RTPS Latency Test

DOCUMENT INFORMATION	
Name	eProsima – RTPS Latency Tests
Product	Documentation
Author	Gonzalo Rodríguez Canosa
Distribution	R 0.5.0
Document Version	1
Date	10-Aug-2014
Status	Draft
Contact	gonzalorodriguez@eprosima.com
Document Code	

eProsima
Proyectos y Sistemas de Mantenimiento SL
Ronda del poniente 2 – 1ºG
28760 Tres Cantos Madrid
Tel: + 34 91 804 34 48
info@eProsima.com – www.eProsima.com

Trademarks

License

Technical Support

- Phone: +34 91 804 34 48
- Email: Support@eProsima.com

Revision history

Date	Author	Description
10 Aug 2014	Gonzalo Rodríguez	First version of the document

Contents

1. Introduction..1

1.1. Purpose..1

1.2. Business Context..1

1.3. Scope..1

1.4. Intended Audience..1

1.5. Reference Material..1

1.6. Document organization..1

2. Test Protocol..3

Figure List

Table List

1 Introduction

1.1 Latency

The latency is usually defined as the amount of time a message takes to traverse a system. In a packet-based network the latency is usually measured either as the one-way latency (the time from the source sending the packet to the destination receiving it) or as the round-trip delay time (the time from source to destination plus the time from the destination back to the source). The latter is more often used since it van be measured from a single point.

In the case of a RTPS communication exchange the latency could be defined as the time it takes a publisher to serialize and send a data message plus the time it takes a matching subscriber to receive and de-serialize it. Applying the same round-trip concept mentioned before, the round-trip latency could be defined as the time it takes a message to be sent by a publisher, received by a subscriber and sent back to the same publisher. An example of this measurement methodology can be observed in Figure 1.1, where the round trip latency can be obtained from T2-T1.

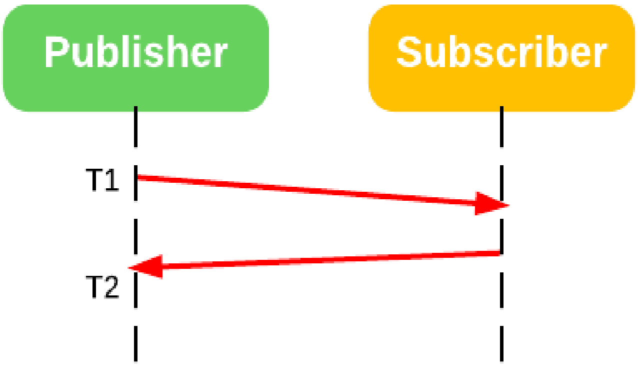

Figure 1.1: One-to-one round trip latency

Latency can also be computed for a multiple subscriber environment, as shown in Figure 1.2, where the round trip latency would be T2-T1.

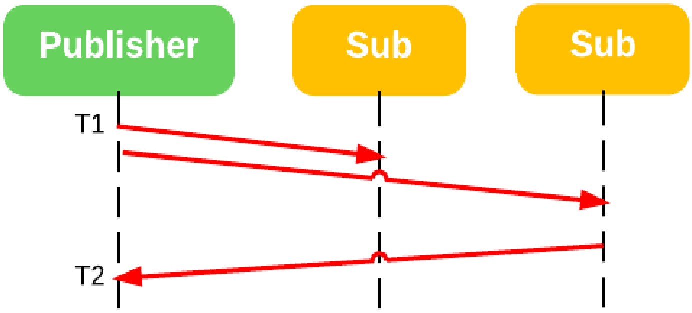

Figure 1.2: One to multiple round trip latency

2 Latency Test

The latency test distributed with this release of eProsima RTPS is develop to be able to determine the round trip latency of different scenarios involving a single publisher and one or multiple subscribers. The following sections will explain the test operation and its results.

2.1 Source Files

The Latency test distributed with this release of ePRosima RTPS consist of the following source files:

- LatencyTestTypes.cpp/.h: Serialization and deserialization method of the two data types used in the test: LatencyType (the one send in the latency test) and CommandType (a specific data used bu the endpoints to communication the beginning of the test).
- LatencyTestPublisher.cpp/.h: This class contains the publisher side of the application. Two DDS publishers and subscribers are created (a pair for the latency data communication and a pair for the commands). This class is also in charge of saving the times and calculating the time statistics.
- LatencyTestSubscriber.cpp/.h: This class contains the subscriber side of the application. As in the publisher side two pairs of publisher/subscriber are created to perform the test.
- main_LatencyTest.cpp: This is the main file of the application that creates a LatencyTestPublisher or a LatencyTestSubscriber depending on the command line options.

