xmpproxy, a proxy server for the XMPP protocol

Project description and plan

Ralph Krimmel

November 1, 2011

Contents

1 Abstract

"The extensible messaging and presence protocol xmpp is an open technology for real time communication which powers a wide range of applications including instant messaging, presence, multi-party chat, voice and video calls, collaboration, lightweight middleware, content syndication and generalized routing of xml data." *http://xmpp.org/about*

Being widely used in several large communication platforms such as `Google talk`, `XMPP` has become an important protocol for communication via network. This document will describe an issue using `XMPP`'s instant messaging capabilities in a multi client environment and propose a possible solution. It also contains a time schedule for implementing this solution as a project for the course "Applied IT project" at the University of Gothenburg.

2 Introduction

The `XMPP` or "jabber"-protocol supports multiple clients to be connected to the same account at the same time. From the server sides view a connected client is called a *ressource*. The consequence of multiple ressources for the same account is that the server has to decide to which ressource an incoming message, or *stanza* in xmpp terminology, is routed to. This is done via the so called *presence priority* a client can connect with. The client with the higher priority will receive the message. If two or more ressources have the same priority, the server may use some other rule to decide between those or deliver the message to all of them. For example, the server could use the most recent connect time or the most recent activity time. However, the server is not allowed to deliver the stanza to an available resource with a negative priority.

This behaviour leads to the problem, that conversations and logfiles of conversations may not be complete on every client, if one client connects with a higher priority. Figure 1 shows an example where this would be the case. Client A does not know the parts of the conversation that are held from Client B. Also, not just incoming messages will be missing, Client A will not even know about outgoing stanzas from Client B.

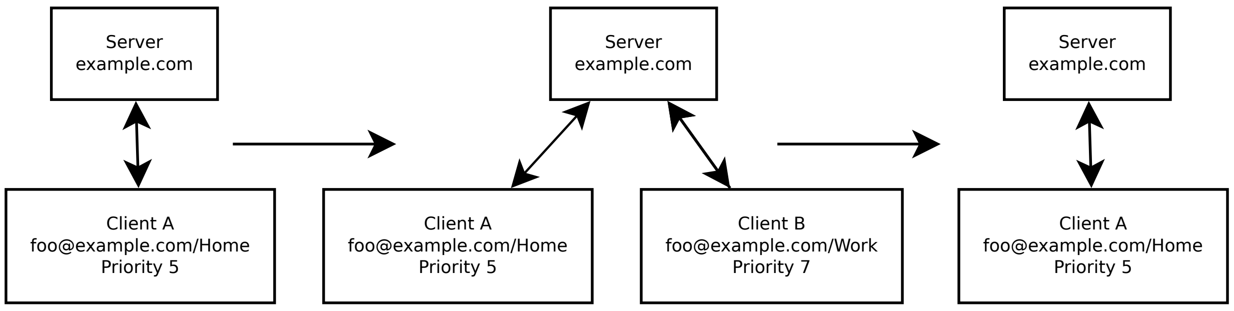

Figure 1: Clients with 2 different priorities are connected to the same account.

A possible solution for this issue may be the use of a proxy server as it can be seen in figure 2. This server will make sure, that every incoming stanza will reach both clients and every outgoing stanza will be sent to every ressource as well as to the targeted server.

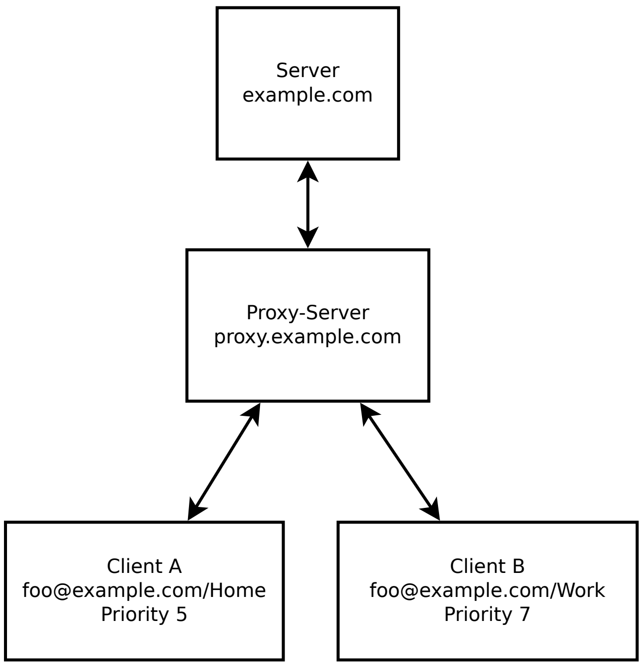

Figure 2: Two clients connected to a proxy.

