Stalwart JMAP

> Welche Entwicklungen gibt es in der Free and Open Source Gemeinschaft abgesehen von Postfix und Dovecot. Nach einem kurzem Rundumschlag sehen wir uns JMAP und Stalwart etwas genauer an.

42 Jahre SMTP

Wie sieht die aktuelle Marktverteilung bei den MTAs aus?

https://www.securityspace.com/s_survey/data/man.202403/mxsurvey.html

- Exim 60%
    - Exim nur noch kleine Änderungen
    - viele Bugfixes
    - kann viele Dinge, die wir aber im MTA eh nicht mehr machen
- Qmail 0,0x%
    - als s/qmail mit neuem Maintainer
    - viele Ansätze nicht mehr zeitgemäß
- Postfix 35%
    - Ist wohl "Feature Complete"

---

30 Jahre IMAP4

Wie sieht die aktuelle Marktverteilung bei den MDAs aus?

https://openemailsurvey.org/2020.html

- Dovecot / Courier / Cyrus
    - Dovecot 76%
        - mit Cluster-Architektur - Ausrichtung auf sehr große Infrastrukturen
        - eher wenig Entwicklungsschritte in Richtung neue Standards SMTPUTF8/JMAP
    - Courier 8%
        - seit Jahren sehr wenig Aktivität
    - Cyrus 0,6%
        - wenig Weiterentwicklung
        - aber JMAP Patches von Fastmail

Was gibt es neues

Node.js Haraka / ZoneMTA

- Pro: sehr flexibel, erweiterbar, verteilte queues etc.
- Cons: Node.js

Mox

- Go - all in one SMTP/IMAP Server
- DMARC/DANE/MTA-STS etc
- Webinterface
- Autoconfig
- Vielleicht etwas für Single Nodes

Apache James

- Java SMTP/IMAP/JMAP Server
- Sehr flexibel durch interne Pipelines und APIs

Stalwart

Buzzword Bingo

- written in Rust
- SMTP/LMTP/IMAP/JMAP
- IMAP4rev2+1 compliant
- Built-in DMARC, DKIM, SPF and ARC verify / sign support incl. Reporting
- Transport security through DANE and MTA-STS incl. Reporting
- Configurable rules mit Expressions und Sieve Scripting
- Datenbanken:
  - RocksDB
  - FoundationDB
  - PostgreSQL
  - MySQL / MariaDB
  - SQLite
  - S3/MinIO
  - ElasticSearch
  - Redis
- DBs Nutzbar für Laufzeitdaten, Mail-Queues, Mail-Storage, Full-Text-Search
- ACME
- Webinterface Administration / Self-Service / API
- Distributed SMTP Multi-Queues
- Fail2ban like Auth tracking

- Advanced Spam and Phishing Filtering (Rules, Bayes, Reputation)
- OAuth Authentication
- Built-in Ratelimits (SMTP, Submission, IMAP ...)
- S/Mime / OpenPGP Support zur Mailbox-Verschlüsselung (mail-crypt)
- Milter Support (Rspamd) + Pipes
- Header / Body Rewrite

JMAP

- Websockets Support
- Push Notifications
- Sieve over JMAP

JMAP / IMAP 5 explained

Clients status?

- Desktop: Swift Mail (macOS)
- Mobile: Ltt.rs, Mailtemi, tmail-flutter
- Web: Cypht, Group-Office

Servers?

- Apache James
- Cyrus Imap
- Stalwart JMAP

Requested

Alps Webmail	Claws Mail
Evolution	Geary
Horde IMP Webmail	K-9 Mail
Mailu	Nextcloud
RainLoop Webmail	Thunderbird

JMAP

- silly clients with clever server
- server side mail disassemble
- multi commands
- JMAP RFC
  - JMAP Protocol
  - JMAP Mail
  - JMAP Websockets
- In development
  - calendar
  - contacts
  - news
  - Tasks
  - files?

