Publication II

Jouni Mäenpää and Gonzalo Camarillo. Analysis of Delays in a Peer-to-Peer Session Initiation Protocol Overlay Network. In *7th IEEE Consumer Communications and Networking Conference (CCNC)*, Las Vegas, USA, pp. 1-6, January 2010.

Analysis of Delays in a Peer-to-Peer Session Initiation Protocol Overlay Network

Jouni Mäenpää and Gonzalo Camarillo
Ericsson
Finland
{jouni.maenpaa, gonzalo.camarillo}@ericsson.com

***Abstract*—Peer-to-Peer Session Initiation Protocol (P2PSIP) is a new decentralized person-to-person communication system that is currently being standardized in the Internet Engineering Task Force (IETF). P2PSIP uses the Session Initiation Protocol (SIP) to enable real-time communication in a peer-to-peer environment. The underlying lookup mechanism is implemented using a Distributed Hash Table (DHT). In this paper, we study delays associated with joining, leaving, and initiating calls in a P2PSIP system through experiments in PlanetLab. We also compare the performance of P2PSIP and traditional client-server SIP.**

I. Introduction

The Session Initiation Protocol (SIP) [1] is traditionally used in architectures that have a fixed hierarchy of SIP proxies and user agents. These architectures use the client-server paradigm and employ a centralized SIP proxy-registrar server for every domain. In contrast, P2PSIP uses SIP in an environment where traditional proxy-registrar and message routing functions are replaced by a P2P overlay network. The overlay is organized using a Distributed Hash Table (DHT) algorithm.

In this paper, we study the performance of P2PSIP through experiments in PlanetLab. The focus is on delays associated with joining, leaving, and establishing calls in a P2PSIP overlay. We also compare these delays to those in a client/server SIP system. The goal is to gain a better understanding of the costs of using a decentralized person-to-person communication system. The paper is structured as follows. Section II gives an introduction to our prototype. Section III presents related work. Section IV describes the experiments and the traffic model used. Section V presents the results of the experiments. Section VI concludes the paper.

II. P2PSIP prototype

Our P2PSIP prototype is implemented in the Java programming language. It uses Peer-to-Peer Protocol (P2PP) [2] as the protocol between the peers in the overlay. P2PP is used to exchange overlay maintenance messages and to store and retrieve data. The RELOAD peer protocol that is currently being standardized in the IETF is based on P2PP. P2PP connections run over TCP. SIP over UDP is used as the call control protocol. SIP uses the P2PSIP overlay as a lookup mechanism to map SIP address-of-record values to contact URIs. To organize the overlay, the prototype uses the Chord DHT algorithm [3]. Chord was chosen since the P2PSIP working group specifies it as mandatory to implement [4].

A. Chord

Chord [3] is a structured P2P algorithm that uses consistent hashing to build a DHT out of several peers. Consistent hashing assigns each peer and key an *m*-bit identifier using SHA-1 as the base hash function. The keys are ordered on an identifier circle of size 2^m, which is called the Chord ring.

On the Chord ring, each peer maintains a routing table with up to *m* entries, called the finger table. In an *N*-node network, each peer maintains information about $O(\log N)$ peers in its finger table. Each peer also maintains another data structure, called the successor list, which contains the peer's immediate successors on the Chord ring. In our Chord implementation, the size of the successor list is set to $\log N$. In addition to the successor list, each peer also maintains a predecessor list containing the three immediate predecessors of the peer on the Chord ring. Although the original Chord algorithm does not use a predecessor list, we chose to do so because this was observed to increase the stability of the overlay.

To ensure that the contents of the finger table, predecessor list, and successor list stay up-to-date with the constantly changing topology of the overlay, each peer runs a stabilization routine in the background periodically. As part of the stabilization routine, a peer synchronizes its predecessor list with its first predecessor and its successor list with its first successor, and incorporates new peers into its finger table.

B. P2PSIP operations

To join the P2PSIP overlay, a peer sends three different P2PP requests: Query, Join, and Publish. The Query request is used to obtain overlay-specific information. The Join request is routed via the bootstrap peer to an admitting peer, which becomes the joining peer's first successor on the Chord ring. The Publish request is used to store the joining peer's contact information in the overlay.

To initiate a call, the caller first sends a P2PP LookupObject request to fetch the contact address of the called user from the overlay. Next, the caller sends a SIP INVITE request directly to the called user to establish the call.

To leave the overlay, a peer first sends a P2PP RemoveObject request to remove its contact information from the overlay. When the RemoveObject transaction has finished, the peer sends two P2PP Leave requests in parallel to its first predecessor and its first successor on the Chord ring.

