Streaming XML With Jabber/XMPP

Ralph Meijer and Peter Saint-Andre

Introduction

This presentation gives an overview of Jabber/XMPP technologies. The following topics will be discussed:

- What is Jabber/XMPP?
- History
- Architecture
- Core Protocol
- Protocol Extensions
- Where is Jabber?
- Example Applications
- Extending Jabber

What is Jabber/XMPP?

Jabber is a set of open technologies for streaming XML between any two points on the Internet.

- Open XML protocols for IM, presence, and more.
- Many open-source implementations.
- Open, peer-to-peer server network.
- Not just IM – a generic XML routing platform.
- Core protocols formalized by the IETF as XMPP.
- Extensions defined by the Jabber Software Foundation.

History of Jabber/XMPP

- Early 1998: Jeremie Miller starts jabberd server project.
- Jan 4 1999: First announcement on Slashdot.
- Late 1999: Core team sponsored by Webb Interactive Services.
- March 2000: Jabber Inc. founded by Webb.
- May 2000: jabberd 1.0 released.
- October 2000: jabberd 1.2 released (core protocols stable).
- January 2001: jabberd 1.4 released.

History (continued)

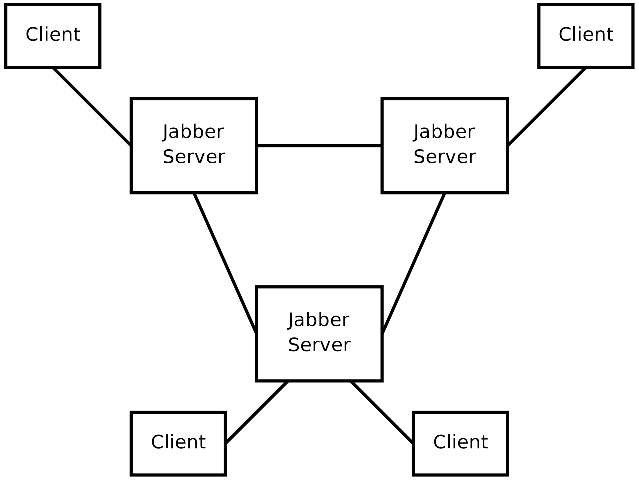
- August 2001: Jabber Software Foundation (JSF) formed to manage protocols.
- January 2002: JSF submits core protocols to IETF as XMPP.
- October 2002: IETF forms XMPP Working Group.
- September 2003: Last Call issued by IESG.
- February 2004: IESG approves XMPP specs as Proposed Standards.
- October 2004: IETF publishes XMPP RFCs (3920-3923).

Architecture

- Usually client-server (logical peer-to-peer).
- Clients connect to servers to access network.
- Direct connections between servers.
- Reverse DNS lookups to prevent server spoofing.
- Domain-based routing, similar to email (but no multi-hop).
- Various services (components) associated with servers.
- Once on network, can communicate with all servers/services.

Architecture (continued)

- Distributed architecture is highly scalable.
- All entities have presence (network availability information).
- Client and server connections are stateful.
- Long-lived TCP connections (or can use HTTP binding).



Core Protocol (RFC 3920)

XML streams: open-ended "document" in each direction between two entities.

```
SEND: <stream:stream to=’my.host’ xmlns=’jabber:client’
       xmlns:stream=’http://etherx.jabber.org/streams’>
RECV: <stream:stream from="my.host" xmlns="jabber:client’
       xmlns:stream=’http://etherx.jabber.org/streams’
       id=’3C5D3B03’>

      ... UNBOUNDED NUMBER OF XML "FRAGMENTS" ...

SEND: </stream:stream>

RECV: </stream:stream>
```

Core Protocol: XML Stanzas

- XML stanzas: first-level children of stream root.
- Message: "push" semantics, similar to email (except faster!).
- Presence: "pubsub" semantics to broadcast network availability.

```
SEND:   <presence/>

RECV:   <message to='ralphm@my.host'
                 from='guy@another.host/resource'>
          <body>Hi</body>
        </message>
```

Core Protocol: XML Stanzas

- IQ: "request-response" semantics, similar to HTTP.

```
SEND:   <iq type=’set’ id=’some-id’>
          <query xmlns=’somenamespace’>
            <foo/>
          </query>
        </iq>
RECV:   <iq type=’result’ id=’some-id’/>
```

Core Protocol: Security / i18n

- SSL/TLS (RFC 2246) for channel encryption.
- SASL (RFC 2222) for strong authentication.
- Unicode/UTF-8 support for internationalization.
- Fully internationalized addresses.
- Addresses: domain, node@domain, node@domain/resource.
- Multiple resources allowed per entity.

Core Protocol: Extensibility

- Stanzas may contain any properly-namespaced XML.
- Rule: if you don’t understand it, don’t process it.
- Many, many extensions have been defined.
- Public extensions defined by JSF in Jabber Enhancement Proposals (JEPs).
- Define your own extensions for custom functionality.

Basic IM Extensions (RFC 3921)

- Contact list management (rosters) using IQ stanzas.
- Subscriptions to presence information.
- One-on-one chat.
- Block/allow lists.

