Network Working Group
Request for Comments: 2762
Category: Experimental

J. Rosenberg
dynamicsoft
H. Schulzrinne
Columbia U.

February 2000

Sampling of the Group Membership in RTP

Status of this Memo

Copyright Notice

Abstract

In large multicast groups, the size of the group membership table maintained by RTP (Real Time Transport Protocol) participants may become unwieldy, particularly for embedded devices with limited memory and processing power. This document discusses mechanisms for sampling of this group membership table in order to reduce the memory requirements. Several mechanisms are proposed, and the performance of each is considered.

1 Introduction

RTP, the Real Time Transport Protocol [1], mandates that RTCP packets be transmitted from each participant with a period roughly proportional to the group size. The group size is obtained by storing a table, containing an entry for each unique SSRC seen in RTP and RTCP packets. As members leave or time out, entries are deleted, and as new members join, entries are added. The table is thus highly dynamic.

For large multicast sessions, such as an mbone broadcast or IP-based TV distribution, group sizes can be extremely large, on the order of hundreds of thousands to millions of participants. In these environments, RTCP may not always be used, and thus the group membership table isn't needed. However, it is highly desirable for RTP to scale well for groups with one member to groups with one million members, without human intervention to "turn off" RTCP when it's no longer appropriate. This means that the same tools and

systems can be used for both small conferences and TV broadcasts in a smooth, scalable fashion.

Previous work [2] has identified three major scalability problems with RTP. These are:

1. Congestion due to floods of RTCP packets in highly dynamic groups;

2. Large delays between receipt of RTCP packets from a single user;

3. Large size of the group membership table.

The reconsideration algorithm [2] helps to alleviate the first of these. This document addresses the third, that of large group size tables.

Storage of an SSRC table with one million members, for example, requires at least four megabytes. As a result, embedded devices with small memory capacity may have difficulty under these conditions. To solve this problem, SSRC sampling has been proposed. SSRC sampling uses statistical sampling to obtain a stochastic estimate of the group membership. There are many issues that arise when this is done. This document reviews these issues and discusses the mechanisms which can be applied by implementors. In particular, it focuses on three methods for adapting the sampling probability as the group membership varies. It is important to note that the IETF has been notified of intellectual property rights claimed in regard to some or all of the specification contained in this document, and in particular to one of the three mechanisms: the binning algorithm described below. For more information consult the online list of claimed rights. The two other approaches presented are inferior to the binning algorithm, but are included as they are believed to be unencumbered by IPR.

2 Basic Operation

The basic idea behind SSRC sampling is simple. Each participant maintains a key K of 32 bits, and a mask M of 32 bits. Assume that m of the bits in the mask are 1, and the remainder are zero. When an RTCP packet arrives with some SSRC S, rather than placing it in the table, it is first sampled. The sampling is performed by ANDing the key and the mask, and also ANDing the SSRC and the mask. The resulting values are compared. If equal, the SSRC is stored in the table. If not equal, the SSRC is rejected, and the packet is treated as if it has never been received.

The key can be anything, but is usually derived from the SSRC of the user who is performing the sampling.

This sampling process can be described mathematically as:

D = (K*M == S*M)

Where the * operator denotes AND and the == operator denotes a test for equality. D represents the sampling decision.

According to the RTP specification, the SSRC's used by session participants are chosen randomly. If the distribution is also uniform, it is easy to see that the above filtering will cause 1 out of 2**m SSRC's to be placed in the table, where m is the number of bits in the mask, M, which are one. Thus, the sampling probability p is 2**-m.

Then, to obtain an actual group size estimate, L, the number of entries in the table N is multiplied by 2**m:

L = N * 2**m

Care must be taken when choosing which bits to set to 1 in the mask. Although the RTP specification mandates randomly chosen SSRC, there are many known implementations which do not conform to this. In particular, the ITU H.323 [3] series of recommendations allows the central control element, the gatekeeper, to assign the least significant 8 bits of the SSRC, while the most significant are randomly chosen by RTP participants.

The safest way to handle this problem is to first hash the SSRC using a cryptographically secure hash, such as MD5 [4], and then choose 32 of the bits in the result as the SSRC used in the above computation. This provides much better randomness, and doesn't require detailed knowledge about how various implementations actually set the SSRC.

