Computer Networks and Distributed Systems - UDP & RMI

Aufar Laksana

February 13th 2017

1 Introduction

The purpose of this coursework was to understand how RMI and UDP work, by coding a server and client for each method.

2 User Datagram Protocol (UDP)

UDP provides an unreliable service. There are no guarantees for the delivery of the messages or prevention for duplication of the data. This reduces the overhead of using the protocol, at the cost of reliability.

By testing the UDP server and client on separate machines in the computer labs, the following results were obtained:

No. Messages Sent	No. Received	No. Lost
10	10	0
100	100	0
200	179	21
250	250	0
350	350	0
500	307	193
1000	790	210
1500	703	797
2000	1143	857

As we can see from the data, initially, from 10 to 350 messages, most of the packets are received by the server. There is one anomalous piece of data at 200 messages. However, when more than 500 messages are sent, the number of lost messages increases.

This may be due to the fact that UDP is very efficient and the lack of overhead means it can send lots of packets very quickly, especially on a LAN. However, due to its lack of congestion control, the packets are easily lost during transmission due to interference. Furthermore, since it does not retransmit any lost messages, there is an increased chance of loss of data.

3 Remote Method Invocation (RMI)

RMI allows an object on a client machine to invoke methods on an object running on a different JVM, in our case, on the server machine.

The *stub* is an object, and acts as a gateway for the client side and represents the remote object. When a caller invokes a method on the *stub* object, it does:

1. Initiates a connection with remote JVM.
2. Writes and transmits parameters to remote JVM.
3. Waits for results.
4. Reads return value.
5. Returns value to caller.

The *skeleton* is an object which acts as a gateway for the server side object. When the *skeleton* receives an incoming request, it does:

1. Reads parameters for the remote method.
2. Invokes method on remote object.
3. Writes and transmits the result to caller.

In order for the client to be able to invoke the remote methods, the server must register/bind the service with the RMI Registry. The client then calls the registry to obtain a reference to the remote object and it will then be able to call its methods. A Security Manager was also implemented for both the server and the client, which is used when RMI downloads code from a remote machine.

When we tested the RMI server and client on different machines in the labs, we obtained:

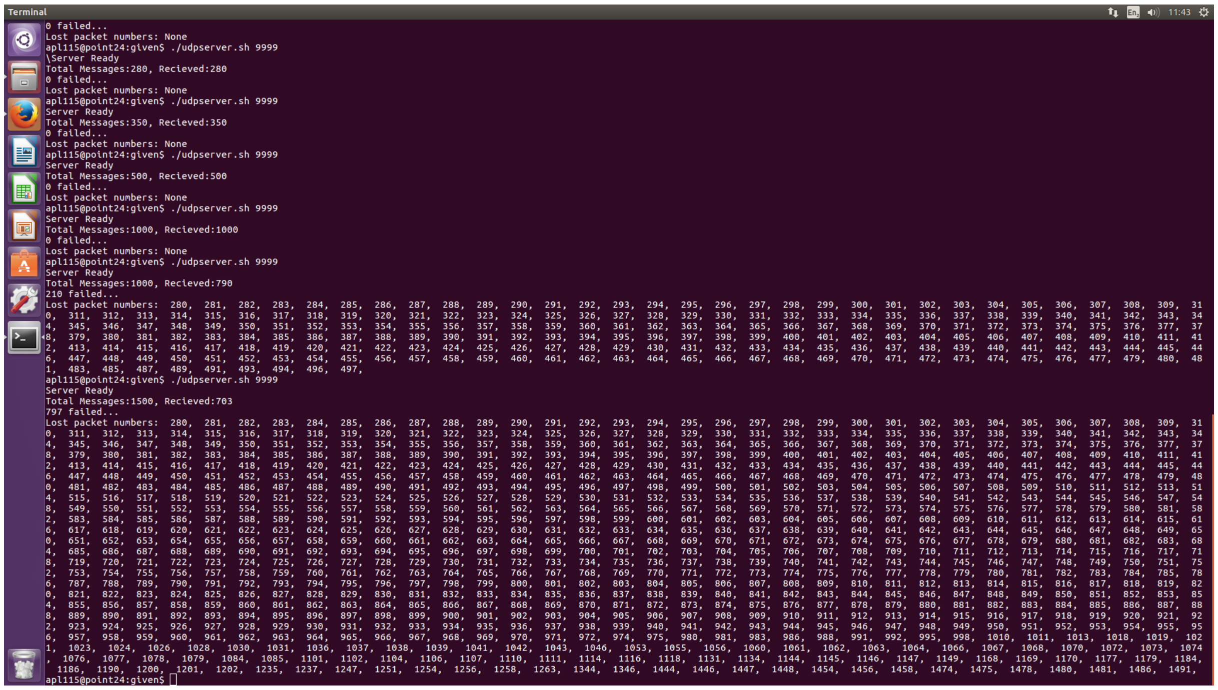
No. Messages Sent	No. Received	No. Lost
100	100	0
300	300	0
500	500	0
1000	1000	0
1500	1500	0
2000	2000	0

As we can see from the results, RMI is very reliable, even when sending thousands of messages, not a single message was lost. However, as the number of messages being sent increased, the time between the client sending the message aand the server confirming it had received all the messages increased. This may be due to the fact that there is a lot of overhead, such as the server binding the service to the RMI Registry and *stub* and *skeleton* structure. There are a lot of steps which must be completed before calling a remote method.