TABLE I
TRAFFIC MODEL AND CHORD PARAMETERS

Parameter	Value
Interarrival time	1s,3s,5s,10s,20s,40s
Network size (N)	250, 500, 1000 peers
Measurement duration	3600s
Busy hour call attemps	2.21 calls per user
% of calls to buddies	66.6
Size of buddy list	22
Finger pointers	$logN$
Successors	$logN$

III. RELATED WORK

To the best knowledge of the authors, this is the first paper that analyzes delays in a real-world P2PSIP overlay.

References [5], [6] present the results of using the public OpenDHT service as a distributed lookup service for SIP. Unlike in our experiments, no P2PSIP protocols were used and the SIP nodes did not take part in the overlay but simply used OpenDHT via XML-based remote procedure calls.

In [7], the performance of a hierarchical distributed SIP system in which only super nodes participate in the overlay is analyzed. Unlike in our experiments, the results are based on simulations and theoretical calculations. No real DHT or P2PSIP protocols were used.

IV. EXPERIMENTS

The experiments were carried out in PlanetLab [8] instead of using a simulator or an emulated network. In each experiment, a global P2PSIP overlay was created from scratch. To run each experiment, the P2PSIP prototype was uploaded to a set of PlanetLab nodes mostly located in Europe, North America, and Asia. Three different network sizes, 250, 500, and 1000 peers, were used. Ideally, we would have used even larger overlays, but the number of simultaneously online PlanetLab nodes was a limiting factor. As an example, a small or middle size enterprise could have a global internal P2PSIP telephony network consisting of 250-1000 peers. Each peer starts collecting data after the maximum network size has been reached and continues this for one hour. After one hour, or before it leaves the overlay, the peer reports the data it has collected to a central server. The bootstrap peer of the overlay was located in Helsinki.

Each measurement with a given churn rate and network size combination was repeated 15 times. In total, 270 hours of measurements were run. The measurements were carried out between November 2008 and April 2009.

A. Traffic model

The one hour measurement period was modeled as a busy hour. Each user was assumed to initiate 13 VoIP calls per day, as suggested in [9]. Out of these 13 calls, 17% were used to represent busy hour traffic [10]. Thus, the number of busy hour call attempts per user was 2.21. This value was used as a mean rate for the arrival of calls, which was modeled as a Poisson process. Since users typically call their friends instead of strangers [11], it was assumed that 2/3 of the calls are placed to users on the buddy list.

TABLE II
CHORD STABILIZATION INTERVALS

IAT	1s	3s	5s	10s	20s	40s
N=250	10s	15s	20s	30s	60s	120s
N=500	15s	22s	30s	45s	90s	180s
N=1000	22s	30s	45s	75s	135s	270s

To model churn, six different peer interarrival times (IATs) and departure times were used, ranging from 1s to 40s. These correspond to average session times from 4 minutes (smallest network, highest churn rate) to 11 hours (largest network, lowest churn rate). Thus, a large number of scenarios is covered, from one where users join the network only to place a short call and leave after the call is over to a scenario in which the users keep the P2PSIP application open for a full working day or longer. The arrival and departure of users was modeled as a Poisson process. Each user is assumed to have a buddy list of size 22. This size was chosen based on the results in [12]. After having joined the overlay, each user initiates lookups to fetch the contact information of her buddies from the overlay. The traffic model is summarized in Table I.

B. Chord parameters

The size of Chord's successor list and finger table were set to $\log N$. In Chord, roughly $\Omega(\log^2 N)$ rounds of stabilization should occur in the time it takes N new peers to join or $N/2$ peers to leave the overlay [13]. This finding together with the results obtained in [14] were used to choose an appropriate stabilization interval for each network size and churn rate combination. The stabilization intervals are listed in Table II.

C. Measurements with SIP proxy-registrar

To compare the performance of P2PSIP to client/server SIP, we carried out separate measurements in which a centralized SIP proxy-registrar server was used instead of the P2PSIP overlay. The SIP proxy-registrar was running on a PlanetLab server located in Helsinki. The server was the same one that was used as the bootstrap peer in the P2PSIP experiments. In the experiments, a SIP User Agent (UA) application was uploaded to a set of PlanetLab nodes. The number of simultaneously registered SIP UAs was 1000, meaning that the size of the user population was the same as in the largest P2PSIP overlay network studied. SIP registrations arrive to the proxy-registrar according to a Poisson process with a mean interarrival time of 1000ms, which corresponds to the highest churn rate used in the P2PSIP experiments. Thus, in the SIP experiments, the proxy-registrar server experienced a load that was similar to the load in a 1000-peer P2PSIP overlay churning at the highest rate.

In this paper, we compare three different delays. First, we compare the SIP registration delay to the delay of joining a P2PSIP overlay. Second, we compare the call setup delays of client/server SIP and P2PSIP. Third, we compare the SIP de-registration delay to the delay of leaving a P2PSIP overlay.