2.1 Performance

The estimate is more accurate as the value of m decreases, less accurate as it increases. This can be demonstrated analytically. If the actual group size is G, the ratio of the standard deviation to mean of the estimate L (coefficient of variation) is:

sqrt((2**m - 1)/G)

This equation can be used as a guide for selecting the thresholds for when to change the sampling factor, as discussed below. For example, if the target is a 1% standard deviation to mean, the sampling

probability p=2**-m should be no smaller than .5 when there are ten thousand group members. More generally, to achieve a desired standard deviation to mean ratio of T, the sampling probability should be no less than:

p > 1 / (1 + G*(T**2))

3 Increasing the Sampling Probability

The above simple sampling procedure would work fine if the group size was static. However, it is not. A participant joining an RTP session will initially see just one participant (themselves). As packets are received, the group size as seen by that participant will increase. To handle this, the sampling probability must be made dynamic, and will need to increase and decrease as group sizes vary.

The procedure for increasing the sampling probability is easy. A participant starts with a mask with m=0. Under these conditions, every received SSRC will be stored in the table, so there is effectively no sampling. At some point, the value of m is increased by one. This implies that approximately half of the SSRC already in the table will no longer match the key under the masking operation. In order to maintain a correct estimate, these SSRC must be discarded from the table. New SSRC are only added if they match the key under the new mask.

The decision about when to increase the number of bits in the mask is also simple. Let’s say an RTP participant has a memory with enough capacity to store C entries in the table. The best estimate of the group is obtained by the largest sampling probability. This also means that the best estimate is obtained the fuller the table is. A reasonable approach is therefore to increase the number of bits in the mask just as the table fills to C. This will generally cause its contents to be reduced by half on average. Once the table fills again, the number of bits in the mask is further increased.

4 Reducing the Sampling Probability

If the group size begins to decrease, it may be necessary to reduce the number of one bits in the mask. Not doing so will result in extremely poor estimates of the group size. Unfortunately, reducing the number of bits in the mask is more difficult than increasing them.

When the number of bits in the mask increases, the user compensates by removing those SSRC which no longer match. When the number of bits decreases, the user should theoretically add back those users whose SSRC now match. However, these SSRC are not known, since the whole

point of sampling was to not have to remember them. Therefore, if the number of bits in the mask is just reduced without any changes in the membership table, the group estimate will instantly drop by exactly half.

To compensate for this, some kind of algorithm is needed. Two approaches are presented here: a corrective-factor solution, and a binning solution. The binning solution is simpler to understand and performs better. However, we include a discussion of the corrective-factor solution for completeness and comparison, and also because it is believed to be unencumbered by IPR.

4.1 Corrective Factors

The idea with the corrective factors is to take one of two approaches. In the first, a corrective factor is added to the group size estimate, and in the second, the group size estimate is multiplied by a corrective factor. In both cases, the purpose is to compensate for the change in sample mask. The corrective factors should decay as the "fudged" members are eventually learned about and actually placed in the membership list.

The additive factor starts at the difference between the group size estimate before and after the number of bits in the mask is reduced, and decays to 0 (this is not always half the group size estimate, as the corrective factors can be compounded, see below). The multiplicative corrective factor starts at 2, and gradually decays to one. Both factors decay over a time of cL(ts-), where c is the average RTCP packet size divided by the RTCP bandwidth for receivers, and L(ts-) is the group size estimate just before the change in the number of bits in the mask at time ts. The reason for this constant is as follows. In the case where the actual group membership has not changed, those members which were forgotten will still be sending RTCP packets. The amount of time it will take to hear an RTCP packet from each of them is the average RTCP interval, which is cL(ts-). Therefore, by cL(ts-) seconds after the change in the mask, those users who were fudged by the corrective factor should have sent a packet and thus appear in the table. We chose to decay both functions linearly. This is because the rate of arrival of RTCP packets is linear.