4 Conclusion: RMI vs UDP

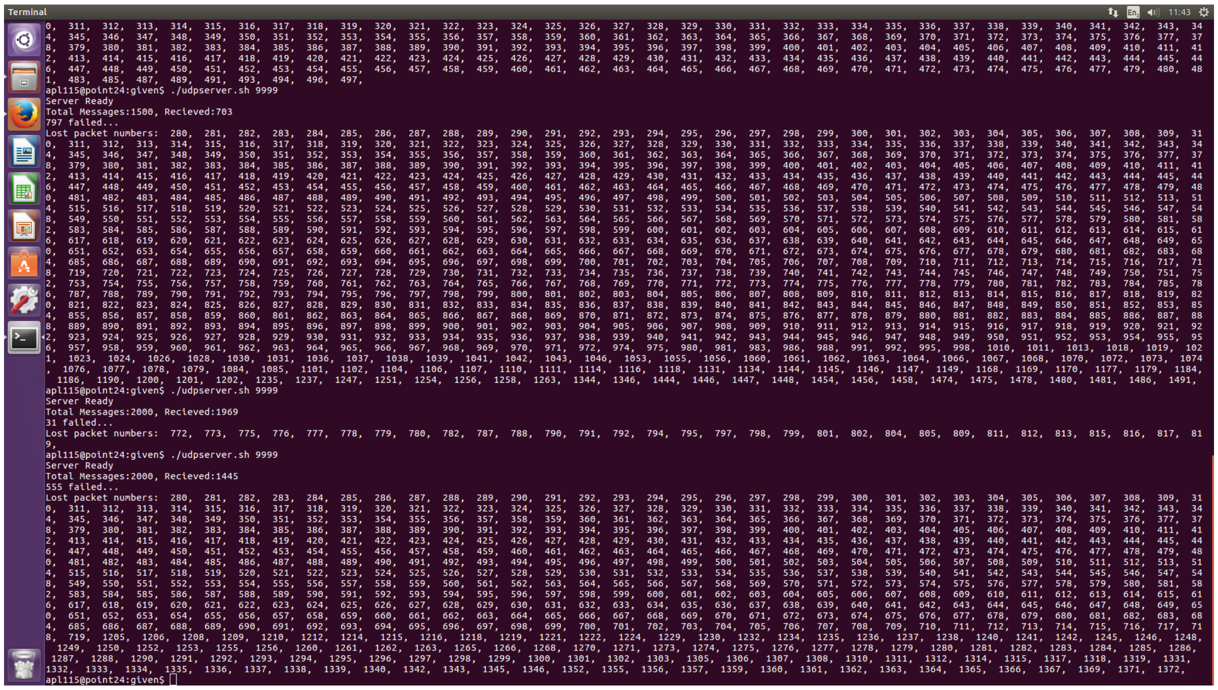
From the tests, we can see that RMI is much more reliable than UDP, since it did not lose a single message. It was also easier to program. However, it is much slower and would not be suitable for a system which requires a lot of exchanging of messages. UDP on the other hand, was much faster, at the cost of reliability, and from the perspective of a new Java programmer, was much harder to implement.

