

# Apache Vysper: XMPP Server - IM and beyond

ApacheCon EU 2008  
Bernd Fondermann

# What is Apache Vysper?

- A XMPP Server

# What is XMPP?

- eXtensible Messaging and Presence Protocol
- “Jabber”

# Who is developing XMPP?

- XSF - XMPP Standards Foundation
  - open community
  - members are individuals, not companies
  - [xmpp.org](http://xmpp.org)
  - even more at [jabber.org](http://jabber.org)

# Where is XMPP defined?

- XMPP core => RFC 3920
- XMPP messaging & presence => RFC 3921
- Many extension (XEPs) defined by XSF
  - <http://xmpp.org/extensions>
- mailing list: standards@xmpp.org

# How do endpoints connect?

- Client/Server
- Server-to-Server (federation)
- TCP/IP
- Synchronous HTTP (BOSH)
- stateful, durable connection
- small communication overhead

# How do endpoints communicate?

- UTF-8 streams
- XML subset
- exchange of “stanzas”:
  - `<message from='...' to='...' >`
  - `<xyz> ... some useful xml ... </xzy>`
  - `</message>`

# How are endpoints addressed?

- JID = Jabber ID = “Entity”
  - very similar to e-mail address
  - berndf@vysper.org
- A connected entity is a ‘bound resource’
  - berndf@vysper.org/laptop
  - berndf@vysper.org/mobile

# Handshake stages

- 1. Secure the stream using StartTLS
- 2. Authenticate using SASL
- 3. Bind logical resource
- 4. Lookup services provided by server
- Stages depend on features
- Finally, the handshake is completed

# Message, Presence, IQ

- 3 types of stanzas
  - <message> point-to-point
  - <presence> broadcast
  - <iq> request-response
- so, all basic message paradigms are covered

# Message

- ```
<message
  to='romeo@example.net'
  from='juliet@example.com/balcony'
  type='chat'
  xml:lang='en'>
  <headline>Greetings</headline>
  <thread>threadID_ab02</thread>
  <body>Wherefore art thou, Romeo?</body>
  <body xml:lang='cz'>Pro&#x010D;e&#x017D; jsi
ty, Romeo?</body>
</message>
```

# IQ (info/query)

- request/response
- both, client & server may send request
- extensively used in service discovery

# IQ example 'set'

- ```
<iq type='set' id='rpc1'
  from='client@site-b.com/jrpc-client'
  to='service@site-a.com/jrpc-server' >
  <query xmlns='jabber:iq:rpc'>
    <methodCall>
      <methodName>asf.getACEUCity<methodName>
      <params><param>
        <value><i4>2008</i4></value>
      </param></params>
    </methodCall>
  </query>
</iq>
```

# IQ example 'result'

- ```
<iq type='result' id='rpc1'  
  from='service@site-a.com/jrpc-server'  
  to='client@site-b.com/jrpc-client'>  
  <query xmlns='jabber:iq:rpc'>  
    <methodResponse>  
      <params><param>  
        <value><string>Amsterdam</string></value>  
      </param></params>  
    </methodResponse>  
  </query>  
</iq>
```

# Presence

- Broadcast to all subscribers
- Advanced subscription model
- Presence: “online”, “offline”, “at hackathon”
- Roster: all direct contacts

# Stanza extensibility

- XML namespaces specify stanza context
  - e.g. “jabber:server”, “jabber:client”
- every stanza has standard XML child nodes
- stanzas can have custom child nodes
  - those need to have custom namespaces
  - if not understood, nodes are ignored

# XMPP Extensions

- 4 “final standard” XEPs

Data Forms	XEP-004
In-Band Registration	XEP-030
Service Discovery	XEP-077
XML-RPC via XMPP	XEP-009

# XMPP Extensions

- many best-practice, clarifications, add-ons

Jingle (voicechat)	XEP 0166, etc.
BOSH (HTTP transport)	XEP124 & XEP206
Groupchat (IRC)	XEP 0045
XHTML chat	XEP 0071
file transfer	XEP 0096
stream compression	XEP 0138

# Apache Vysper

- pronounced “whisper”
- XMPP server implementation
- Java 5
- MINA
- Spring Framework

# Status

- in development: since 2006
- @labs.apache.org: since 2007
- committers: me + contributors welcome
- RFC3920: good coverage
- RFC3921: dev in progress
  - message exchange working

# Goals

- fully implement RFC 3920 & 3921
- fully implement “standard final” XEPs
- plug-in extension mechanism
  - other XEPs
  - custom extensions

# Thank you!

mailing list: [labs@labs.apache.org](mailto:labs@labs.apache.org)

private mail: [bernd@brainlounge.de](mailto:bernd@brainlounge.de)