Other Popular IM Extensions (JEPs)

- Service Discovery to find entities and supported features.
- Entity Capabilities for dynamic feature advertisement.
- Multi-User Chat for chat rooms (similar to IRC).
- File Transfer to exchange large or binary files.
- XHTML-IM for formatted messages.
- Extended Presence (geolocation, mood, tunes, avatars, etc.).

Even More Extensions (JEPs)

- SOAP Over XMPP.
- Jabber-RPC – XML-RPC over XMPP.
- Data Forms – lightweight forms processing (workflow, etc.).
- Advanced Message Processing – reliable delivery of message stanzas.
- Publish-Subscribe – generic pubsub semantics for content syndication (RSS/Atom) etc.

Where is Jabber? (IM Applications)

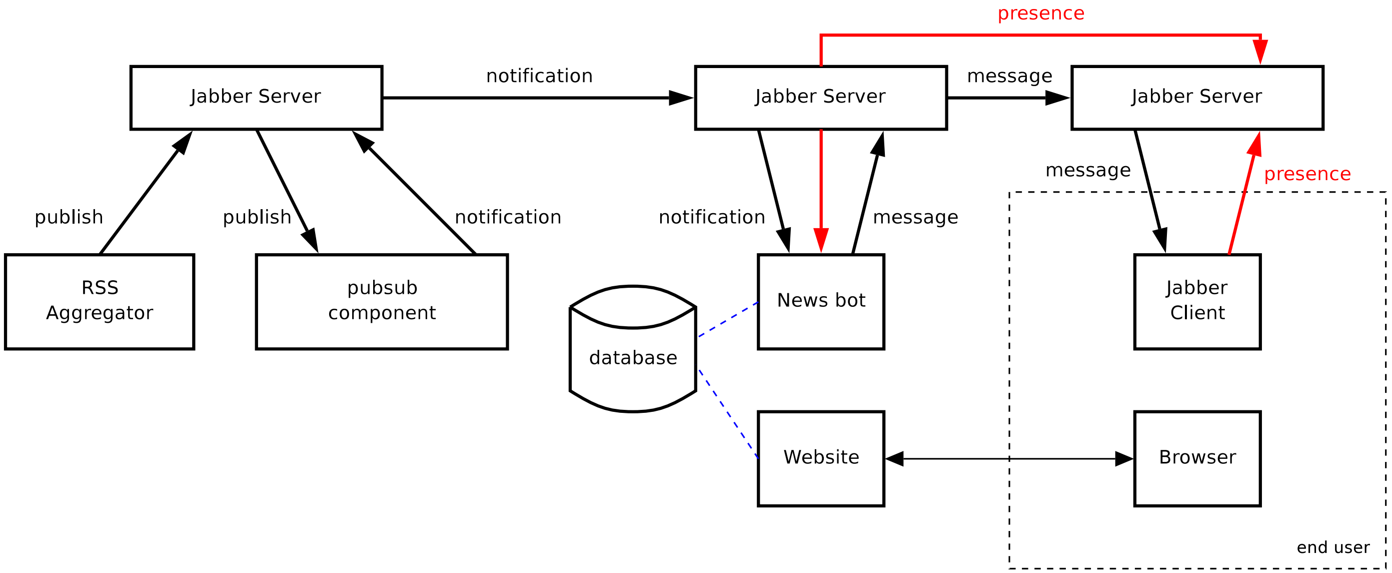
- Most major Wall Street firms are running Jabber.
- Big telcos/ISPs (France Telecom, Bell South, Orange, etc.).
- U.S. Government: U.S. Army Future Combat Systems (etc.).
- IBM: emergency management network in Washington D.C. (CAPWIN).
- Large companies (HP, FedEx, EDS, Qualcomm, AT&T, etc.).
- Lots of small companies, too: 300,000+ server downloads.
- 10+ million Jabber IM users.

Beyond IM

- EBS: $100-billion-a-day spot trading system.
- PubSub.com: real-time notifications from 10+ million RSS/Atom feeds.
- Nokia: pan-European Nokia game.
- Sputnik: wireless access point.
- TrakM8: vehicle tracking system.
- Reynolds & Reynolds: automotive dealer management system.
- HighStreet Networks: real-time network management.
- Inkboard: open-source project for SVG whiteboarding over XMPP.