5 Screendumps



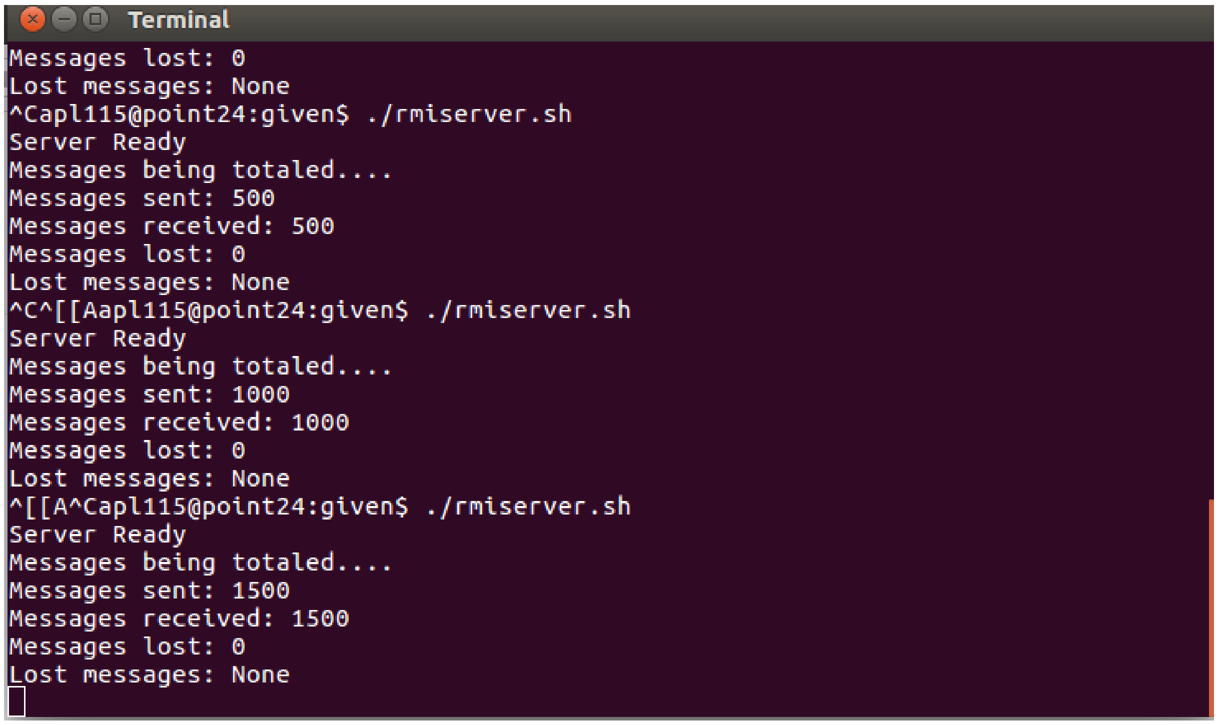
```
0 failed...
Lost packet numbers: None
apl115@point24:given$ ./udpserver.sh 9999
\Server Ready
Total Messages:280, Recieved:280
0 failed...
Lost packet numbers: None
apl115@point24:given$ ./udpserver.sh 9999
Server Ready
Total Messages:350, Recieved:350
0 failed...
Lost packet numbers: None
apl115@point24:given$ ./udpserver.sh 9999
Server Ready
Total Messages:500, Recieved:500
0 failed...
Lost packet numbers: None
apl115@point24:given$ ./udpserver.sh 9999
Server Ready
Total Messages:1000, Recieved:1000
0 failed...
Lost packet numbers: None
apl115@point24:given$ ./udpserver.sh 9999
Server Ready
Total Messages:1000, Recieved:790
210 failed...
Lost packet numbers:  280,  281,  282,  283,  284,  285,  286,  287,  288,  289,  290,  291,  292,  293,  294,  295,  296,  297,  298,  299,  300,  301,  302,  303,  304,  305,  306,  307,  308,  309,  31
0,  311,  312,  313,  314,  315,  316,  317,  318,  319,  320,  321,  322,  323,  324,  325,  326,  327,  328,  329,  330,  331,  332,  333,  334,  335,  336,  337,  338,  339,  340,  341,  342,  343,  34
4,  345,  346,  347,  348,  349,  350,  351,  352,  353,  354,  355,  356,  357,  358,  359,  360,  361,  362,  363,  364,  365,  366,  367,  368,  369,  370,  371,  372,  373,  374,  375,  376,  377,  37
8,  379,  380,  381,  382,  383,  384,  385,  386,  387,  388,  389,  390,  391,  392,  393,  394,  395,  396,  397,  398,  399,  400,  401,  402,  403,  404,  405,  406,  407,  408,  409,  410,  411,  41
2,  413,  414,  415,  416,  417,  418,  419,  420,  421,  422,  423,  424,  425,  426,  427,  428,  429,  430,  431,  432,  433,  434,  435,  436,  437,  438,  439,  440,  441,  442,  443,  444,  445,  44
6,  447,  448,  449,  450,  451,  452,  453,  454,  455,  456,  457,  458,  459,  460,  461,  462,  463,  464,  465,  466,  467,  468,  469,  470,  471,  472,  473,  474,  475,  476,  477,  479,  480,  48
1,  483,  485,  487,  489,  491,  493,  494,  496,  497,
apl115@point24:given$ ./udpserver.sh 9999
Server Ready
Total Messages:1500, Recieved:703
797 failed...
Lost packet numbers:  280,  281,  282,  283,  284,  285,  286,  287,  288,  289,  290,  291,  292,  293,  294,  295,  296,  297,  298,  299,  300,  301,  302,  303,  304,  305,  306,  307,  308,  309,  31
0,  311,  312,  313,  314,  315,  316,  317,  318,  319,  320,  321,  322,  323,  324,  325,  326,  327,  328,  329,  330,  331,  332,  333,  334,  335,  336,  337,  338,  339,  340,  341,  342,  343,  34
4,  345,  346,  347,  348,  349,  350,  351,  352,  353,  354,  355,  356,  357,  358,  359,  360,  361,  362,  363,  364,  365,  366,  367,  368,  369,  370,  371,  372,  373,  374,  375,  376,  377,  37
8,  379,  380,  381,  382,  383,  384,  385,  386,  387,  388,  389,  390,  391,  392,  393,  394,  395,  396,  397,  398,  399,  400,  401,  402,  403,  404,  405,  406,  407,  408,  409,  410,  411,  41
2,  413,  414,  415,  416,  417,  418,  419,  420,  421,  422,  423,  424,  425,  426,  427,  428,  429,  430,  431,  432,  433,  434,  435,  436,  437,  438,  439,  440,  441,  442,  443,  444,  445,  44
6,  447,  448,  449,  450,  451,  452,  453,  454,  455,  456,  457,  458,  459,  460,  461,  462,  463,  464,  465,  466,  467,  468,  469,  470,  471,  472,  473,  474,  475,  476,  477,  478,  479,  48
0,  481,  482,  483,  484,  485,  486,  487,  488,  489,  490,  491,  492,  493,  494,  495,  496,  497,  498,  499,  500,  501,  502,  503,  504,  505,  506,  507,  508,  509,  510,  511,  512,  513,  51