What happens if the number of bits in the mask is reduced once again before the previous corrective factor has expired? In that case, we compound the factors by using yet another one. Let fi() represent the ith additive correction function, and gi() the ith multiplicative correction function. If ts is the time when the number of bits in the mask is reduced, we can describe the additive correction factor as:

$$fi(t) = \begin{cases} 0 & , \quad t < ts \\ (L(ts-) - L(ts+)) \dfrac{ts + cL(ts-) - t}{cL(ts-)} & , \quad ts < t < ts+cL(ts-) \\ 0 & , \quad t > ts + cL(ts-) \end{cases}$$

and the multiplicative factor as:

$$gi(t) = \begin{cases} 1 & , t < ts \\ \dfrac{ts + 2cL(ts-) - t}{cL(ts-)} & , ts < t < ts + cL(ts-) \\ 1 & , t > ts + cL(ts-) \end{cases}$$

Note that in these equations, L(t) denotes the group size estimate obtained including the corrective factors except for the new factor. ts- is the time right before the reduction in the number of bits, and ts+ the time after. As a result, L(ts-) represents the group size estimate before the reduction, and L(ts+) the estimate right after, but not including the new factor.

Finally, the actual group size estimate L(t) is given by:

$$L(t) = \sum_i fi(t) + N*(2**m)$$

for the additive factor, and:

$$L(t) = \prod_i N*(2**m)*gi(t)$$

for the multiplicative factor.

Simulations showed that both algorithms performed equally well, but both tended to seriously underestimate the group size when the group membership was rapidly declining [5]. This is demonstrated in the performance data below.

As an example, consider computation of the additive factor. The group size is 1000, c is 1 second, and m is two. With a mask of this size, a participant will, on average, observe 250 (N = 250) users. At t=0, the user decides to reduce the number of bits in the mask to 1. As a result, L(0-) is 1000, and L(0+) is 500. The additive factor therefore starts at 500, and decays to zero at time ts + cL(ts-) = 1000. At time 500, lets assume N has increased to 375 (this will, on average, be the case if the actual group size has not changed). At time 500, the additive factor is 250. This is added to 2**m times N, which is 750, resulting in a group size estimate of 1000. Now, the user decides to reduce the number of bits in the mask again, so that m=0. Another additive factor is computed. This factor starts at L(ts-) (which is 1000), minus L(ts+). L(ts+) is computed without the new factor; it is the first additive factor at this time (250) plus 2**m (1) times N (375). This is 625. As a result, the new additive factor starts at 1000 - 625 (375), and decays to 0 in 1000 seconds.

4.2 Binning Algorithm

In order to more correctly estimate the group size even when it is rapidly decreasing, a binning algorithm can be used. The algorithm works as follows. There are 32 bins, same as the number of bits in the sample mask. When an RTCP packet from a new user arrives whose SSRC matches the key under the masking operation, it is placed in the mth bin (where m is the number of ones in the mask) otherwise it is discarded.

When the number of bits in the mask is to be increased, those members in the bin who still match after the new mask are moved into the next higher bin. Those who don't match are discarded. When the number of bits in the mask is to be decreased, nothing is done. Users in the various bins stay where they are. However, when an RTCP packet for a user shows up, and the user is in a bin with a higher value than the current number of bits in the mask, it is moved into the bin corresponding to the current number of bits in the mask. Finally, the group size estimate L(t) is obtained by:

$$L(t) = \sum_{i=0}^{31} B(i) * 2^{**}i$$

Where B(i) are the number of users in the ith bin.

The algorithm works by basically keeping the old estimate when the number of bits in the mask drops. As users arrive, they are gradually moved into the lower bin, reducing the amount that the higher bin contributes to the total estimate. However, the old estimate is still updated in the sense that users which timeout are removed from the higher bin, and users who send BYE packets are also removed from the higher bin. This allows the older estimate to still adapt, while gradually phasing it out. It is this adaptation which makes it perform much better than the corrective algorithms. The algorithm is also extremely simple.

4.3 Comparison

The algorithms are all compared via simulation in Table 1. In the simulation, 10,001 users join a group at t=0. At t=10,000, 5000 of them leave. At t=20,000, another 5000 leave. All implement an SSRC sampling algorithm, unconditional forward reconsideration and BYE reconsideration. The table depicts the group size estimate from time 20,000 to time 25,000 as seen by the single user present throughout the entire session. In the simulation, a memory size of 1000 SSRC was assumed. The performance without sampling, and with sampling with the additive, multiplicative, and bin-based correction are depicted.

