Implementation description

Varain

Helsinki December 3, 2006

Software Engineering Project

UNIVERSITY OF HELSINKI
Department of Computer Science

Course
581260 Software Engineering Project (6 cr)

Project Group
Kankaanniemi Marko
Lampila Pekka
Ruottu Toni
Viita Antti

Instructor
Yrjänäinen Sampo

Client
Manner Jukka

Project Masters
Taina Juha

Homepage
`http://www.cs.helsinki.fi/group/varain`

Change Log

Version	Date	Modifications
1.0	1.12.2006	First version

Contents

1 Introduction

This document describes various things about the Varain software project and it's product that should be useful for maintaining the software. The document is not meant to be read by itself, but rather with the Implementation Plan document and the source code.

First part of the document describes the installation process of the software. After that everything that wasn't done according to the Implementation Plan is discussed. Rest of the document deals with different kinds of information that concerns the maintenance of the code.

2 Glossary

RSVP Resource Reservation Protocol. Protocol for reserving resources in a IP network.

LRSVP Local Resource Reservation Protocol. Modification of the RSVP protocol, that provides the means to reserve local resources.

KOM RSVP RSVP implementation made by Multimedia Communications Lab at Technische Universität Darmstadt.

3 Installation

You need to get the source package for KOM RSVP 3.0f [Kar] and a patch containing the LRSVP-related modifications mostly written by Adrien Bauffe and partly by the Varain project team. After that you need to patch KOM RSVP and compile it. The procedure is a little different depending on whether you are building KOM RSVP to run as an RSVP daemon or as an LRSVP proxy. The following two subsections provide the instructions for both.

3.1 LRSVP Proxy

1. Get the source package `rsvp-rel3.0f.tgz` and the patch file `lrsvp.diff`

2. Untar the package by running

```
tar zxvf rsvp-rel3.0f.tgz
```

3. Go to the directory `rsvp-rel3.0f/`

4. Patch the source code by executing the following command

```
patch -p1 < /path/to/lrsvp.diff
```

5. Check your GCC version. You need at least GCC version 3 but no later than 3.3.4 to compile KOM RSVP. If your default GCC version is not compatible, you can install, e.g., version 3.3.4 and set the environment variables CC and CXX to point to the compatible version.

For example:

```
export CC=/opt/gcc-3.3.4/bin/gcc
export CXX=/opt/gcc-3.3.4/bin/g++
```

6. Execute the following command

```
./configure --disable-mpls --disable-java --disable-java-api \
--disable-refresh-reduction --disable-dns --disable-onepass \
--enable-virt --enable-active-api --disable-lrsvp \
--enable-lrsvp-proxy
```

7. Execute the following command

```
make dep && make
```

If everything went well, you should now be able to see binaries in the `rsvp-rel3.0f/bin/` directory

3.2 Daemon

1. Get the source package `rsvp-rel3.0f.tgz` and the patch file `lrsvp.diff`

2. Untar the package by running

```
tar zxvf rsvp-rel3.0f.tgz
```

3. Go to the directory `rsvp-rel3.0f/`

4. Patch the source code by executing the following command

```
patch -p1 < /path/to/lrsvp.diff
```

5. Check your GCC version. You need at least GCC version 3 but no later than 3.3.4 to compile KOM RSVP. If your default GCC version is not compatible, you can install, e.g., version 3.3.4 and set the environment variables CC and CXX to point to the compatible version.

For example:

```
export CC=/opt/gcc-3.3.4/bin/gcc
export CXX=/opt/gcc-3.3.4/bin/g++
```

6. Execute the following command

```
./configure --disable-mpls --disable-java --disable-java-api \
--disable-refresh-reduction --disable-dns --disable-onepass \
--enable-virt --enable-active-api --enable-lrsvp
```

7. Execute the following command

```
make dep && make
```

If everything went well, you should now be able to see binaries in the `rsvp-rel3.0f/bin/` directory.

4 Unimplemented Features

The original low priority goal of creating expedited refresh (ER-flag) support was dropped due to complexity of the required changes and difficulties in understanding KOM RSVP code base.

We initially planned that you could give the local network interfaces as parameters to `lrsvp_proxy`, but gave up that idea when it resulted in really strange behavior. There is apparently some kind of a memory-related bug somewhere in the code that appears only when you give the interface names as parameters. The bug was obviously not in our code as we went through it very carefully line by line.

5 Source code maintenance

5.1 Overview

Our project is based on already implemented project called KOM RSVP [Kar]. We also had a add-on project made by Adrian Bauffe [Bau05] which gives LRSVP functionality to KOM RSVP suite. Only Bauffe did not quite succeed on his task and some functionality was lacking.

KOM RSVP itself is a quite big project which consists over 20000 lines of code. It implements fully RSVP specification and includes a few add-ons also. At the moment KOM RSVP project is a dormant ant latest release is from year 2003. The source code compiles but it is very badly documented.