JMAP vs IMAP

Am Beispiel vom Anzeigen der letzten Zehn E-Mails der INBOX

IMAP Login

Als erstes Login:

Request:

```
a1 LOGIN user password
```

Response:

```
a1 OK [CAPABILITY IMAP4rev2 IMAP4rev1 ENABLE SASL-IR LITERAL+ ID UTF8=ACCEPT IDLE NAMESPACE CHILDREN MULTI
```

IMAP Mailbox List

Auflisten der Mailboxen
Request:

```
a2 list "" "."
```

Response:

```
* LIST () "/" "Deleted Items"
* LIST () "/" "Drafts"
* LIST () "/" "INBOX"
* LIST () "/" "Junk Mail"
* LIST () "/" "Sent Items"
a2 OK LIST completed
```

IMAP Select

Wechseln in die INBOX

Request:

```
a3 select INBOX
```

Response:

```
* 4 EXISTS
* FLAGS (\Answered \Flagged \Deleted \Seen \Draft)
* 0 RECENT
* OK [PERMANENTFLAGS (\Deleted \Seen \Answered \Flagged \Draft \*)] All allowed
* OK [UIDVALIDITY 674887985] UIDs valid
* OK [UIDNEXT 1] Next predicted UID
* OK [MAILBOXID (a)] Unique Mailbox ID
a3 OK [READ-WRITE] SELECT completed
```

IMAP Fetch

Nun das Eigentliche herunterladen der Mails

Request:

```
a4 FETCH 1:4 (BODY[Header])
* 1 FETCH (BODY[HEADER] {1577}
```

Response:

```
* 1 FETCH (BODY[HEADER] {1577}
ate: Sat, 04 May 2024 16:18:23 +0200
To: manu@stalwart.zurmuehl.org
From: m.zurmuehl@heinlein-support.de
Subject: test Sat, 04 May 2024 16:18:23 +0200
Message-Id: <20240504161823.083663@stalwart.zurmuehl.org>

)
* 2 FETCH (BODY[HEADER] {1597}
...
```

Und hier nun mit JMAP

JMAP Mailbox/get

Hier holen wir uns die Liste an Mailboxen ab, wir gehen davon aus, dass wir nichts im Cache haben

Request:

```
[[ "Mailbox/get", {
    "accountId": "user1",
    "ids": null
}, "0" ]]
```

Response:

```
[[ "Mailbox/get", {
    "accountId": "user1",
    "state": "123",
    "list": [{
        "id": "mailbox1",
        "name": "Inbox",
        "parentId": null,
        "role": "inbox",
        "sortOrder": 0,
        "totalEmails": 1424,
        "unreadEmails": 3,
        "totalThreads": 1213,
        "unreadThreads": 2,
        "myRights": {
            "mayAddItems": true,
            ...
        },
        "isSubscribed": true
    }, {
        "id": "mailbox2",
        "name": "Sent",
        ...
        "myRights": {
            "mayAddItems": true,
            ...
        },
        "isSubscribed": true
    } ],
  "notFound": []
}, "0" ]]
```

JMAP Email/get

Jetzt können wir die letzten Zehn E-Mails aus der INBOX abholen, dies passiert in nur einer Abfrage

Request:

```
[
    // Abfrage der ersten 10 IDs aus der Mailbox
    [ "Email/query", {
        "accountId": "user1",
        "filter": {
            "inMailbox": "mailbox1"
        },
        "sort": [
            { "property": "receivedAt", "isAscending": false }
        ],
        "position": 0,
        "collapseThreads": true,
        "limit": 10,
        "calculateTotal": true
    }, "0" ],

    // Abfrage der threadIds
    [ "Email/get", {
        "accountId": "user1",
        "#ids": {
            "name": "Email/query",
            "path": "/ids",
            "resultOf": "0"
        },
        "properties": [ "threadId" ]
    }, "1" ],

    // Jetzt die E-Mail IDs aus den Threads
    [ "Thread/get", {
        "accountId": "user1",
        "#ids": {
            "name": "Email/get",
            "path": "/list/*/threadId",
            "resultOf": "1"
        }
    }, "2" ],

    // Und nun die Daten der E-Mails selbst
    [ "Email/get", {
        "accountId": "user1",
        "#ids": {
            "name": "Thread/get",
            "path": "/list/*/emailIds",
            "resultOf": "2"
        },
        "properties": [ ... ]
    }, "3" ]
]
```

