RTP SIMULATION UNDER OMNeT++: PROBLEMS AND SOLUTIONS

Pantelis Stampoulis
University of Macedonia
Egnatias 156
540 06 Thessaloniki, Greece
pand1988@hotmail.com

Panayotis Fouliras
University of Macedonia
Egnatias 156
540 06 Thessaloniki, Greece
pfoul@uom.gr

1. INTRODUCTION

RTP [1] is one of the most important protocols for multimedia streaming. Consequently it was implemented in the INET framework for use with OMNeT++. However, its implementation status had been termed 'incomplete' for a long period of time. Even recently, the 'Protocol Matrix' of INET characterizes its state as 'likely incomplete' and its rating as 'todo' [2].

Having decided to opt for OMNeT++ as the simulator of choice, this has been a serious drawback in an otherwise excellent simulation tool. We, therefore, studied the relevant code, determined the essential problems that hindered its use in simulation, and wrote the necessary code amendments necessary for the RTP implementation to become functional.

In the rest of this paper we present briefly the problems we identified as well as the solutions we implemented and propose, which are detailed in a separate document submitted for this purpose.

2. RTP IMPLEMENTATION PROBLEMS

The actual RTP implementation in v4.0 of the OMNeT++ package is composed of many files contributed by Matthias Oppitz, Arndt Buschmann, Ahmed Ayadi and Andras Varga [3]. Unfortunately it was not possible to run RTP-based simulations because many errors appear during the respective build phase. We briefly present the problems we identified as well as the proposed solutions.

The first problem appears in 'RTP.ned', where the input gate names were wrongly used by the dependent files. We, therefore, chose to modify these names in a uniform way.

The second problem appears in 'RTP.h', where we added the lines 'int rtp_counter;' and 'simtime_t rtp_pkt_delay;' to be able to count the number of packets and bytes reaching an rtp receiver.

In 'RTP.cc' we changed the gate names as in 'RTP.ned' and initialize and update the values of 'rtp_counter' and 'bytesRcvd'.

In 'RTPProfile.h' we had to modify the code for SSRCGate(uint32 src), because findSSRCGate(uint32 ssrc) searches for an object of type SSRCGate. However, the SSRCGate constructor creates an object with no name, hence it cannot be found during the search. Also, the SSRCGate class was initially defined as of type 'cNamedObject' instead of the correct 'cArray'.

The fifth problem appears in the file 'RTPProfile.cc'. There is an error in the line 'rtpPayloadSender->initialize();', which should be modified to 'rtpPayloadSender->callinitialize();', because we need to initialize not only the module from where we call this, but also all the respective submodules, something which is achieved by the proposed modification. Also, several gate names had to be modified.

The sixth problem appears in the file 'RTPLayer.ned', where several gate names have again to be modified.

The seventh problem appears in 'RTPHost.ned', which is a typical example of a network topology containing hosts capable of RTP traffic generation and consumption. Such hosts represented here as compound modules, cannot be automatically assigned IP addresses from the 'flatNetworkConfigurator'. Therefore, we had to include the '@node' attribute and make some changes in several gate names. Unfortunately, this is not enough, since we need to make the necessary modifications in dependent files that implement 'RTPApplication'. These are outlined below.

In 'RTPApplication.ned' only gate names are modified.

In 'RTPApplication.h' we modified the definition of initialize(), by including the parameter 'int stage'. This is because in compound modules, the initialization should start from simpler modules, run flatNetworkConfigurator for IP assignment, and then initialize RTPApplication. With our parameter, 'initialize' can identify the present initialization stage, whereas our function 'virtual int numInitStages() const{return 4};' makes 'initialize' to run for each stage upto the desired one (here stage 3). Also, the type of variable '_destinationAddress' was changed from IPAddress to IPvXAddress, so that the module IPAddressResolver could be used in the 'RTPApplication.cc' file which accepts the name of a module as a string and converts it to the respective IP address.

Finally, in 'RTPApplication.cc' we included 'IPvXAddress.h' and 'IPAddressResolver.h'. We added 'if (stage!=3) return' in 'initialize()' so that it does nothing until we reach stage 3 of initialization. Also, we added the necessary code for converting the destination name to an ip address and in function 'enterSession' we could not use as an input a parameter of type IPvXAddress as we had decided to do, but of type IPAdress, we included the function 'get4()' which performs the necessary conversion.

After inclusion of the amendments above we were able to perform simulations using RTP without problems. We, therefore, decided to contribute this work for the benefit of everyone interested in using OMNeT++ for RTP related simulations.

3. CONCLUSION

In this paper we discuss the problems related with the RTP implementation as provided in the INET framework for use with OMNeT++. Although our work is related to the v4.0 release of the OMNeT++ package, the reported status of the RTP implementation in the present release (v4.1) is 'likely incomplete'. We, therefore, studied the respective code, identified all the relevant problems and made amendments to the original code, which resulted in a functional RTP implementation that can now be used to run RTP related simulations.

4. REFERENCES

[1] RFC 3550. RTP: A Transport Protocol for Real-Time Applications.

[2] INET Framework Protocol Matrix, 29 October 2010, http://inet.omnetpp.org/index.php?n=Main.Status.

[3] INET Framework for OMNeT++, v4.0. 15 May 2010. http://inet.omnetpp.org.

APPENDIX

Please note that the RTP Amendments Guide, the directory with the Original files and the directory with the Modified files are available online as a single .rar file:

http://users.uom.gr/~pfoul/STUDENTS/BSc/RTP_Corrected_User_Guide_and_Files.rar