4,  515,  516,  517,  518,  519,  520,  521,  522,  523,  524,  525,  526,  527,  528,  529,  530,  531,  532,  533,  534,  535,  536,  537,  538,  539,  540,  541,  542,  543,  544,  545,  546,  547,  54
8,  549,  550,  551,  552,  553,  554,  555,  556,  557,  558,  559,  560,  561,  562,  563,  564,  565,  566,  567,  568,  569,  570,  571,  572,  573,  574,  575,  576,  577,  578,  579,  580,  581,  58
2,  583,  584,  585,  586,  587,  588,  589,  590,  591,  592,  593,  594,  595,  596,  597,  598,  599,  600,  601,  602,  603,  604,  605,  606,  607,  608,  609,  610,  611,  612,  613,  614,  615,  61
6,  617,  618,  619,  620,  621,  622,  623,  624,  625,  626,  627,  628,  629,  630,  631,  632,  633,  634,  635,  636,  637,  638,  639,  640,  641,  642,  643,  644,  645,  646,  647,  648,  649,  65
0,  651,  652,  653,  654,  655,  656,  657,  658,  659,  660,  661,  662,  663,  664,  665,  666,  667,  668,  669,  670,  671,  672,  673,  674,  675,  676,  677,  678,  679,  680,  681,  682,  683,  68
4,  685,  686,  687,  688,  689,  690,  691,  692,  693,  694,  695,  696,  697,  698,  699,  700,  701,  702,  703,  704,  705,  706,  707,  708,  709,  710,  711,  712,  713,  714,  715,  716,  717,  71
8,  719,  720,  721,  722,  723,  724,  725,  726,  727,  728,  729,  730,  731,  732,  733,  734,  735,  736,  737,  738,  739,  740,  741,  742,  743,  744,  745,  746,  747,  748,  749,  750,  751,  75
2,  753,  754,  755,  756,  757,  758,  759,  760,  761,  762,  763,  764,  765,  766,  767,  768,  769,  770,  771,  772,  773,  774,  775,  776,  777,  778,  779,  780,  781,  782,  783,  784,  785,  78
6,  787,  788,  789,  790,  791,  792,  793,  794,  795,  796,  797,  798,  799,  800,  801,  802,  803,  804,  805,  806,  807,  808,  809,  810,  811,  812,  813,  814,  815,  816,  817,  818,  819,  82
0,  821,  822,  823,  824,  825,  826,  827,  828,  829,  830,  831,  832,  833,  834,  835,  836,  837,  838,  839,  840,  841,  842,  843,  844,  845,  846,  847,  848,  849,  850,  851,  852,  853,  85
4,  855,  856,  857,  858,  859,  860,  861,  862,  863,  864,  865,  866,  867,  868,  869,  870,  871,  872,  873,  874,  875,  876,  877,  878,  879,  880,  881,  882,  883,  884,  885,  886,  887,  88
8,  889,  890,  891,  892,  893,  894,  895,  896,  897,  898,  899,  900,  901,  902,  903,  904,  905,  906,  907,  908,  909,  910,  911,  912,  913,  914,  915,  916,  917,  918,  919,  920,  921,  92
2,  923,  924,  925,  926,  927,  928,  929,  930,  931,  932,  933,  934,  935,  936,  937,  938,  939,  940,  941,  942,  943,  944,  945,  946,  947,  948,  949,  950,  951,  952,  953,  954,  955,  95
6,  957,  958,  959,  960,  961,  962,  963,  964,  965,  966,  967,  968,  969,  970,  971,  972,  974,  975,  980,  981,  983,  986,  988,  991,  992,  995,  998,  1010,  1011,  1013,  1018,  1019,  102
1,  1023,  1024,  1026,  1028,  1030,  1031,  1036,  1037,  1038,  1039,  1041,  1042,  1043,  1046,  1053,  1055,  1056,  1060,  1061,  1062,  1063,  1064,  1066,  1067,  1068,  1070,  1072,  1073,  1074
,  1076,  1077,  1078,  1079,  1084,  1085,  1101,  1102,  1104,  1106,  1107,  1110,  1111,  1114,  1116,  1118,  1131,  1134,  1144,  1145,  1146,  1147,  1149,  1168,  1169,  1170,  1177,  1179,  1184,
  1186,  1190,  1200,  1201,  1202,  1235,  1237,  1247,  1251,  1254,  1256,  1258,  1263,  1344,  1346,  1444,  1446,  1447,  1448,  1454,  1456,  1458,  1474,  1475,  1478,  1480,  1481,  1486,  1491,
apl115@point24:given$ 
```