Response:

```
[
    [ "Email/query", {
        "accountId": "user1",
        "ids": [
            "fm1u314",
            "fm1u312",
                                                ...
        ],
        "queryState": "123:0",
        "position": 0,
        "canCalculateChanges": true,
        "total": 10
    }, "0" ],
    [ "Email/get", {
        "accountId": "user1",
        "list": [
            { "id": "fm1u314", "threadId": "4f512aafed75e7fb" },
            { "id": "fm1u312", "threadId": "fed75e7fb4f512aa" },
                                                ...
        ],
        "state": "123",
        "notFound": []
    }, "1" ],
    [ "Thread/get", {
        "accountId": "user1",
        "list": [
            { "id": "4f512aafed75e7fb", "emailIds": [ ... ] },
            { "id": "fed75e7fb4f512aa", "emailIds": [ ... ] },
                                                ...
        ],
        "state": "123"
    }, "2" ],
    [ "Email/get", {
        "accountId": "user1",
        "list": [{
            "id": "fm1u314",
            "threadId": "4f512aafed75e7fb",
            "mailboxIds": {
                "mailbox1": true
            },
            "keywords": {
                "$seen": true
            },
            "hasAttachment": false,
            "from": [
                { "name": "Joe Bloggs", "email": "joebloggs@example.com" }
            ],
            "to": [
                { "name": "Jane Doe", "email": "janedoe@example.com" }
            ],
            "subject": "Camping trip",
            "receivedAt": "2014-07-24T15:04:51Z",
            "preview": "Hey Joe. Fancy a trip out west next week? I hea…"
        }, ... ],
        "state": "123",
```

```
        "notFound": []
    }, "3" ]
]
```

Sources
https://jmap.io/client.html

Stalwart Feature Details

- All-in-one Rust Binary
- Konfigurations-Philosophie
  - Toml Style Konfiguration
  - Konfiguration ist aufgeteilt in Datei und DB Daten
  - Stalwart verschiebt bei der API Nutzung auch manchmal Optionen in die Datenbank
  - (noch?) nicht alles über das Webinterface einstellbar
  - Optionen können auch Ausdrücke beinhalten

```
[queue.outbound]
next-hop = [ { if = "is_local_domain('', rcpt_domain)", then = "'local'" }, { else = false } ]
```

- mehrere Regeln bei bestimmten Optionen über hochzählbare Config-Optionen

```
session.rcpt.relay.0.if = !is_empty(authenticated_as)
session.rcpt.relay.0.then = true
session.rcpt.relay.1.if = rcpt_domain == 'mail.lxc'
session.rcpt.relay.1.then = true
session.rcpt.relay.2.else = false
```

- Alternativ können oft Sieve Skripte verwendet werden

```
[sieve.trusted.scripts]
ehlo = '''
    require ["variables", "extlists", "reject"];

    if string :list "${env.helo_domain}" "list/blocked-domains" {
        reject "551 5.1.1 Your domain '${env.helo_domain}' has been blocklisted.";
    }
'''
```

Stalwart

Sieve Scripting

- klassische Nutzung beim LMTP
- SMTP Protocol Stages
  - vergleichbar Postfix Restrictions

- Greylisting in Sieve
- Spam Filter in Sieve
- Sieve Extensions für benötigte erweiterte Funktionalität
  - SQL Abfragen
  - Listen abfragen (vgl. Postfix Lookup Tables)
  - Bayes Filter

Webinterface

- wie Rspamd ist das Webinterface nur ein Client für die API
- CLI Tool für die API kann mehr
- aber alles wichtige für den Tagesbetrieb einstellbar
- User Self-Service
  - Passwörter ändern (nicht bei SQL/LDAP)
  - für Postfachverschlüsselung: S/Mime/PGP Zertifikat managen

---

MTA detailed

- eingehend Sieve und expression Rules

```
[session.connect]
script = "'connect_filter'"

[sieve.trusted.scripts]
connect_filter = '''
require ["variables", "reject"];

if string "${env.remote_ip}" "192.0.2.88" {
    reject "Your IP '${env.remote_ip}' is not welcomed here.";
}
'''
```

