Dovecot IMAP Server

http://www.dovecot.org/

Date: July, 2009

Rackspace Email

- Dovecot is used to serve IMAP for over a million paid mailboxes (MS Exchange also available)
- Users assigned to specific backend servers
  - With proprietary replication software
- Dovecot IMAP/POP3 proxies in front
  - Also for Exchange IMAP/POP3 users
- Amazon S3 for (encrypted) backups
- More about clustering later..

The Talk

- Dovecot features
- IMAP & Dovecot performance
- dbox mailbox format
- Clustering

Dovecot

Pictures from Wikipedia, by *Cyril Thomas and* Carcharoth

History

- Dovecot design was started around June 2002
- First release was July 2002
- Late 2003 a redesign started
- v1.0.0 released April 13th 2007
- v1.1.0 released June 21st 2008
- v1.2.0 released July 1st 2009
- v2.0 betas hopefully this year

Features

- Often has better performance than competition.
  - Optimized for minimizing disk I/O (index/cache files)
  - Hosting my own mails on 10 years old Sparc helps
- Highly configurable for different environments
  - Standard mbox and Maildir with **transparent** indexing (external mailbox modifications are ok)
  - dbox: Dovecot's high-performance mailbox format
  - Many different ways of clustering
  - Extremely flexible authentication
    - Postfix and Exim support Dovecot for SMTP AUTH

Features

- Admin-friendly / self-healing
  - All errors are logged
  - Understandable error messages
    - Improved constantly (to reduce my email load)
  - Detected (index) corruption gets fixed automatically

- file_dotlock_create(/home/timo/Maildir/dovecot-uidlist) failed: Permission denied (euid=1000(timo) egid=1000(timo) missing +x perm: /home/timo)
- chown(/home/timo/Maildir/.box, -1, 0(root)) failed: Operation not permitted (egid=1000(timo), group based on /home/timo/Maildir)

v1.2 New Features

- Virtual mailboxes (search views)
  - ”All unread emails in all mailboxes”
  - All messages in all mailboxes (except Trash)
    - Virtual POP3 INBOX
    - For searching messages from all mailboxes
    - gmail-like conversation views
- Users can share mailboxes to each others
  - IMAP ACL commands
- Modification sequences (CONDSTORE)
  - Custom code wanting quick sync? (e.g. backups)

Authentication

- Password and user database separation
  - Passdb for verifying user's password
  - Userdb for looking up how to access mailbox
- Support for almost everything: SQL, LDAP, PAM, checkpassword scripts, etc.
  - Everything is configurable (e.g. full SQL queries)
  - Supports multiple dbs (e.g. system + virtual users)
- Auth mechanisms: PLAIN, CRAM-MD5, DIGEST-MD5, Kerberos, OTP, etc.
- Password schemes: Plaintext, CRYPT, MD5, SHA1, SHA256, SSHA, SSHA256, etc.

Authentication Cache

- Passdb and userdb lookups can be cached
- Password changes are automatically detected:
  If auth is unsuccessful, and previous auth was
  a) successful: do uncached passdb lookup
  b) usuccessful: fail login
- Negative caching can be disabled
  - User doesn't exist caching
  - Password failures (v1.2+)
- Avoids a need for imapproxy with webmails?

IMAP Protocol

- Base protocol is complex – difficult to implement it correctly (both client & server)
- Flexible – many different ways to implement a client (online & offline clients)
- Extensible – there are a lot of extensions
  - Clients rarely support more than some basic extensions, such as IDLE.
  - Thunderbird v3 adds support for several new extensions, such as CONDSTORE.

ImapTest IMAP Server Tester

- Written originally for Dovecot stress testing
  - Found a lot of crashes, hangs and mailbox corruption on other IMAP servers as well
- Tests IMAP server compliance with scripted tests and dynamic random stress testing.
- Dovecot is currently the only IMAP server that fully passes all of ImapTest tests.
  - Panda IMAP is practically there too
- Most other servers fail in many different ways.
- http://imapwiki.org/ImapTest

Offline IMAP Clients

- Typically download newly seen messages' bodies once and cache them locally
- Often can be configured to download immediately vs. download when reading
- Some use server side searches (Thunderbird) and some don't (Outlook – if some messages haven't been downloaded, those aren't searched)
- Usually also fetch messages' metadata once (headers, received date)
- Server-side caching may help, but not that much
  - It's extra disk I/O -> more likely just hurts

Online IMAP Clients

- Webmails often keep asking for the same information over and over and over again
- Pine and some webmails cache what they've already seen, but not permanently
- Mutt (without local cache) and some others fetch all messages' metadata every time when opening a mailbox
- Caching is very useful, but different clients want different metadata

IMAP Server Performance

- Difficult to benchmark
- Depends a lot on clients: Whether clients use a local cache makes a huge difference.
  - Online vs. offline clients
- What data to index/cache?
- SPECmail2009 adds support for IMAP
  - Emulates different IMAP clients. Client amounts are configurable.
  - The only benchmark giving realistic results.