Bauffe's addition to KOM RSVP is over 2000 lines in total. It suppose to give LRSVP functionality to KOM RSVP suite. The implementation is not complete and it has some major problems. The code documentation is also really bad or non existing.

Our team concentrated to correct Bauffe's mistakes and sanitize the code. We did not make so many new lines of code but we corrected many bugs and made the whole systems work. We also made the configuration and compilation process much easier.

5.2 KOM RSVP Code Base

KOM RSVP is implemented using C++ programming language. There are also a configuration file parser done with lex. Build system on Unix based machines is based on GNU autotools, but it has many unique things like using some ready made Makefiles etc.

Like mentioned above, the KOM RSVP code base is over 20000 lines. There is about 60 files with C++ code, not counting the headers. The files have been separated in different directories based on their functionality, and on which platforms they work in. Unfortunately this causes some problems browsing the code, since all the files are deep in the directory tree. There is some documentation about the KOM RSVP code base, but it is not by any means exhaustive. Furthermore the code itself has almost no comments at all.

Unorthodox build system and file layout, sheer amount of the code and lack of documentation means that it is very hard for a developer to get to know the KOM RSVP code base.

5.3 Changes Made by Adrien Bauffe

Our implementation builds on existing, but broken implementation of Adrien Bauffe's [Bau05], which is shortly described in this chapter and in detail in Bauffe's "LRSVP Extension - Technical Report" that is included in the implementation.

Bauffe added KOM RSVP some C++ classes that implements the LRSVP protocol core functionality. He wrote a proxy application which runs in a separate process and uses the classes. He also wrote an LRSVP-specific command line tool (`lrsvpTest`) for creating reservations by hand. His package also included some redundant command line utilities (`Kask`, `Kresv`) for managing reservations. We believe that they are left over code that was used before lrsvpTest was written. He added some LRSVP extensions to the KOM RSVP API and also modified the KOM RSVP core to detect LRSVP packets and hand them over UDP to his LRSVP proxy process.

5.4 Changes Made by Us

The major problem with Bauffe's implementation was the fact that intermediate RSVP routers didn't pass Path and PathReq messages to the RSVP daemon. The solution to this problem was to include the router alert flag in Path and PathReq messages. The router alert flag is set in the options field of an IP packet. It instructs the router to pass the packet to the user software instead of just forwarding it to the next hop.

We also modified the configure script so that you can use configuration parameters to compile LRSVP daemon or proxy instead of having to write definitions in `RSVP_Config.h`.

We changed the location of the proxy's local network interface configuration file from `$PWD/config.cfg` to `/etc/komrsvp-proxy.conf`.

All LRSVP specific options were taken from the `lrsvpTest` tool written by Bauffe and incorporated into the original `recvapi` and `sendapi` tools.

6 Conclusion

There are many concerns about this project's outcome in the future. The mother project KOM RSVP is at the moment dormant and the latest release is starting to be also quite old. KOM RSVP also includes some add-ons which are not described in any RSVP RFC. There are no public information about KOM RSVP suite future. If they are not going to update and maintain the whole suite either the whole software becomes obsolete or there will be a spin-off. In case of a spin-off most probably our patch set won't work any more. And this would apply also to other extensions too.

Lack of clear and good documentation of KOM RSVP can be considered to be a real disadvantage. All the code is very little commented and there are no automatic testing system in place. So the value of the code can be taken really low. We found out reverse engineering and modelling was painful and time taking. Also we though that design selections were not as smart as they could be.

LRSVP extension to RSVP in only in draft status at the time of writing this document. It has already changed during this project (ER-bit was separated from it) so it is not clear what will be in the final definition of LRSVP (our implementation is based on what is the definition currently). So there might be modifications to this specification we are using compared to the one that will get the RFC status. This can become an issue if modifications to the draft are going to be significant.

Adrian Bauffe may also have had really hard time doing his job with this code. But unfortunately he decided not to do good commentary on his code or even document his findings well. He also had mis-information in his documents and some assumptions were clearly wrong. Overall he made lots of changes but as he realized that he is not going to get it working he simply deleted some modifications from code and left it broken instead of documenting those and just commenting out broken parts. So Bauffe's modifications alone are not so useful and the overall value of his production is also at least questionable.

Starting level to continue this project from the code base we had is really high. And we strongly suggest that one should look all the other alternatives well before starting to continue with this 'mess'. Complete re-write and re-design should not be left out of consideration and our team will recommend this choice.

References

Bau05 Bauffe, A., LRSVP extension to KOM RSVP, 2005.

Kar Karsten, M., KOM RSVP engine. URL `http://www.kom.tu-darmstadt.de/en/downloads/software/kom-rsvp-engine/`.