address issues of reliability, load-balancing, speed, and perceived quality of service

Scaling: Future Improvements

- Yasushi99, continued
 - Could be combined with INN timecaf/timehash-like message “heap”
 - Could calculate “load” for re-balancing of messages on different criteria
 - Old messages could be migrated to specialized servers with more disk space, perhaps less disk I/O capacity

Best Current Practice

- Per message store server
 - Single instance message store
 - Hard links for multiple recipients of same message
 - Hashed mailbox directories
 - Two base-32 chars per subdir = 1024 max per dir
 - Minimizes path length
 - Message locks in fast and reliable database
 - Berkeley db, not SQL

Best Current Practice

- Per message store server, continued
 - Most important headers and MIME structure in database
 - Most meta-data queries answerable from database
 - User mailbox on single server (cluster)
 - Archive all messages at appl. level, if req'd
 - Clustered servers for HA

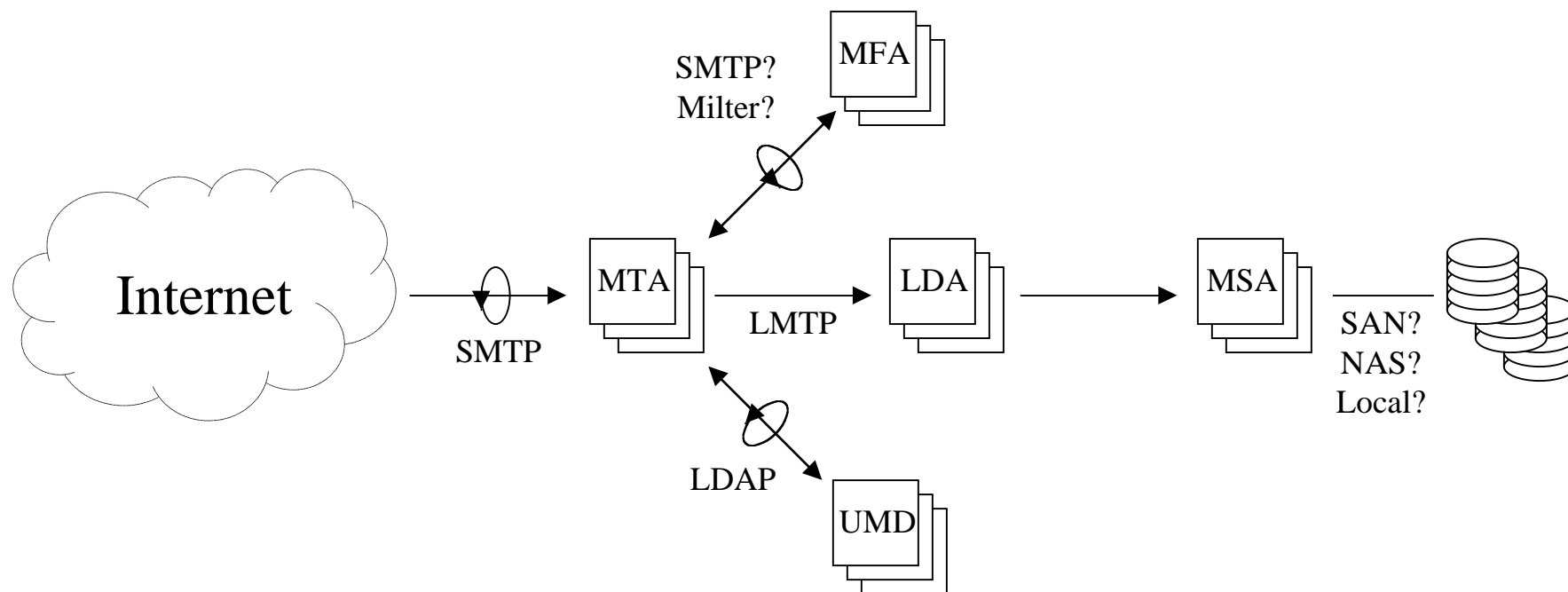
Best Current Practice

- User meta-data database kept outside of message store servers
- Minimize interface protocols
- Use application proxies to distribute traffic across n number of message store servers
- Use Layer 4 load-balancing switches in HA mode to hide number of application proxies