Figure 1: A console showing transmission of UDP. Some messages are lost.



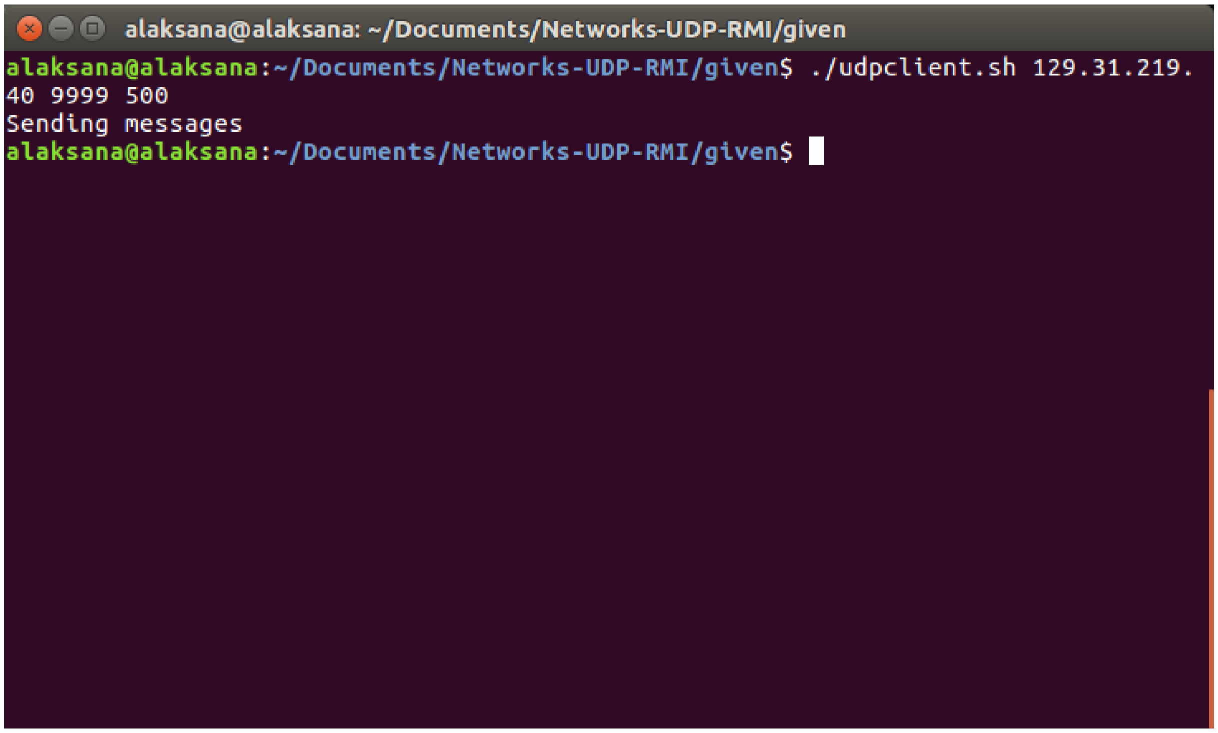
```
0,  311,  312,  313,  314,  315,  316,  317,  318,  319,  320,  321,  322,  323,  324,  325,  326,  327,  328,  329,  330,  331,  332,  333,  334,  335,  336,  337,  338,  339,  340,  341,  342,  343,  34
4,  345,  346,  347,  348,  349,  350,  351,  352,  353,  354,  355,  356,  357,  358,  359,  360,  361,  362,  363,  364,  365,  366,  367,  368,  369,  370,  371,  372,  373,  374,  375,  376,  377,  37
8,  379,  380,  381,  382,  383,  384,  385,  386,  387,  388,  389,  390,  391,  392,  393,  394,  395,  396,  397,  398,  399,  400,  401,  402,  403,  404,  405,  406,  407,  408,  409,  410,  411,  41
2,  413,  414,  415,  416,  417,  418,  419,  420,  421,  422,  423,  424,  425,  426,  427,  428,  429,  430,  431,  432,  433,  434,  435,  436,  437,  438,  439,  440,  441,  442,  443,  444,  445,  44
6,  447,  448,  449,  450,  451,  452,  453,  454,  455,  456,  457,  458,  459,  460,  461,  462,  463,  464,  465,  466,  467,  468,  469,  470,  471,  472,  473,  474,  475,  476,  477,  479,  480,  48
1,  483,  485,  487,  489,  491,  493,  494,  496,  497,
apl115@point24:given$ ./udpserver.sh 9999
Server Ready
Total Messages:1500, Recieved:703
797 failed...
Lost packet numbers:  280,  281,  282,  283,  284,  285,  286,  287,  288,  289,  290,  291,  292,  293,  294,  295,  296,  297,  298,  299,  300,  301,  302,  303,  304,  305,  306,  307,  308,  309,  31
0,  311,  312,  313,  314,  315,  316,  317,  318,  319,  320,  321,  322,  323,  324,  325,  326,  327,  328,  329,  330,  331,  332,  333,  334,  335,  336,  337,  338,  339,  340,  341,  342,  343,  34
4,  345,  346,  347,  348,  349,  350,  351,  352,  353,  354,  355,  356,  357,  358,  359,  360,  361,  362,  363,  364,  365,  366,  367,  368,  369,  370,  371,  372,  373,  374,  375,  376,  377,  37
8,  379,  380,  381,  382,  383,  384,  385,  386,  387,  388,  389,  390,  391,  392,  393,  394,  395,  396,  397,  398,  399,  400,  401,  402,  403,  404,  405,  406,  407,  408,  409,  410,  411,  41
2,  413,  414,  415,  416,  417,  418,  419,  420,  421,  422,  423,  424,  425,  426,  427,  428,  429,  430,  431,  432,  433,  434,  435,  436,  437,  438,  439,  440,  441,  442,  443,  444,  445,  44
6,  447,  448,  449,  450,  451,  452,  453,  454,  455,  456,  457,  458,  459,  460,  461,  462,  463,  464,  465,  466,  467,  468,  469,  470,  471,  472,  473,  474,  475,  476,  477,  478,  479,  48
0,  481,  482,  483,  484,  485,  486,  487,  488,  489,  490,  491,  492,  493,  494,  495,  496,  497,  498,  499,  500,  501,  502,  503,  504,  505,  506,  507,  508,  509,  510,  511,  512,  513,  51
4,  515,  516,  517,  518,  519,  520,  521,  522,  523,  524,  525,  526,  527,  528,  529,  530,  531,  532,  533,  534,  535,  536,  537,  538,  539,  540,  541,  542,  543,  544,  545,  546,  547,  54
8,  549,  550,  551,  552,  553,  554,  555,  556,  557,  558,  559,  560,  561,  562,  563,  564,  565,  566,  567,  568,  569,  570,  571,  572,  573,  574,  575,  576,  577,  578,  579,  580,  581,  58
2,  583,  584,  585,  586,  587,  588,  589,  590,  591,  592,  593,  594,  595,  596,  597,  598,  599,  600,  601,  602,  603,  604,  605,  606,  607,  608,  609,  610,  611,  612,  613,  614,  615,  61
6,  617,  618,  619,  620,  621,  622,  623,  624,  625,  626,  627,  628,  629,  630,  631,  632,  633,  634,  635,  636,  637,  638,  639,  640,  641,  642,  643,  644,  645,  646,  647,  648,  649,  65
0,  651,  652,  653,  654,  655,  656,  657,  658,  659,  660,  661,  662,  663,  664,  665,  666,  667,  668,  669,  670,  671,  672,  673,  674,  675,  676,  677,  678,  679,  680,  681,  682,  683,  68
4,  685,  686,  687,  688,  689,  690,  691,  692,  693,  694,  695,  696,  697,  698,  699,  700,  701,  702,  703,  704,  705,  706,  707,  708,  709,  710,  711,  712,  713,  714,  715,  716,  717,  71
8,  719,  720,  721,  722,  723,  724,  725,  726,  727,  728,  729,  730,  731,  732,  733,  734,  735,  736,  737,  738,  739,  740,  741,  742,  743,  744,  745,  746,  747,  748,  749,  750,  751,  75
2,  753,  754,  755,  756,  757,  758,  759,  760,  761,  762,  763,  764,  765,  766,  767,  768,  769,  770,  771,  772,  773,  774,  775,  776,  777,  778,  779,  780,  781,  782,  783,  784,  785,  78
6,  787,  788,  789,  790,  791,  792,  793,  794,  795,  796,  797,  798,  799,  800,  801,  802,  803,  804,  805,  806,  807,  808,  809,  810,  811,  812,  813,  814,  815,  816,  817,  818,  819,  82
0,  821,  822,  823,  824,  825,  826,  827,  828,  829,  830,  831,  832,  833,  834,  835,  836,  837,  838,  839,  840,  841,  842,  843,  844,  845,  846,  847,  848,  849,  850,  851,  852,  853,  85
4,  855,  856,  857,  858,  859,  860,  861,  862,  863,  864,  865,  866,  867,  868,  869,  870,  871,  872,  873,  874,  875,  876,  877,  878,  879,  880,  881,  882,  883,  884,  885,  886,  887,  88