Dovecot Cache File

- **dovecot.index.cache** files
- The main reason for Dovecot's good performance
- Dynamic: caches only what clients want.
  - Specific message headers (From:, Subject:, etc),
  - MIME structure information,
  - Sent / received date, etc.
- Caching decisions for each field: "no", "temporary", "permanent"
- Unused fields dropped after a month.
- Cached data never changes (IMAP guarantees)
- Cache file gets "compressed" once in a while
- Often about 10-20% of mailbox size

Dovecot Index Files

- **dovecot.index** contains messages' metadata
  - IMAP Unique ID number (**UID**) identifies messages
  - Flags (\Seen, \Answered, keywords, etc.)
  - Extension data: mbox file offsets, cache file offsets, modseq number (v1.2 CONDSTORE), etc.
- Lazily created/updated since v1.1
  - **dovecot.index.log** has all the latest changes. **dovecot.index** is updated after 8 kB of new data has been written to the .log

Dovecot Index Files

- **dovecot.index.log** is a mailbox transaction log
  - Somewhat similar to databases' transaction logs or filesystem journals.
  - Contains all changes to be done to **dovecot.index**.
- **dovecot.index** is read to memory once and then updated from **dovecot.index.log**
  - Very efficient with NFS / clustered filesystems!
  - Very efficient to find out what changes another session had done!

Plugins

- Dovecot plugins can hook into almost anything and modify Dovecot's behavior. Some existing features implemented as plugins:
  - Access Control Lists
  - Quota
  - Full text search indexes
  - Reading compressed mbox/maildir files
- Can add new IMAP commands
- Implement new mail storage backends (virtual, SQL, IMAP proxying)

Mailbox Formats

- mbox
  - One mailbox = one file
    - Slow to delete old messages
- Maildir
  - One file = one message
    - Fast to delete messages
    - Slow(er) to read through all messages
    - File read order affects performance, even 2x or more!
- Single-dbox and multi-dbox
  - Dovecot's extensible and high-peformance mailbox formats

Single-dbox

- Available in Dovecot v1.1 and later
- Main advantage over Maildir: filenames don't change.
- Directory layout looks like:
  - **mailboxes/INBOX/dbox-Mails/**
    - **dbox.index** – dbox index (removed in v2.0)
    - **dovecot.index*** - Dovecot's index files
    - **u.123** - Message data for IMAP UID 123
    - **u.125** - Message data for IMAP UID 125
  - **mailboxes/Trash/dbox-Mails/**
  - **mailboxes/Trash/temp/dbox-Mails/**

Single-dbox

- Primary metadata storage is Dovecot's index files
  - Metadata backups written about once a day to dbox files -> losing indexes won't lose all flags
- Automatically fixes/rebuilds broken/lost indexes
- Future: Dovecot v2.0 no longer writes flags to dbox files. It creates separate index file backups instead.

dbox File Format

- File header
  - Message header size
  - File creation data
- Message header (extensible)
  - Message size
- Message body
- Message metadata (extensible)
  - Message's globally Unique ID (GUID)
  - Receive and save date/time
  - Message's "virtual size"
  - etc.
- [multi-dbox: Next message…]

Single-dbox: Maildir Migration

- Superfast migration from Maildir:
  - Renames Maildir/cur/ to dbox-Mails/
  - Moves other useful Maildir files too
- New mails will be saved using native dbox format
- Old mails get converted to dbox later when user changes old mails' flags.
  - Mails might stay as Maildir for a long time

Single-dbox: Alternative Storage

- Users rarely access their old mails
- Lower performance storage is cheaper

  -> Move old mails to low performance storage
- dbox supports ”alternative path” setting: If a dbox file isn’t found from primary path, it’s looked up from alternative path.
  - mail_location = dbox:~/dbox:ALT=/slow/%u/dbox
- Future: Support for cloud storage (like CloudFiles/S3)?