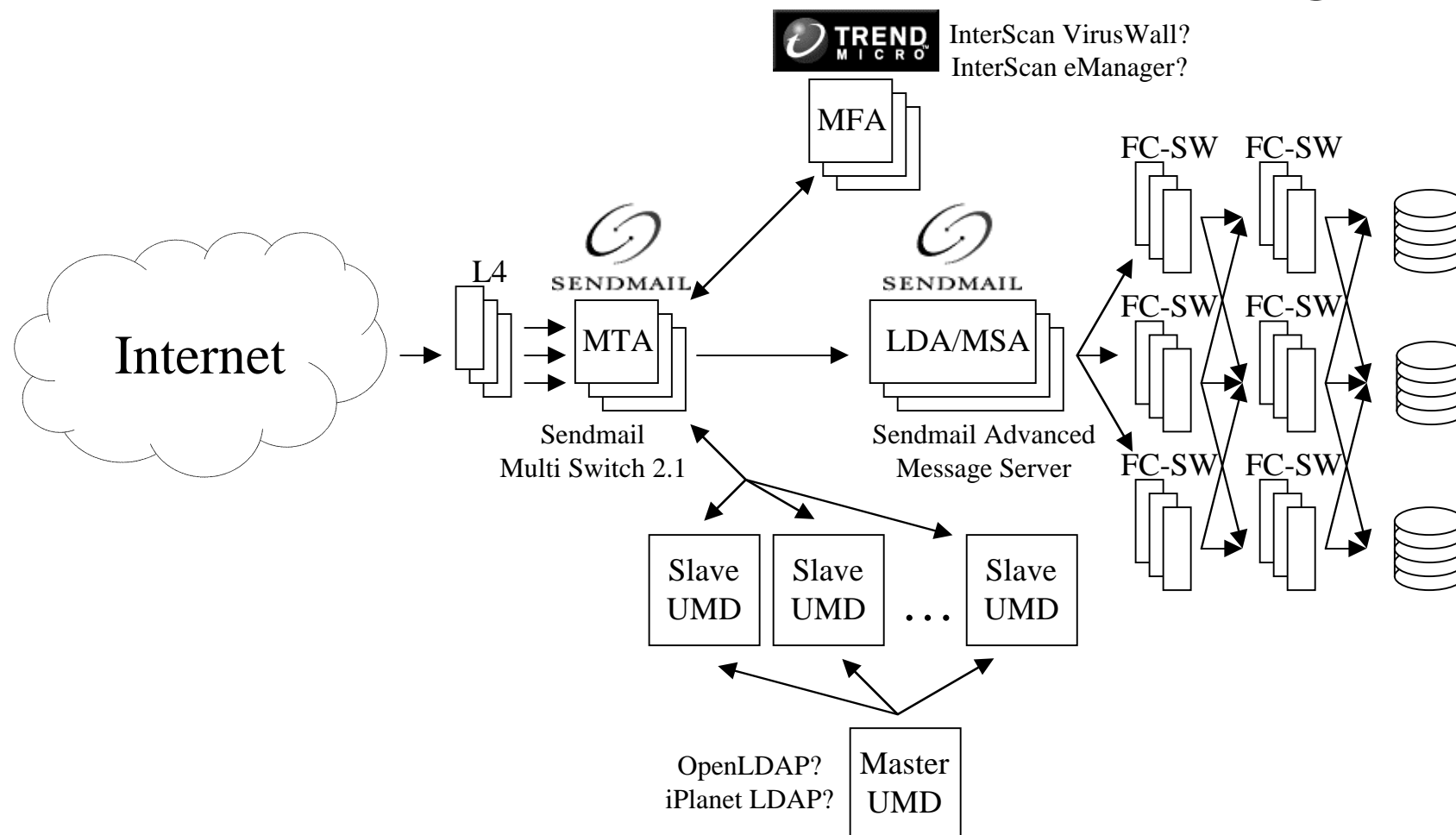
Best Current Practice

- Everything becomes LEGO™ building blocks
- However, scaling is still not quite linear
 - 1 million users = one servers
 - 10 million users ?= ten servers
 - 100 million users != hundred servers