8,  889,  890,  891,  892,  893,  894,  895,  896,  897,  898,  899,  900,  901,  902,  903,  904,  905,  906,  907,  908,  909,  910,  911,  912,  913,  914,  915,  916,  917,  918,  919,  920,  921,  92
2,  923,  924,  925,  926,  927,  928,  929,  930,  931,  932,  933,  934,  935,  936,  937,  938,  939,  940,  941,  942,  943,  944,  945,  946,  947,  948,  949,  950,  951,  952,  953,  954,  955,  95
6,  957,  958,  959,  960,  961,  962,  963,  964,  965,  966,  967,  968,  969,  970,  971,  972,  974,  975,  980,  981,  983,  986,  988,  991,  992,  995,  998,  1010,  1011,  1013,  1018,  1019,  102
1,  1023,  1024,  1026,  1028,  1030,  1031,  1036,  1037,  1038,  1039,  1041,  1042,  1043,  1046,  1053,  1055,  1056,  1060,  1061,  1062,  1063,  1064,  1066,  1067,  1068,  1070,  1072,  1073,  1074
,  1076,  1077,  1078,  1079,  1084,  1085,  1101,  1102,  1104,  1106,  1107,  1110,  1111,  1114,  1116,  1118,  1131,  1134,  1144,  1145,  1146,  1147,  1149,  1168,  1169,  1170,  1177,  1179,  1184,
  1186,  1190,  1200,  1201,  1202,  1235,  1237,  1247,  1251,  1254,  1256,  1258,  1263,  1344,  1346,  1444,  1446,  1447,  1448,  1454,  1456,  1458,  1474,  1475,  1478,  1480,  1481,  1486,  1491,
apl115@point24:given$ ./udpserver.sh 9999
Server Ready
Total Messages:2000, Recieved:1969
31 failed...
Lost packet numbers:  772,  773,  775,  776,  777,  778,  779,  780,  782,  787,  788,  790,  791,  792,  794,  795,  797,  798,  799,  801,  802,  804,  805,  809,  811,  812,  813,  815,  816,  817,  81
9,
apl115@point24:given$ ./udpserver.sh 9999
Server Ready
Total Messages:2000, Recieved:1445
555 failed...
Lost packet numbers:  280,  281,  282,  283,  284,  285,  286,  287,  288,  289,  290,  291,  292,  293,  294,  295,  296,  297,  298,  299,  300,  301,  302,  303,  304,  305,  306,  307,  308,  309,  31
0,  311,  312,  313,  314,  315,  316,  317,  318,  319,  320,  321,  322,  323,  324,  325,  326,  327,  328,  329,  330,  331,  332,  333,  334,  335,  336,  337,  338,  339,  340,  341,  342,  343,  34
4,  345,  346,  347,  348,  349,  350,  351,  352,  353,  354,  355,  356,  357,  358,  359,  360,  361,  362,  363,  364,  365,  366,  367,  368,  369,  370,  371,  372,  373,  374,  375,  376,  377,  37
8,  379,  380,  381,  382,  383,  384,  385,  386,  387,  388,  389,  390,  391,  392,  393,  394,  395,  396,  397,  398,  399,  400,  401,  402,  403,  404,  405,  406,  407,  408,  409,  410,  411,  41
2,  413,  414,  415,  416,  417,  418,  419,  420,  421,  422,  423,  424,  425,  426,  427,  428,  429,  430,  431,  432,  433,  434,  435,  436,  437,  438,  439,  440,  441,  442,  443,  444,  445,  44
6,  447,  448,  449,  450,  451,  452,  453,  454,  455,  456,  457,  458,  459,  460,  461,  462,  463,  464,  465,  466,  467,  468,  469,  470,  471,  472,  473,  474,  475,  476,  477,  478,  479,  48
0,  481,  482,  483,  484,  485,  486,  487,  488,  489,  490,  491,  492,  493,  494,  495,  496,  497,  498,  499,  500,  501,  502,  503,  504,  505,  506,  507,  508,  509,  510,  511,  512,  513,  51
4,  515,  516,  517,  518,  519,  520,  521,  522,  523,  524,  525,  526,  527,  528,  529,  530,  531,  532,  533,  534,  535,  536,  537,  538,  539,  540,  541,  542,  543,  544,  545,  546,  547,  54
8,  549,  550,  551,  552,  553,  554,  555,  556,  557,  558,  559,  560,  561,  562,  563,  564,  565,  566,  567,  568,  569,  570,  571,  572,  573,  574,  575,  576,  577,  578,  579,  580,  581,  58
2,  583,  584,  585,  586,  587,  588,  589,  590,  591,  592,  593,  594,  595,  596,  597,  598,  599,  600,  601,  602,  603,  604,  605,  606,  607,  608,  609,  610,  611,  612,  613,  614,  615,  61
6,  617,  618,  619,  620,  621,  622,  623,  624,  625,  626,  627,  628,  629,  630,  631,  632,  633,  634,  635,  636,  637,  638,  639,  640,  641,  642,  643,  644,  645,  646,  647,  648,  649,  65
0,  651,  652,  653,  654,  655,  656,  657,  658,  659,  660,  661,  662,  663,  664,  665,  666,  667,  668,  669,  670,  671,  672,  673,  674,  675,  676,  677,  678,  679,  680,  681,  682,  683,  68
4,  685,  686,  687,  688,  689,  690,  691,  692,  693,  694,  695,  696,  697,  698,  699,  700,  701,  702,  703,  704,  705,  706,  707,  708,  709,  710,  711,  712,  713,  714,  715,  716,  717,  71
8,  719,  1205,  1206,  1208,  1209,  1210,  1212,  1214,  1215,  1216,  1218,  1219,  1221,  1222,  1224,  1229,  1230,  1232,  1234,  1235,  1236,  1237,  1238,  1240,  1241,  1242,  1245,  1246,  1248,
  1249,  1250,  1252,  1253,  1255,  1256,  1260,  1261,  1262,  1263,  1265,  1266,  1268,  1270,  1271,  1273,  1274,  1275,  1276,  1277,  1278,  1279,  1280,  1281,  1282,  1283,  1284,  1285,  1286,
 1287,  1288,  1290,  1291,  1292,  1293,  1294,  1295,  1296,  1297,  1298,  1299,  1300,  1301,  1302,  1303,  1305,  1306,  1307,  1308,  1310,  1311,  1312,  1314,  1315,  1317,  1318,  1319,  1331,
1332,  1333,  1334,  1335,  1336,  1337,  1338,  1339,  1340,  1342,  1343,  1345,  1346,  1352,  1355,  1356,  1357,  1359,  1360,  1361,  1362,  1363,  1364,  1365,  1366,  1367,  1369,  1371,  1372,
apl115@point24:given$ 
```

