MongooseIM - Messaging that Scales

Michał Ślaski
What is MongooseIM?
What is MongooseIM?

- **Instant messaging** for Social Media, Gaming and Telecommunications
What is MongooseIM?

• **Instant messaging** for Social Media, Gaming and Telecommunications

• Solution designed for **high volume**
What is MongooseIM?

- **Instant messaging** for Social Media, Gaming and Telecommunications
- Solution designed for **high volume**
- Easily **scalable** distributed system
What is MongooseIM?

- **Instant messaging** for Social Media, Gaming and Telecommunications
- Solution designed for **high volume**
- Easily **scalable** distributed system
- Highly **customisable** platform thanks to industry standards and Open Source technology
Target domains
Target domains

Telecoms

(chat features are a part of standard communication software)
Target domains

Telecoms
(chat features are a part of standard communication software)

Social Media
(messaging is the key part of social web sites)
Target domains

**Telecoms**
(chat features are a part of standard communication software)

**Social Media**
messaging is the key part of social web sites

**Gaming**
multi-user chats are an integral part of the gameplay and enhance the user experience
Easy to work with

• Support of XMPP
  - a protocol used for Facebook chat and Google Talk
  - seamless integration with many libraries and client applications
XMPP Properties

Open standard
Secure
Flexible
Decentralised
Efficient
Proven
XMPP Properties

Open standard

Secure

Flexible

Decentralised

Efficient

Proven

Standard formalised by the IETF through RFCs and extensions are published by the XSF
XMPP Properties

Open standard

Secure

Flexible

Decentralised

Efficient

Proven

Channel encryption, strong authentication, may be isolated from the public network
XMPP Properties

Open standard

Secure

Flexible

Decentralised

Efficient

Proven

Custom functionality can be built on top of XMPP
XMPP Properties

Open standard

Secure

Flexible

Decentralised

Efficient

Proven

Anyone can run XMPP server and there is no central master server
XMPP Properties

Open standard

Secure

Flexible

Decentralised

Efficient

Proven

Solves issues of polling approaches
XMPP Properties

Open standard

Secure

Flexible

Decentralised

Efficient

Proven

In use since 1998, many implementations and deployments, millions of end users
XMPP Properties

Open standard

Secure

Flexible

Decentralised

Efficient

Proven
Features

- Support for **WebSockets** - ready for modern chat-enabled web applications
- **Multi-User Chat** (XEP-0045)
- Service Discovery (XEP-0030)
- vCard (XEP-0054)
- Privacy Lists (XEP-0016)
- Private XML Storage (XEP-0049)
Extensible

- Customisable - provides a base for bespoke solution to one’s specific needs
  - push notifications to mobile devices
  - authentication plugins
  - reliable message delivery in mobile networks

- Has open sourced regression tests
  - github.com/esl/ejabberd_tests
Scalable

- Configurable database backends
  - Mnesia for simple deployments
  - MySQL for persistent data
  - Mnesia or Redis for transient data
Scalable

MongooseIM

Pluggable modules
- Websockets
- Multi-User Chat
- Rosters
- Offline Storage
- Last Activity
- Privacy
... and more!

Authentication
- SQL
- Mnesia
- External
- Anon

Session manager
- Mnesia
- Redis

Internal router

Client handlers

Redis

MySQL

Other Jabber Server

Monday, 8 April 13
Load tests - environment

Test environment

Erlang/OTP R15B02
MongooseIM 1.1
MySQL 5.5.24
Ubuntu 12.04 LTS 3.2.0-23-generic
Tsung 1.5.0a
redis 2.6.10
ejabberd 2.1.11
Load tests - dedicated box

MongooseIM
AMD 8x3.3GHz, 32GB RAM

MySQL
Load tests - “max users”

User arrival rate: 150/s
User count: 400k
Roster size: 100
No message exchange

The aim was to check resource usage with maximum possible count of online users connected to server.
Load tests - “max message rate”

User arrival rate: 100/s
User count: 75k
Roster size: 100
Message rate: ~21k per second

The aim was to check resource usage when maximum message traffic is generated.
Load tests - Amazon EC2

• M1 Extra Large Instance
  - 15 GiB memory
  - 8 EC2 Compute Units
    (4 virtual cores with 2 EC2 Compute Units each)
  - I/O Performance: High
Load tests - Amazon EC2

MongooseIM
m1.xlarge

MongooseIM
m1.xlarge

MongooseIM
m1.xlarge
Load tests - 3 MongooseIM + Mnesia

User arrival rate: 130/s
User count: 50k
Roster size: 100
Presence rate: 120 per second (12k msg/s)

The aim was to check resource usage with all user data like passwords and rosters in Mnesia.
Load tests - Amazon EC2

MongooseIM
m1.xlarge

MongooseIM
m1.xlarge

MongooseIM
m1.xlarge

MySQL
m1.xlarge

Monday, 8 April 13
Load tests - 3 MongooseIM + MySQL

User arrival rate: 130/s
User count: 330k
Roster size: 100
Presence rate: 120 per second (12k msg/s)

The aim was to check resource usage with all user data like passwords and rosters in MySQL
Load tests - Amazon EC2

- MongooseIM
  - ml.xlarge
- MongooseIM
  - ml.xlarge
- MongooseIM
  - ml.xlarge
- redis
  - ml.xlarge
- MySQL
  - ml.xlarge
Load tests - 3 MIM + redis + MySQL

User arrival rate: 2000/s
User count: 380k
Message rate: 8k per second

The aim was to check characteristics of a cluster of 3 MongooseIM nodes with session data in redis instead of mnesia.
ooVoo is the largest independent video communication service provider

- 70mm users
- Free, up to 12 way high-quality service. Leader in group calling
- Apps for PC, Mac, Web, iPhone, iPad, Android, Facebook
- 10+ billion video minutes in 2012
- Platform for Shared Experiences

courtesy of Alex Fok, System Architect at ooVoo
Main Challenge - Scale

- 2010
  - 600K online users
  - 10M registered users
  - 500 messages/sec
- Due to constant and rapid usage growth – infinite scale requirement
Today (2 years after)

- 11 scale units in 2 geographical locations
- x10 Usage growth
  - 2.2M online users (connected)
  - 70M registered users
  - 5,000+ messages per second
- 99.98% uptime

courtesy of Alex Fok, System Architect at ooVoo
courtesy of Alex Fok, System Architect at ooVoo
courtesy of Alex Fok, System Architect at ooVoo
Benefits

- Proven solution from an Erlang-focused company with expertise in instant messaging (over 300 clients since 1999)
- Rapid deployment
- Interoperability ensured by industry standards
- Minimal capital expenditure due to efficient runtime platform
- Pluggable authentication for ease of integration
Download
www.erlang-solutions.com/downloads/

Fork and contribute
https://github.com/esl/ejabberd

Contact us
ejabberd@erlang-solutions.com
Load tests - “WebSockets vs. BOSH”

User arrival rate: 90/s
User count: 10k
Message rate: 6k/s

The aim was to compare resource usage of WebSockets vs. BOSH.