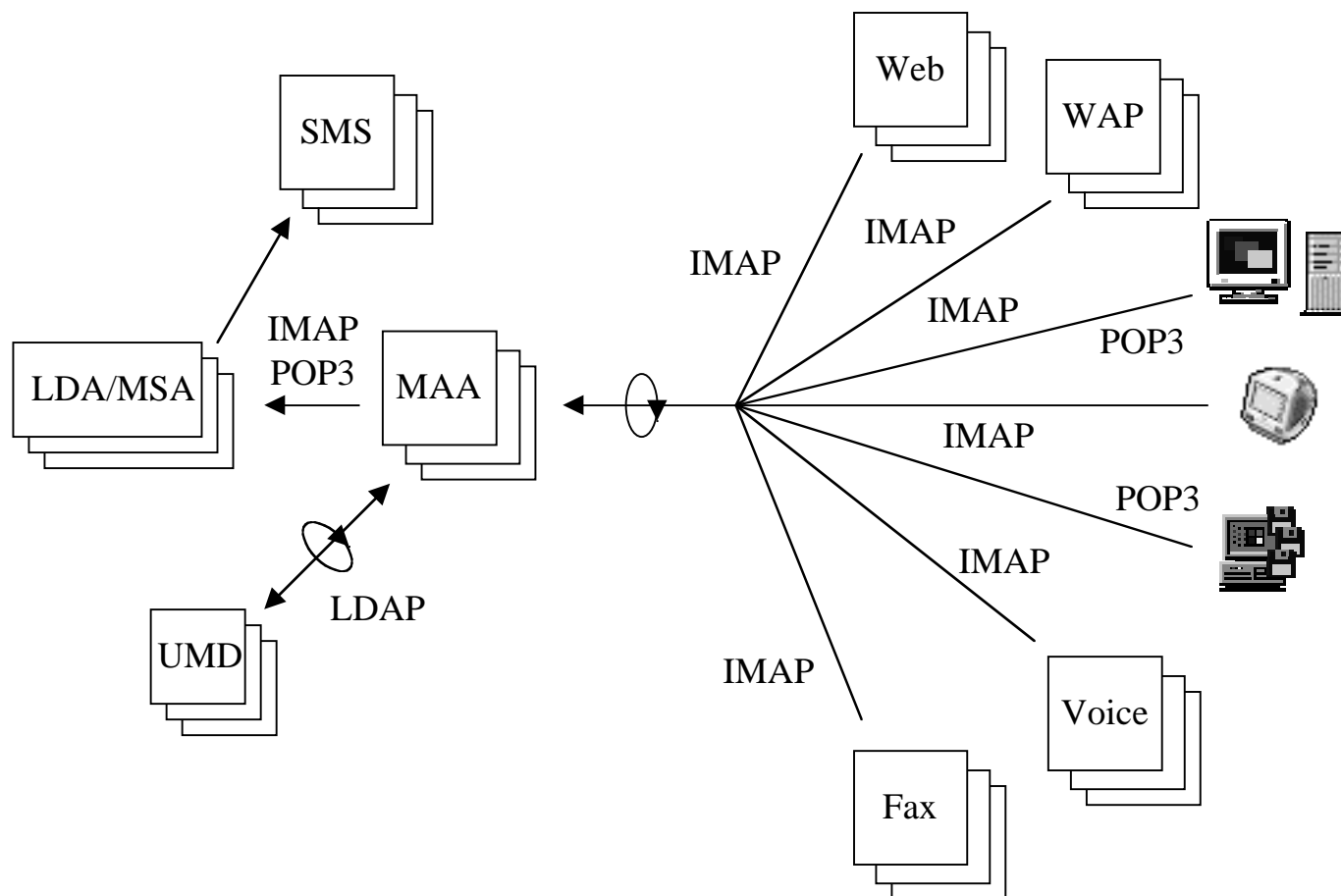
Functional Architecture: Storage



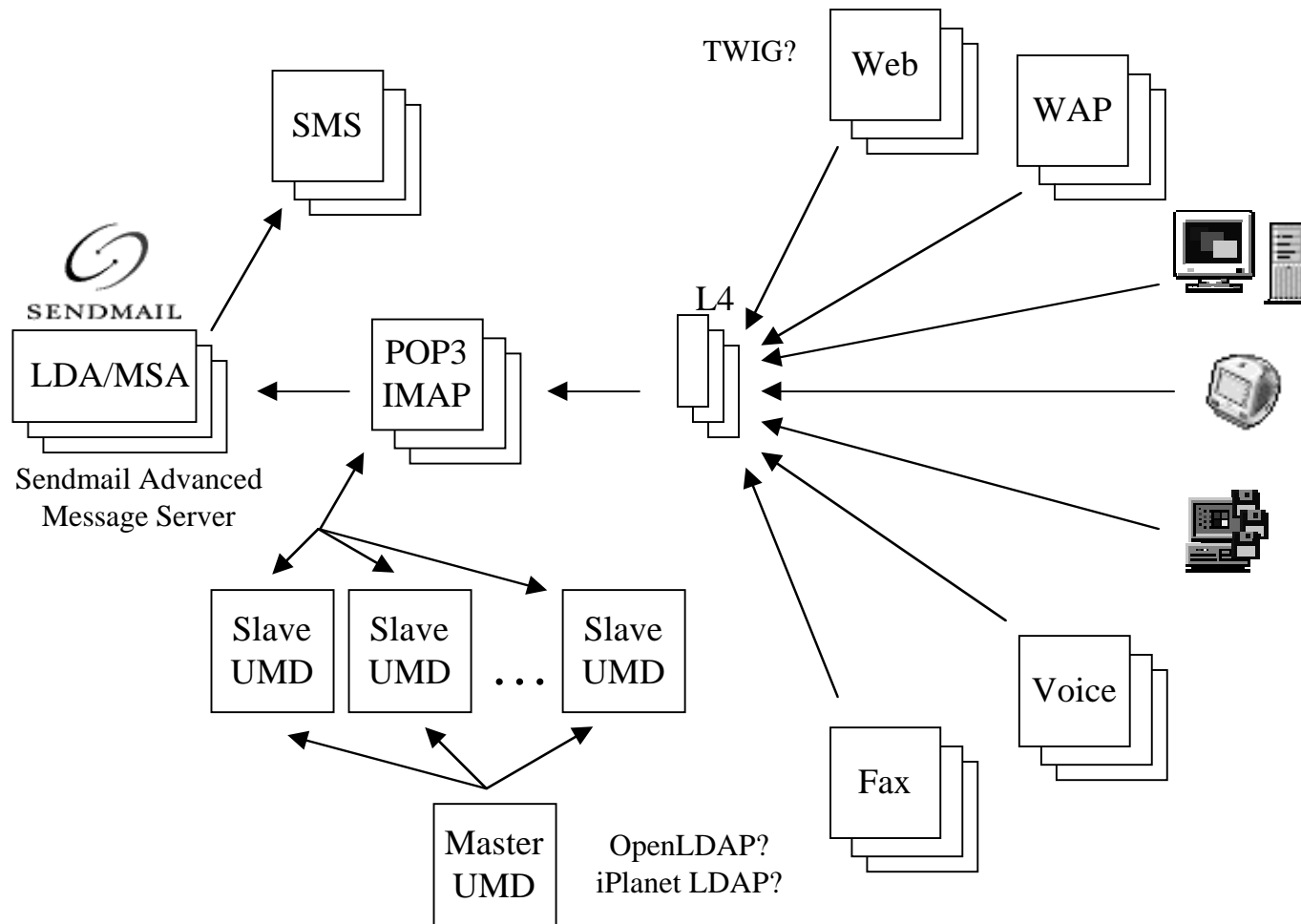
Detailed Architecture: Storage



Functional Architecture: Retrieval



Detailed Architecture: Retrieval



SMTP/POP3 Benchmarking

- Standard Performance Evaluation Committee
 - SPECmail2001
 - <<http://www.spec.org/osg/mail2001/>>
- Russell Coker
 - postal
 - <<http://www.coker.com.au/postal/>>
- Dan Christian, Mozilla Organization
 - mstone
 - <<http://www.mozilla.org/projects/mstone/>>