- bei der End-Of-Data Stage Milter und interner Spam-Filter (Sieve)
- Header/Body können am End-Of-Data über Sieve manipuliert werden
- Ratelimits einzelne oder mehrere Werte
- Multiple Queues definierbar
  - Timings, Routing und DSN-Config einzeln einstellbar
  - Queues werden in einer der DBs zwischengespeichert
- ausgehendes Routing per Remotes und Routing Definitionen

MDA detailed

- Shared Folders
- Quota
- Sieve (nicht aber IMAP-Sieve)

Stalwart at scale

- Stalwart schreibt alle Warteschlangen-Mails, Laufzeitdaten, Sieve Skripte und Mailbox-Daten (gespeicherte Mails) in eine Datenbank
- FoundationDB/SQL/S3 können High-Available betrieben werden
- Configuration für unsere üblichen externern MX und Mailout und interne Hubs/Router sind möglich
- dabei können Mail-Queues zwischen den (gleichartigen) Systemen sogar geteilt werden
- Es gibt keine Entsprechung für den Dovecot Director/Proxy
- Aber, da viele Configs noch unklar sind und das FoundationDB Build gerade kaputt ist, haben wir noch keine ganze Infrastruktur testweise aufgebaut!

What we have missed

- Liste aller Optionen
- detailiertere technische Dokumentation
- klarere oder überhaupt Fehlermeldungen
- flexibleres Logging
- Verify / SMTP Look-Ahead
- Keine Queue-ID im EOD Return Code
- TLS CA Optionen zur Nutzung der System CAs

Zusammenfassung

- Stalwart ist eine tolle Software mit schicken neuen Ansätzen und guten Konzepten
- Stalwart ist zur Zeit vor allem ein All-in-One System
- Wir müssen aber noch einiges über die Optionen und die API lernen
- Vieles fühlt sich nach Blackbox an
- Einige Dinge haben noch Bugs und bestimmte Einträge in die Datenbank können Stalwart ganz lahmlegen ohne das eine Fehlermeldung geloggt wird

---

Im Vergleich zu Postfix

- Stalwart könnte Postfix ersetzen und um einige Dinge wie MTA-STS und TLSRCPT Reporting erweitern.
- Auch ARC/DMARC/DKIM etc. wären dabei, aber das machen wir ja eh im Rspamd.
- Gerade dort, wo wir beim Postfix an Grenzen stoßen, scheinen wir mit Stalwart (oder anderen) auch (noch) nicht besser dran zu sein.

Im Vergleich zu Rspamd

- Die Anti-Spam Komponente ist vom Regelwerk durch Sieve sehr flexibel.
- Das mitgelieferte Regelwerk erinnert stark an den die Rspamd-Rules.
- Bayes und die IP-Reputation sind nett.

- Aber der Umfang der Funktionen und die Möglichkeiten des Rspamd sind viel größer

---

Im Vergleich zu Dovecot:

- Stalwart hat derzeit als IMAP Server weniger Funktionen als Dovecot
- bietet sich aber als Backend Server mit seinen vielen Datenbank-Möglichkeiten geradezu an.
- Für größere (kritische) Umgebungen fehlt aber flexibleres und strukturierteres Logging.
- Zum Dovecot Direktor/Proxy ist Stalwart keine Alternative. Das brauchen wir vielleicht aber auch nicht

---

- Also: Wir beobachten, testen weiter und freuen uns auf eine Stalwart 1.0 ☺

Fragen ???

Bonus

Structured E-Mail

- Die Idee strukturierte Daten (Zusatzinformationen) versteckt in E-Mails zu transportieren
  - vgl. Flugbuchungsdaten in Gmail / Schema.org
  - bisher in E-Mail nicht spezifiziert
- Die Strukturierten Daten können im Client automatisch verarbeitet werden
  - Buchungen
  - Bestellungen und Rechnungen
  - Abwesenheiten (vielleicht auch vorab)
  - Einbettbare Medien - wie im Messenger
  - Locations
- https://structured.email/