Pubsub Applications

- Geolocation systems (e.g., package tracking).
- WebDAV events (draft-hildebrand-webdav-notify-01).
- Information Content Exchange (www.icestandard.org).
- Content syndication: Mimir



Extending Jabber

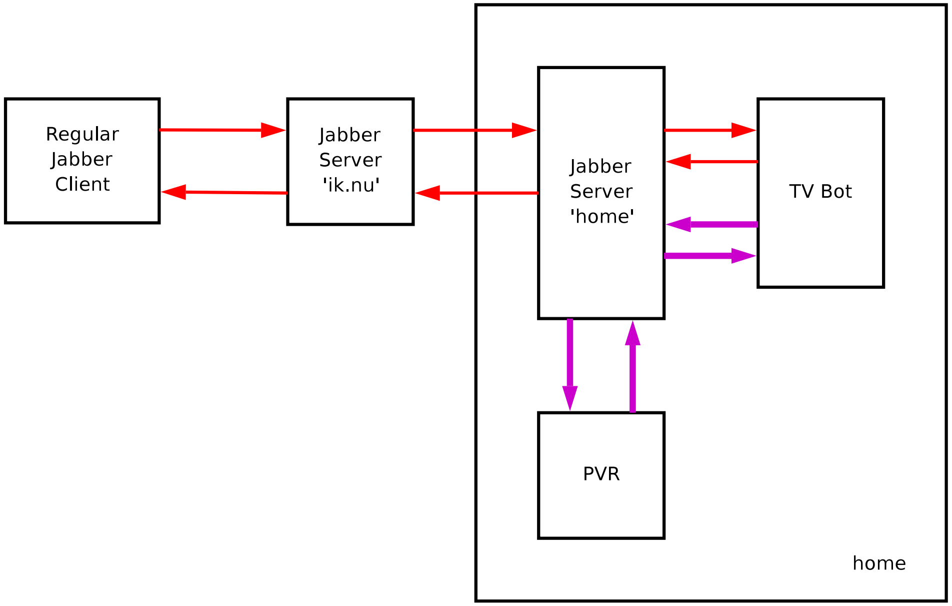
Programming your PVR using Jabber

What do we need?

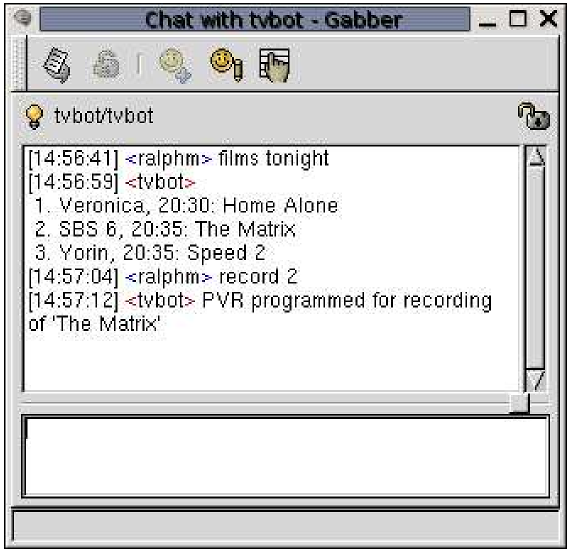
- A regular Jabber client (`ralphm@ik.nu/work`)
- A Jabber bot (`tvbot@home/tvbot`)
- A PVR that is also a Jabber client (`pvr@home/pvr`)
- A namespace (`http://ralphm.net/protocols/pvr`)

We chat to the bot like it is another person. The bot chats to the pvr using our new namespace.

The architecture



Conversation with the bot:



```
[14:56:41] <ralphm> films tonight
[14:56:59] <tvbot>
 1. Veronica, 20:30: Home Alone
 2. SBS 6, 20:35: The Matrix
 3. Yorin, 20:35: Speed 2
[14:57:04] <ralphm> record 2
[14:57:12] <tvbot> PVR programmed for recording
of 'The Matrix'
```

Conversation in protocol:

```
<<< <message to=’tvbot@home/tvbot’ type=’chat’>
      <body>films tonight</body>
    </message>
>>> <message to=’ralphm@ik.nu/work’ type=’chat’
             from=’tvbot@home/tvbot’>
      <body>
        1. Veronica, 20:30: Home Alone
        2. SBS 6, 20:35: The Matrix
        3. Yorin, 20:35: Speed 2
      </body>
    </message>
<<< <message to=’tvbot@home/tvbot’ type=’chat’>
      <body>record 2</body>
    </message>
>>> <message to=’ralphm@ik.nu/work’ type=’chat’
             from=’tvbot@home/tvbot’>
      <body>PVR programmed for &apos;The Matrix&apos;</body>
    </message>
```

Conversation bot with PVR

```
<<< <iq type=’set’ to=’pvr@home/pvr’>
      <pvr xmlns=’http://ralphm.net/protcols/pvr’>
        <record>
          <date>20031016</date>
          <station>36</station><!-- SBS 6 -->
          <program_id>14</program_id>
        </record>
      </pvr>
    </iq>
>>> <iq type=’result’ from=’pvr@home/pvr’ to=’tvbot@home/tvbot’
      <pvr xmlns=’http://ralphm.net/protcols/pvr’>
        <info>
          <program_name>The Matrix</program_name>
          <station_name>SBS 6</station_name>
          <start>20031016T20:35:00</start>
          <end>20031016T22:05:00</start>
        </info>
      </pvr>
    </iq>
```

Questions?

Resources:

- XMPP: `<http://www.xmpp.org/>`
- Extensions: `<http://www.jabber.org/jeps/>`
- Software: `<http://www.jabber.org/software/>`
- Your own server: `<http://www.jabber.org/admin/>`
- Peter: `stpeter@jabber.org`
- Ralph: `ralphm@jabber.org`