The Perfect XMPP Server Setup (Reloaded)

Holger Weiß

16. Oktober 2018

zedat

Vorwort: Was ist XMPP?

- 1999 unter dem Namen »Jabber« entstanden
- XMPP: eXtensible, Messaging, Presence
- Modularer Standard: Zwei RFCs, hunderte Extensions (XEPs)
- XEPs für Gruppenchat, Avatare, Push, Dateitransfer, Audio/Video, usw.
- So auch für andere/neue Anforderungen nutzbar

zedat

Vorwort: Wie funktioniert's?

- Client-Server-Modell
- Verbindung mit anderen Usern auf demselben, aber auch anderen Servern, wie bei E-Mail
- Adressierung auch wie bei E-Mail (ggf. dieselbe Adresse)
- Plattformübergreifend, diverse Clients zur Auswahl

zedat

Vorwort: Föderation

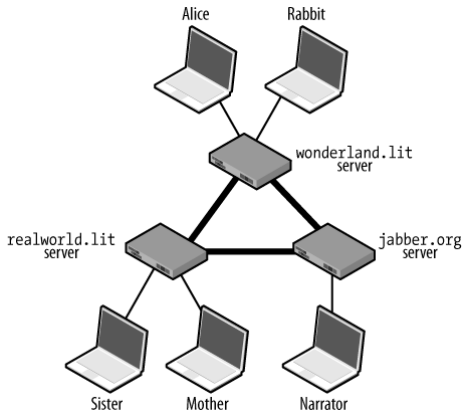

Abbildung: XMPP-Netzwerk

zedat

Vorwort: Wie sieht eine Nachricht aus?

Message-Stanza

```
<message from='bob@example.com'
         to='alice@example.org'
         type='chat'
         id='e0afdf55'>
  <body>Hallo Alice!</body>
</message>
```

zedat

Vorwort: Wofür wird's genutzt?

- Offene Föderation
- Firmeninterner Chat
- Geschlossene Chatlösungen (Game-Server etc.)
- Militär (NATO)
- Unterschiedlichste IoT-Anwendungen

zedat

Modernes IM mit XMPP

- Stream Management
- Carbon Copies
- Message Archive Management
- Client State Indication
- Push Notifications
- HTTP Upload
- OMEMO

Server-Setup

- Am populärsten: ejabberd und Prosody
- Beide Server lagern XMPP-Extensions in Module aus
- Beispiele: `mod_muc`, `mod_push`, `mod_http_upload`
- Problem bei Prosody: Wichtige Funktionalität in Community-Modulen

Prosody-Module

- `mod_csi`
- `mod_csi_battery_saver`
- `mod_http_upload`
- `mod_mam` (*nicht* die Community-Version)
- `mod_mam_muc`
- `mod_smacks`
- `mod_smacks_noerror` *oder:*
- `mod_smacks_offline`
- `mod_pep_vcard_avatar`
- `mod_vcard_muc`
- `mod_cloud_notify` (aktuelle Version)

Weitere Prosody-Module

- `mod_omemo_all_access`
- `mod_auto_answer_disco_info`
- `mod_cache_c2s_caps`
- `mod_graceful_shutdown`
- `mod_invite`
- `mod_s2s_keepalive`
- `mod_server_contact_info`

ejabberd-Konfiguration

- Default-Konfiguration seit ejabberd 18.09 stark vereinfacht
- Alle wichtigen Extensions per Default aktiviert
- Zwingend: XMPP-Domain eintragen
- Optional: OMEMO erlauben
- Optional: SQL als Storage-Backend

ejabberd-Konfiguration

XMPP-Domain eintragen

```
1 hosts:
2   - "example.org" # Eigene Domain statt "localhost"
```

ejabberd-Konfiguration

OMEMO aktivieren

```
modules:
  mod_pubsub:
    # [...]
    force_node_config:
      "eu.siacs.conversations.axolotl.*":
        access_model: open # Statt "whitelist"
```


ejabberd-Konfiguration

SQLite als Storage-Backend

```
default_db: sql
auth_method: sql
sql_type: sqlite
```

- *Für MySQL/MariaDB siehe:* `https://docs.ejabberd.im/admin/databases/mysql/`

ejabberd: Updates einspielen

- SQL-Schema-Updates derzeit noch manuell
- `https://docs.ejabberd.im/admin/upgrade/`

HTTP-Upload an Webserver delegieren

- Webserver als Reverse-Proxy
- Alternativ: Nur PUT-Requests an XMPP-Server forwarden
- Alternativ: Upload komplett an Webserver delegieren
  - Prosody: `mod_http_upload_external`
  - ejabberd: `mod_http_upload` mit `external_secret`
  - Siehe: `https://modules.prosody.im/mod_http_upload_external.html`

Peer-to-Peer-Dateitransfer

- SOCKS5-Proxy
- ejabberd/Prosody: `mod_proxy65`
- DNS-Eintrag, z.B. ein CNAME `proxy.example.org` auf `example.org`
- Port öffnen
  - Default-Port bei ejabberd: 7777
  - Default-Port bei Prosody: 5000