As the table shows, the bin based algorithm performs particularly well at capturing the group size estimate towards the tail end of the simulation.

Time	No Sampling	Binned	Additive	Multiplicative
20000	5001	5024	5024	5024
20250	4379	4352	4352	4352
20500	3881	3888	3900	3853
20750	3420	3456	3508	3272
21000	3018	2992	3100	2701
21250	2677	2592	2724	2225
21500	2322	2272	2389	1783
21750	2034	2096	2125	1414
22000	1756	1760	1795	1007
22250	1476	1472	1459	582
22500	1243	1232	1135	230
22750	1047	1040	807	80
23000	856	864	468	59
23250	683	704	106	44
23500	535	512	32	32
23750	401	369	24	24
24000	290	257	17	17
24250	198	177	13	13
24500	119	129	11	11
24750	59	65	8	8
25000	18	1	2	2

4.4 Sender Sampling

Care must be taken in handling senders when using SSRC sampling. Since the number of senders is generally small, and they contribute significantly to the computation of the RTCP interval, sampling should not be applied to them. However, they must be kept in a separate table, and not be "counted" as part of the general group membership. If they are counted as part of the general group membership, and are not sampled, the group size estimate will be inflated to overemphasize the senders.

This is easily demonstrated analytically. Let Ns be the number of senders, and Nr be the number of receivers. The membership table will contain all Ns senders and (1/2)**m of the receivers. The total group size estimate in the current memo is obtained by 2**m times the number of entries in the table. Therefore, the group size estimate becomes:

L(t) = (2**m) Ns + Nr

which exponentially weights the senders.

This is easily compensated for in the binning algorithm. A sender is always placed in the 0th bin. When a sender becomes a receiver, it is moved into the bin corresponding to the current value of m, if its SSRC matches the key under the masked comparison operation.

5 Security Considerations

The use of SSRC sampling does not appear to introduce any additional security considerations beyond those described in [1]. In fact, SSRC sampling, as described above, can help somewhat in reducing the effect of certain attacks.

RTP, when used without authentication of RTCP packets, is susceptible to a spoofing attack. Attackers can inject many RTCP packets into the group, each with a different SSRC. This will cause RTP participants to believe the group membership is much higher than it actually is. The result is that each participant will end up transmitting RTCP packets very infrequently, if ever. When SSRC sampling is used, the problem can be amplified if a participant is not applying a hash to the SSRC before matching them against their key. This is because an attacker can send many packets, each with different SSRC, that match the key. This would cause the group size to inflate exponentially. However, with a random hash applied, an attacker cannot guess those SSRC which will match against the key. In fact, an attacker will have to send 2**m different SSRC before finding one that matches, on average. Of course, the effect of a match causes an increase of group membership by 2**m. But, the use of sampling means that an attacker will have to send many packets before an effect can be observed.

6 Acknowledgements

The authors wish to thank Bill Fenner and Vern Paxson for their comments.

7 Bibliography

[1] Schulzrinne, H., Casner, S., Frederick, R. and V. Jacobson, "RTP: a transport protocol for real-time applications", RFC 1889, January 1996.

[2] J. Rosenberg and H. Schulzrinne, "Timer reconsideration for enhanced RTP scalability", IEEE Infocom, (San Francisco, California), March/April 1998.

[3] International Telecommunication Union, "Visual telephone systems and equipment for local area networks which provide a non-guaranteed quality of service," Recommendation H.323, Telecommunication Standardization Sector of ITU, Geneva, Switzerland, May 1996.

[4] Rivest, R., "The MD5 message-digest algorithm", RFC 1321, April 1992.

[5] Rosenberg, J., "Protocols and Algorithms for Supporting Distributed Internet Telephony," PhD Thesis, Columbia University, Dec. 1999. Work in Progress.

8 Authors' Addresses

Jonathan Rosenberg
dynamicsoft
200 Executive Drive
West Orange, NJ 07052
USA

EMail: jdrosen@dynamicsoft.com

Henning Schulzrinne
Columbia University
M/S 0401
1214 Amsterdam Ave.
New York, NY 10027-7003
USA

EMail: schulzrinne@cs.columbia.edu

9 Full Copyright Statement

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.