3 Technology

3.1 Software choice

For implementing the xmpp proxy server several design descisions had to me made. First of all a programming language was chosen. Possible languages were Perl, C, C++ and Java because of the preferences of the author. The criteria for the programming language was first of all the availability of proper libraries that deal with the xmpp protocol, both server and client side. Unfortunately, this excluded all languages except perl because there are just few xmpp server libraries at all. Therefore the xmpp proxy server will make use of the `djabberd` xmpp/jabber server and the perl modules `Net::XMPP/Net::XMPP::Client` although it lacks a proper documentation.

3.2 Djabberd

Djabberd is a modular, scalable and extensible jabber server written in perl where almost everything defers to hooks to be implemented by plugins. It is written and maintained by *Brad Fitzpatrick*, *Artur*

Bergman and *Jonathan Steinert*. When this document was created, the most recent version of djabberd was 0.85, released on the thirteenth of June in 2011. Having a modular structure, djabberd is a perfect framework for building the required proxy server because it offers hooks for modifying the behaviour of the authentication and authorization process, roster storage and message delivery. Also, `djabberd` can be used in environments which require a high performance for it is asynchrone/event-based, using epoll on linux 2.6.

3.3 Net::XMPP

`Net::XMPP` is a perl package that provides access to XMPP client side functions for the perl programming language. It is maintaned by *Eric Hacker* and versioned 1.02 at the time this document was written. It can be utilized using its object oriented interface. There are objects available which hold the actual client functions like login, wait for messages, send messages and logout. Also there are objects for storing and handling Jabber IDs (`Net::XMPP::JID`), messages (`Net::XMPP::Message`), presence information (`Net::XMPP::Presence`) and Info/Query namespaces (`Net::XMPP::IQ`).

4 Tasks

The following tasks are to be completed:

- Studies of literature
- Write and test wrapper classes for djabberd and Net::XMPP data structures
- Write and test module for proxy authentication
- Write and test module for proxy roster handling
- Write and test module for proxy presence handling
- Write and test module for message delivery
- (Optional) Write and test module for proxy groupchat
- (Optional) Write and test module for proxy jingle (Voice/Video via jabber)
- Write Documentation
- Write Report
- Create presentation

5 Schedule

The xmpproxy software will be written in the context of the course "Applied IT Project" course at the University of Gothenburg. This requires to "formulate and execute a finite Applied IT project within a limited time-frame". Therefore it is neccessary to schedule the tasks described in the previous sections. Table 1 shows the percentage of effort and the estimated time every task will take. The minimum amoount of time this project will need is about 200 hours.

Task	**Percentage**	**Estimated time (in h)**
Studies of literature	10	20
Wrapper classes for djabberd and Net::XMPP data structures	4	8
Testing	1	2
Module for proxy authentication	4	8
Testing	1	2
Module for proxy roster handling	8	16
Testing	2	4
Module for proxy presence handling	8	16
Testing	2	4
Write module for message delivery	16	32
Testing	4	8
Documentation	15	30
Report	15	30
Presentation	10	20
(Optional) Module for proxy groupchat	-	
(Optional) Module for proxy jingle (Voice/Video via jabber)	-	

Table 1: Estimated time for several tasks

6 Literature

- `Programming jabber - extending XML messaging` by DJ Adams
- `http://xmpp.org/rfcs/rfc6120.html`
- `http://xmpp.org/rfcs/rfc6121.html`
- `http://xmpp.org/rfcs/rfc6122.html`
- `http://search.cpan.org/~hacker/Net-XMPP-1.02/`
- `http://search.cpan.org/~mart/DJabberd-0.85/`