TLS-Einstellungen und Zertifikate

- Valides Zertifikat (z.B. von Let's Encrypt)
- Zertifikatsprüfung bei s2s-Verbindungen vs. Dialback
- TLS-Settings: Strikt vs. Interoperabilität

TLS-on-Connect auf Port 443

- Restriktive Netze erlauben ggf. nur Port 443
- XEP-0368: SRV records for XMPP over TLS
- Prosody: `legacy_ssl_ports = 5223`
- ejabberd: `ejabberd_c2s`-Listener mit `tls: true`
- Port 443 nach 5223 umleiten, z.B. per iptables
- Wenn Webserver auf demselben Port: sslh oder Nginx
- `https://wiki.xmpp.org/web/Tech_pages/XEP-0368`

XMPP-Discovery per DNS

XMPP-Zugang

```
1 _xmpp-client._tcp  IN SRV 5 1 5222 xmpp.example.org.
2 _xmpp-server._tcp  IN SRV 5 1 5269 xmpp.example.org.
```

XMPPS-Zugang (XEP-0368)

```
1 _xmpps-client._tcp IN SRV 1 1  443 xmpp.example.org.
2 _xmpps-server._tcp IN SRV 1 1 5270 xmpp.example.org.
```


Web-Clients: BOSH

- ejabberd: `mod_bosh {}` aktivieren, im `ejabberd_http`-Listener darauf verweisen
- Prosody: `https://prosody.im/doc/setting_up_bosh`

Web-Clients: WebSocket

- ejabberd: Im `ejabberd_http`-Listener auf `ejabberd_http_ws` verweisen
- Prosody: `https://prosody.im/doc/websocket`

BOSH-/WebSocket-Discovery per DNS

TXT-Records

```
1 _xmppconnect IN TXT \
2   "_xmpp-client-xbosh=https://ex.org:5280/bosh"
3 _xmppconnect IN TXT \
4   "_xmpp-client-websocket=wss://ex.org:443/ws"
```


BOSH-/WebSocket-Discovery per HTTP

/.well-known/host-meta

```
1  <?xml version='1.0' encoding='utf-8'?>
2  <XRD
3   xmlns='http://docs.oasis-open.org/ns/xri/xrd-1.0'>
4
5    <Link rel="urn:xmpp:alt-connections:xbosh"
6          href="https://ex.org:5280/bosh" />
7    <Link rel="urn:xmpp:alt-connections:websocket"
8          href="wss://ex.org:443/ws" />
9
10 </XRD>
```

BOSH-/WebSocket-Discovery per HTTP

Alternativ: /.well-known/host-meta.json

```
{
  "links": [
    {
      "rel": "urn:xmpp:alt-connections:xbosh",
      "href": "https://ex.org:5280/bosh"
    },
    {
      "rel": "urn:xmpp:alt-connections:websocket",
      "href": "wss://ex.org:443/ws"
    }
  ]
}
```


Audio/Video-Support

- ejabberd: STUN- und TURN-Server eingebaut (`ejabberd_stun`-Listener)
- Prosody: Externer Server z.B. coturn oder restund
- Authentisierung benötigt Plain-Text-Passwörter!

STUN/TURN-Discovery

SRV-Einträge

```
_stun._udp  IN SRV 0 0 3478 xmpp.example.org.
_stun._tcp  IN SRV 0 0 3478 xmpp.example.org.
_stuns._tcp IN SRV 0 0 5349 xmpp.example.org.
```


Videokonferenzen

- Jitsi Videobridge: `https://jitsi.org/jitsi-videobridge/`
- ejabberd: `ejabberd_service`-Listener
- Prosody: `Component`-Eintrag

ejabberd-Konfiguration für Movim

PubSub: Items-pro-Node-Limit erhöhen

```
modules:
  mod_pubsub:
    # [...]
    max_items_node: 1000
    default_node_config:
      max_items: 1000
    plugins:
      - "flat"
      - "pep" # Requires mod_caps.
```


ejabberd-Konfiguration für Movim

Upload: CORS-Header

```
listen:
  -
    module: ejabberd_http
    # [...]
    custom_headers:
      "Access-Control-Allow-Origin": "*"
      "Access-Control-Allow-Credentials": "true"
      "Access-Control-Allow-Methods": \
        "OPTIONS, HEAD, GET, PUT"
      "Access-Control-Allow-Headers": \
        "Authorization, Content-Type"
```

ejabberd-Konfiguration für Movim

- Dokumentation:
  `https://github.com/movim/movim/wiki/Configure-ejabberd`

Worauf sonst noch achten?

- Privacy Policy (DSGVO)
- Haltezeit von MAM/Uploads (Cron-Job?)
- Spam-Schutz
  - Prosody: `mod_firewall`
  - ejabberd: Notfalls `mod_block_strangers`
- Passwort-Recovery?
- Web-Interface? Zum Beispiel:
  `https://github.com/jabber-at/hp`
- Web-Client? Zum Beispiel:
  `https://conversejs.org/`