SMTP/POP3 Benchmarking

- Wietse Venema
 - smtpsink & smtpstone
<<http://www.postfix.org/>>
- Yasushi Saito
 - porctest
<<http://porcupine.cs.washington.edu/porcl/distribution.html>>
- Stalker Software
 - SMTPTest & POP3Test
<<http://www.stalker.com/MailTests/>>

SMTP/POP3 Benchmarking

- dREI C Systems
 - DeJam Analyzing Suite (Java)
<<http://www.dejam.de/>>
- Quest Software
 - Benchmark Factory (NT)
<http://www.benchmarkfactory.com/benchmark_factory/>
- Mindcraft
 - DirectoryMark (LDAP)
<<http://www.mindcraft.com/directorymark/>>

Bibliography

- Beattie, M.
“Design and Implementation of a Linux Mail Cluster”
UKUUG Linux ‘99 Conference, June 1999
<<http://users.ox.ac.uk/~mbeattie/herald-ukuug99.ps>>
- Chalup, S. R., Hogan, C., Kulosa, G., et. al
“Drinking from the Fire(walls) Hose: Another
Approach to Very Large Mailing Lists”
USENIX, LISA XII Proceedings, December 1998
<[http://www.usenix.org/events/lisa98/full_papers/chalup/
chalup_html/chalup.html](http://www.usenix.org/events/lisa98/full_papers/chalup/chalup_html/chalup.html)>

Bibliography

- Christenson, N., Bosserman, T., Beckemeyer, D., et. al
“A Highly Scalable Electronic Mail System Using
Open Systems”
USENIX, USENIX Symposium on Internet
Technologies and Systems, December 1997
<http://www.jetcafe.org/~npc/doc/mail_arch.html>
- Christenson, N.
“Performance Tuning Your *sendmail* System”
O'Reilly Open Source Conference, August 1999
<http://www.jetcafe.org/~npc/doc/performance_tuning.pdf>

Bibliography

- Golanski, Y.
“The Exim Mail Transfer Agent in a Large Scale Deployment”
April 2000
<<http://www.kierun.org/academic/lsm.pdf.gz>>
- Grubb, M.
“How to Get There From Here: Scaling the Enterprise-Wide Mail Infrastructure”
USENIX, LISA X Proceedings, October 1996
<<http://www.oit.duke.edu/~mg/email/email.paper.html>>

Bibliography

- Horman, S.
“High Capacity Email”
Conference of Australian Linux Users, July 1999
<http://www.us.vergenet.net/linux/mail_farm/html/>
- Knowles, B.
“Sendmail Performance Tuning for Large Systems”
SANE ‘98, November 1998
<<http://www.shub-internet.org/brad/papers/sendmail-tuning/>>

Bibliography

- Kolstad, R.
“Tuning Sendmail for Large Mailing Lists”
USENIX, LISA XI Proceedings, October 1997
<http://www.usenix.org/publications/library/proceedings/lisa97/full_papers/21.kolstad/21_html/main.html>
- Stevens, L.
“Serving Internet Email for 60,000”
Internet Expo, February 1997
<<http://staff.washington.edu/lrs/ew/>>

Bibliography

- Venema, W.
“Postfix”
`<ftp://ftp.porcupine.org/pub/security/postfix-sane-1998.ps.gz>`
- Yasushi, S., Bershad, B., and Levy, H.
“Manageability, availability and performance in
Porcupine: a highly scalable, cluster-based mail
service”
17th ACM Symposium on Operating System
Principles (SOSP '99), December 1999
`<http://porcupine.cs.washington.edu/porcl/sosp99/index.html>`

Questions?

- Slides will be made available
 - Via USENIX/SAGE web site
 - Or via my “papers” sub-page
`<http://www.shub-internet.org/brad/papers/>`
 - At very least, will be linked from my “papers” sub-page