2.2 Building the test

A current version of eProsima RTPS library needs to be installed and compiled to be able to build and execute the test.

2.2.1 Unix

The test is provided with a Makefile that should allow the user the compilation in any Unix-based machine, just executing *make* in the home directory of the test. Two executable files are generated: LatencyTest and LatencyTestd, one for the release and one for the debug version of the test. To obtain reliable results the release version should be used. Boost 1.53 needs to be installed on the system.

2.2.2 Windows

The test is provided with a Visual Studio 2010 project, shared with other use tests provided with the release. The test compiles without problem generating two executable files: LatencyTest.exe and LatencyTestd.exe, again the release and debug version. In case the user would want to generate its own project it must be noted that, along with boost 1.53 libraries, the following dependencies are also needed: Shlwapi.lib and Iphlpapi.lib.

2.3 Test operation

The same application file is used in either side of the test (publisher or subscriber). The behavior of the test will depend on the command line arguments provided to the application.

For the best performance of the test the publisher should be launched first followed by all the subscribers. If more subscribers are created than the specified number the test will stop. Also, if multiple publishers are created the behaviour is unpredicted.

2.3.1 Command line options

Three arguments can be passed to the application:

1. “publisher or subscriber”: The first argument indicates the application whether is going to be acting as the publisher side or the subscriber side in this test.
2. The second argument is different depending on the type of endpoint the application is meant to be:
    1. Number of subscribers: The publisher must be informed of the number of subscribers it needs to expect, in order to wait for all of them to be online.
    2. Echo policy: The subscriber must be told whether it should echo back the data or not. In a multiple subscriber scenario only the last subscriber is supposed to return the data to the publisher.
3. Number of samples. The number of samples can also be passed as an argument. If no third argument is passed the default value is 10000.

2.3.2 Command line examples

The following table present two example of command line options for two different test.

One to one test	***Command***
Publisher:	*LatencyTest publisher 1 10000*
Subscriber:	*LatencyTest subscriber echo 10000*
One to multiple test:	
Publisher:	*LatencyTest publisher 2 10000*
Subscriber 1 (executed first):	*LatencyTest subscriber noecho 10000*
Subscriber 2 (executed last):	*LatencyTest subscriber echo 10000*

2.4 Test Results

The results are measured in the publisher side and expressed in microseconds. An example of the obtained results can be seen below:

```
Printing round-trip times in us, statistics for 10000 samples
   Bytes,   stdev,    mean,     min,     50%,     90%,     99%,  99.99%,     max
--------,--------,--------,--------,--------,--------,--------,--------,--------,
      16,   48.00,  101.00,   41.15,   90.15,  183.15,  261.15,  574.15,  589.15
      32,   60.00,  110.00,   37.15,   90.15,  194.15,  277.15, 1090.15, 1134.15
      64,   73.00,  138.00,   37.15,  127.15,  207.15,  300.15, 1168.15, 2823.15
     128,   62.00,  128.00,   38.15,   97.15,  203.15,  283.15,  878.15,  913.15
     256,   54.00,  114.00,   49.15,   92.15,  193.15,  267.15,  542.15,  795.15
     512,   53.00,  114.00,   48.15,   93.15,  196.15,  266.15,  547.15,  736.15
    1024,   50.00,  110.00,   44.15,   95.15,  190.15,  256.15,  818.15,  910.15
    2048,   64.00,  126.00,   38.15,   98.15,  204.15,  293.15, 1306.15, 1975.15
    4096,   70.00,  140.00,   48.15,  108.15,  213.15,  311.15, 2009.15, 2102.15
    8192,   62.00,  174.00,   60.15,  197.15,  234.15,  326.15,  969.15,  973.15
   12288,   61.00,  176.00,   73.15,  193.15,  241.15,  330.15,  924.15,  981.15
```

If the test is carried out in a Windows environment, only the mean time is obtained.