Multi-dbox

- Available in upcoming Dovecot v2.0
- Multiple messages in a single file
- Multiple files in a single mailbox
  - Files are about 2 MB (configurable)
  - Can be rotated every n days (for incremental backups)
  - Larger files -> less fragmentation, but deletion slower
  - Delayed ioniced nightly deletions
- Tries very hard to preserve as much data as possible in case of (filesystem) corruption.
- Crash or power loss can’t corrupt or lose data

dbox Future

- Single instance attachment storage
- Abstract out filesystem access and implement
  - Regular POSIX I/O
  - Async I/O
  - Cloud storage I/O
- Make Dovecot do more parallel processing to get good performance for (high latency) cloud storage and to get full advantage of async I/O.

Dovecot Clustering

- Two different ways to do it:
- Globally shared filesystem
  - Many IMAP servers, each able to handle any user
  - NFS, cluster filesystems
- Sharding
  - Each user's data mirrored in 2-3 servers
  - IMAP proxy forwards users to correct server(s)
  - DRBD, proprietary clustering software/hardware

Clustering: Two Types of Data

- Message data
  - Existing messages (files) don’t change
  - Users typically read messages once -> message is read from disk only once (or few times)
  - Latency hurts, but not badly (in future even less)
- Index data
  - Constant lookups: ”Has mailbox changed?”
  - Latency is very bad for performance
  - Existing files change constantly -> caching trouble!
- Different storages for messages/index?

Clustering: NFS

- NFS server is often single point of failure
  - Performance problems affect everyone. Might be difficult to diagnose/fix.
  - Example: NFS locking broke -> restarted -> Dovecot became unusably slow
- Caching problems, especially with index files
  - mail_nfs_* settings try to solve these
- Index files on local disk helps performance
- http://wiki.dovecot.org/NFS

Clustering: NFS

- Sticky servers for users = only one server modifies a user's mailbox
  - IMAP proxy looks up destination server from db
  - Avoids caching problems
  - If mail delivery updates indexes, must be done by the same server as IMAP.
    - Each server receives mails with SMTP/LMTP
  - Storing indexes on local disks helps performance
    - If server goes down, reindexing may be slow
    - DRBD hybrid?

Clustering: Cluster FS

- Dovecot known to work with GFS, OCFS2, ..
- Less caching problems than with NFS
  - Performance still better when user accesses only single server (better caching, less lock waits)
- Performance?
  - Many small files are bad?

Clustering: Sharding

- Typically in active/passive server pairs:
- Dedicated active and passive servers
  - Wastes servers
- Crossed pairs
  - Each server is active for one set of users and passive for another set of users
  - Server failure doubles the passive's load
- Dovecot IMAP/POP3 proxy cluster in front

Clustering: Sharding

- Distribute individual users (not entire domains) to different servers
  - Reduces load spikes
- Use statistics to automatically distribute heavy users to different servers
  - v1.2 can export very detailed statistics via plugin
  - v2.0's upcoming dsync utility

Clustering: DRBD

- Filesystem corruption gets replicated
- Synchronous replication
  - No mail loss on failures
  - Too slow for cross-datacenter(?)
- Asynchronous replication
  - Some data loss on failure
- 3 servers: Sync replication for in-datacenter and async for cross-datacenter backup?

Clustering Future: The Cloud

- Save message data in cheap cloud storage
  - Typically simple APIs to access files
    - dbox designed for this
  - Typically higher latency
    - Dovecot needs to do more work while waiting
- Index data kept primarily in memory
  - Must be very low latency -> direct communication between servers that access the same mailbox
  - Permanent (backup) storage may still be in cloud
- Result: multi-master replication

Dovecot v2.0

- Some new features already implemented:
  - Redesigned master process
    - Easy to add external services, e.g. ManageSieve
  - Redesigned configuration
    - Local/remote IP/mask -specific configuration
      - SSL certs
    - Allow changing config data source (e.g. SQL?)
  - LMTP server
  - dsync: Realiably and efficiently sync two mailboxes (e.g. via SSH)

Dovecot v2.0

- Features not yet implemented, but hopefully will be by the end of this year:
  - Index file improvements
    - No locking (with atomic appends)
    - Small checksums all around for detecting corruption
    - In general make the code simpler and more robust
  - Multi-master replication
    - dbox cloud storage (for some existing cloud API(s)?)
    - Index sharing/replication between servers

Questions?