Figure 2: Another console showing loss of messages through UDP



```
Terminal
Messages lost: 0
Lost messages: None
^Capl115@point24:given$ ./rmiserver.sh
Server Ready
Messages being totaled....
Messages sent: 500
Messages received: 500
Messages lost: 0
Lost messages: None
^C^[[Aapl115@point24:given$ ./rmiserver.sh
Server Ready
Messages being totaled....
Messages sent: 1000
Messages received: 1000
Messages lost: 0
Lost messages: None
^[[A^Capl115@point24:given$ ./rmiserver.sh
Server Ready
Messages being totaled....
Messages sent: 1500
Messages received: 1500
Messages lost: 0
Lost messages: None
```

Figure 3: A console showing successful transmission of RMI. No messages lost



```
alaksana@alaksana: ~/Documents/Networks-UDP-RMI/given
alaksana@alaksana:~/Documents/Networks-UDP-RMI/given$ ./udpclient.sh 129.31.219.
40 9999 500
Sending messages
alaksana@alaksana:~/Documents/Networks-UDP-RMI/given$
```

Figure 4: Sending UDP from the client side

6 Program Listings

6.1 RMI Client

```
/*
 * Created on 13-Feb-2017
 */
package rmi;

import java.rmi.Naming;
import java.rmi.NotBoundException;
import java.rmi.RemoteException;
import java.rmi.RMISecurityManager;

import java.rmi.registry.LocateRegistry;
import java.rmi.registry.Registry;

import common.MessageInfo;

public class RMIClient {

        public static void main(String[] args) {
                RMIServerI iRMIServer = null;

                // Check arguments for Server host and number of messages
                if (args.length < 2){
                        System.out.println("Needs 2 arguments: ServerHostName/IPAddress, T
                        System.exit(-1);
                }

                String urlServer = new String("rmi://" + args[0] + "/RMIServer");
                int numMessages = Integer.parseInt(args[1]);

                // TO-DO: Initialise Security Manager
                System.setSecurityManager(new RMISecurityManager());

                try{    // TO-DO: Bind to RMIServer

                        //Registry reg = LocateRegistry.getRegistry(args[0]);
                        iRMIServer = (RMIServerI) Naming.lookup(urlServer);
                        // Attempt to send messages the specified number of times
                        for(int i = 0; i < numMessages; i++) {
                         MessageInfo msg = new MessageInfo(numMessages, i);
                         iRMIServer.receiveMessage(msg);
                        }
                }
                catch(Exception e){
                        System.out.println("err: " + e.getMessage());
            e.printStackTrace();
                }
        }
}
```

6.2 RMI Server

```
/*
 * Created on 13-Feb-2017
 */
package rmi;

import java.net.MalformedURLException;
import java.rmi.Naming;
import java.rmi.registry.LocateRegistry;
import java.rmi.RemoteException;
import java.rmi.server.UnicastRemoteObject;
import java.util.Arrays;

import java.rmi.registry.Registry;

import common.*;

public class RMIServer extends UnicastRemoteObject implements RMIServerI {

        private int totalMessages = -1;
        private int[] receivedMessages;

        public RMIServer() throws RemoteException {
        }

        public void receiveMessage(MessageInfo msg) throws RemoteException {

                // TO-DO: On receipt of first message, initialise the receive buffer

                // TO-DO: Log receipt of the message

                // TO-DO: If this is the last expected message, then identify
                //        any missing messages

                if(receivedMessages == null){
                        totalMessages = msg.totalMessages;
                        receivedMessages = new int[msg.totalMessages];
                }
                // TO-DO: Log receipt of the message
                receivedMessages[msg.messageNum] = 1;

                if (msg.messageNum + 1 == totalMessages) {
                        System.out.println("Messages being totaled....");

                        String lostmes = "Lost messages: ";
                        int count = 0;
                        for (int i = 0; i < totalMessages; i++) {
                                if (receivedMessages[i] != 1) {
                                        count++;
                                        lostmes = lostmes + " " + (i+1) + ", ";
```

```
                    }
                }

                if (count == 0) {
                        lostmes = lostmes + "None";
                }
                System.out.println("Messages sent: " + totalMessages);
                System.out.println("Messages received: " + (totalMessages - count)
                System.out.println("Messages lost: " + count);
                System.out.println(lostmes);
                }
        }

        public static void main(String[] args) {

                System.setProperty("RMIServer","129.31.219.40");

                RMIServer rmis = null;

                // TO-DO: Initialise Security Manager
                if (System.getSecurityManager() == null) {
                System.setSecurityManager(new SecurityManager());
                }

                try{
                        rmis = new RMIServer();
                        //RMIServerI stub = (RMIServerI) UnicastRemoteObject.exportObject(

                        rebindServer("RMIServer", rmis);

                        System.out.println("Server Ready");
                }
                catch(Exception e){
            System.out.println("RMIServer err: " + e.getMessage());
            e.printStackTrace();
                }

                // TO-DO: Instantiate the server class

                // TO-DO: Bind to RMI registry

        }

        protected static void rebindServer(String serverURL, RMIServer server) {

                // TO-DO:
                // Start / find the registry (hint use LocateRegistry.createRegistry(...)
                // If we *know* the registry is running we could skip this (eg run rmiregi
                try{
                        LocateRegistry.createRegistry(8080);
                        Naming.rebind(serverURL, server);
                }
                catch(Exception e){
                        System.out.println("RMIServer err: " + e.getMessage());
                        e.printStackTrace();
                }

                // TO-DO:
                // Now rebind the server to the registry (rebind replaces any existing ser
                // Note - Registry.rebind (as returned by createRegistry / getRegistry) do
                // expects different things from the URL field.

        }
}
```

6.3 UDP Client

```
package udp;


import java.io.IOException;
import java.io.ObjectOutputStream;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.net.SocketException;
import java.net.UnknownHostException;
import java.io.BufferedOutputStream;
import java.io.ByteArrayOutputStream;

import common.MessageInfo;

public class UDPClient {

        private DatagramSocket sendSoc;

        public static void main(String[] args) {
                InetAddress     serverAddr = null;
                int                     recvPort;
                int             countTo;
                String          message;

                // Get the parameters
                if (args.length < 3) {
                        System.err.println("Arguments␣required:␣server␣name/IP,␣recv␣port,
                        System.exit(-1);
                }

                try {
                        serverAddr = InetAddress.getByName(args[0]);
                } catch (UnknownHostException e) {
                        System.out.println("Bad␣server␣address␣in␣UDPClient,␣" + args[0] +
                        System.exit(-1);
                }
                recvPort = Integer.parseInt(args[1]);
                countTo = Integer.parseInt(args[2]);


                // TO-DO: Construct UDP client class and try to send messages
                UDPClient client = new UDPClient();
                System.out.println("Sending␣messages");
                client.testLoop(serverAddr, recvPort, countTo);
        }

        public UDPClient() {
                // TO-DO: Initialise the UDP socket for sending data
        try {
                sendSoc = new DatagramSocket();
        } catch (SocketException e) {
                System.out.println("Error creating socket for sending data.");
        }


    }

    private void testLoop(InetAddress serverAddr, int recvPort, int countTo) {
        MessageInfo m;
        ByteArrayOutputStream byteStream;
        ObjectOutputStream os;
        // TO-DO: Send the messages to the server
        for(int i = 0; i < countTo; i++) {
               m = new MessageInfo(countTo,i);

                byteStream = new ByteArrayOutputStream(5000);
                try {
                        os = new ObjectOutputStream(new BufferedOutputStream(byteSt
                        os.flush();
                        os.writeObject(m);
                        os.flush();
                } catch (IOException e) {
                        System.out.println("Error serializing object");
                        System.exit(-1);
                }

                //retrieves byte array
                byte[] sendBuf = byteStream.toByteArray();
                send(sendBuf, serverAddr, recvPort);


        }
    }

    private void send(byte[] data, InetAddress destAddr, int destPort) {
        DatagramPacket          pkt;

        // TO-DO: build the datagram packet and send it to the server
        pkt = new DatagramPacket(data, data.length, destAddr, destPort);
        try {
                sendSoc.send(pkt);
        } catch (IOException e) {
                System.out.println("Err transmitting packet");
                System.exit(-1);
        }
    }
}
```

6.4 UDP Server

```
package udp;


import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.SocketException;
import java.net.SocketTimeoutException;
import java.io.*;
import common.MessageInfo;


public class UDPServer {
        private DatagramSocket socket;
        private int port;
        private int totalMessages = -1;
        private int[] receivedMessages;
        private boolean close;

        private void run() throws SocketTimeoutException {
                int                             pacSize;
                byte[]                  pacData;
                DatagramPacket  packet;

                System.out.println("Server Ready");
                while(!close){

                    //Receive request from client
                        pacSize = 5000;
                        pacData = new byte[5000];

                        packet = new DatagramPacket(pacData, pacSize);
                        try {
                          socket.setSoTimeout(10000);
                          socket.receive(packet);
                        } catch (IOException e) {
                                System.out.println("Error IO exception receiving packet.")
                                System.exit(-1);
                        }

                        processMessage(packet.getData());

                }

        }

        public void processMessage(byte[] data) {

                MessageInfo msg = null;
```

```
            // Use the data to construct a new MessageInfo object
        ByteArrayInputStream byteStream = new ByteArrayInputStream(data);
        ObjectInputStream is;

            try {
                    is = new ObjectInputStream(new BufferedInputStream(byteStream));
                    msg = (MessageInfo) is.readObject();
                    is.close();
            } catch (ClassNotFoundException e) {
                    System.out.println("Could␣not␣find␣class␣match␣");
            } catch (IOException e) {
                    System.out.println("IOexception:␣ObjectInputStream.");
            }

            // On receipt of first message, initialize the receive buffer
            if (receivedMessages == null) {
                    totalMessages = msg.totalMessages;
                    receivedMessages = new int[totalMessages];
            }

            // Log receipt of the message
            receivedMessages[msg.messageNum] = 1;

            // If this is the last expected message, then identify
            //         any missing messages
            if (msg.messageNum + 1 == msg.totalMessages) {
                    close = true;

                    String lostmes = "Lost␣packet␣numbers:␣";
                    int count = 0;
                    for (int i = 0; i < totalMessages; i++) {
                            if (receivedMessages[i] != 1) {
                                    count++;
                                    lostmes = lostmes + "␣" + (i+1) + ",␣";
                            }
                    }

                    if (count == 0){
                    lostmes = lostmes + "None";
                    }

                    System.out.println("Total␣Messages:" + msg.totalMessages + ",␣Recie
                    System.out.println(count + "␣failed");
                    System.out.println(lostmes);
            }


    }
    public UDPServer(int rp) {
            // Initialize UDP socket for receiving data
            try {
                    port = rp;
                    socket = new DatagramSocket(port);
            } catch (SocketException e) {
                    System.out.println("Error: Could not create socket on port " + por
                    System.exit(-1);
            }
            // Make it so the server can run.
            close = false;
    }

    public static void main(String args[]) {
            int         recvPort;

            // Get the parameters from command line
            if (args.length < 1) {
                    System.err.println("Error: Arguments required - recv-port");
                    System.exit(-1);
            }
            recvPort = Integer.parseInt(args[0]);

            // Initialize Server object and start it by calling run()
            UDPServer udpsrv = new UDPServer(recvPort);
            try {
                    udpsrv.run();
            } catch (SocketTimeoutException e) {}
    }

}
```