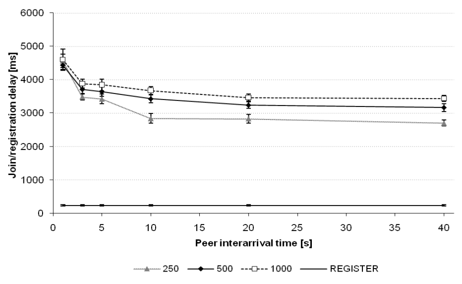

Fig. 1. Join operation delay

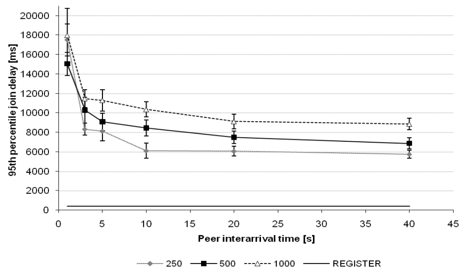

Fig. 2. 95th percentile join operation delay

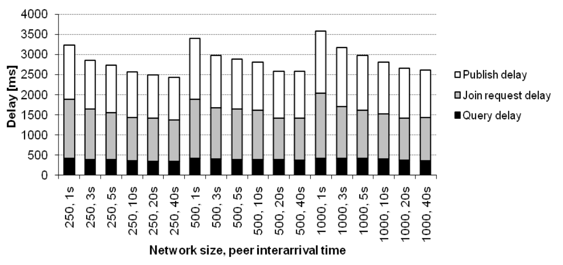

Fig. 3. Components of join operation delay

V. Results

This section presents the results of the experiments. The error bars in the figures represent 95% confidence intervals.

A. Join operation delay

Figure 1 shows the average delay a peer joining the P2PSIP overlay experiences. The figure depicts the join operation delay for all network sizes, N=250, N=500, and N=1000. The figure includes also join operations for which the Publish transaction fails. Also the SIP registration delay is included. The join operation delay consists of P2PP Query, Join, and Publish transaction delays. From the figure, we can observe that for each network size, the join delay grows significantly (statistically at the 95% confidence level) as churn increases. We can also see that differences between join delays at different network sizes are statistically significant for interarrival times higher than 5s. As expected, the lowest average delay, 2.7s, is experienced in a 250-peer network churning at the lowest rate and the highest average delay, 4.6s, in a 1000-peer network churning at the highest rate.

From Figure 1, we can also observe that the average SIP registration delay, 233ms, is over 11 times smaller than the lowest P2PSIP join delay. However, even the highest average join delay is likely to be tolerated by users; in the worst case, users need to wait on average for 4.6s after having started the P2PSIP application before being able to initiate a call.

In addition to the average delay, we also study the 95th percentile delay because that is what regulators and service level agreements use. The 95th percentile P2PSIP join operation and SIP registration delays are depicted in Figure 2. We can see that the lowest 95th percentile join delay is achieved in a 250-peer overlay churning at the lowest rate. In this case, the 95th percentile delay is 5.7s, which is 45 times higher than the 95th percentile SIP registration delay, 390ms. As expected, the highest 95th percentile delay, 18.0s, is experienced in a 1000-peer overlay churning at the highest rate. From Figure 1, we can also observe that SIP registration delay is much more predictable than P2PSIP join delay; the variance of the SIP registration delay is considerably smaller.

Figure 3 shows the components of join delay. Unlike in Figures 1 and 2, join operations for which the Publish transaction fails are not included. The average contributions of Query, Join, and Publish transactions to the join delay are 13.3%, 42.3%, and 44.4%, respectively. The delay of the Query transaction is lower than that of the Join and Publish transactions since the Query request is sent directly from the joining peer to the bootstrap peer unlike the other requests, which are routed via multiple hops across the overlay.

From Figures 1 and 2, we can conclude that joining a P2PSIP overlay is a rather expensive operation when compared to SIP registration. However, since at reasonable churn rates, the 95th percentile join delay is roughly 6-10s and this delay occurs only once when starting the P2PSIP application, it should not in practice pose a problem to users. Of course, if the network size grows beyond sizes used in the experiments, the average delay will increase since the average hop count of Join and Publish requests grows. However, since the growth in hop count is logarithmic [3], also the delay should grow in a logarithmic fashion. As an example, in the case of the lowest churn rates, doubling the network size from 250 to 500 increases the join delay by 17%, whereas doubling the size from 500 to 1000 increases the delay only by 9%.

B. Call setup delay

Figure 4 shows the average call setup delay for all network sizes and churn rates. The call setup delay consists of the delay between sending a lookup request to fetch the called user's contact information and receiving a final response to the SIP INVITE request sent to initiate the call. From the figure, we can observe that the differences between the network sizes are statistically significant starting from the interarrival time of 5s. The differences between the lowest and the highest network size are statistically significant also for other interarrival times. The call setup delay is higher in larger networks because in Chord, the average hop count is $\frac{1}{2}\log N$ [3]. This means that the hop count grows as a function of network size. The

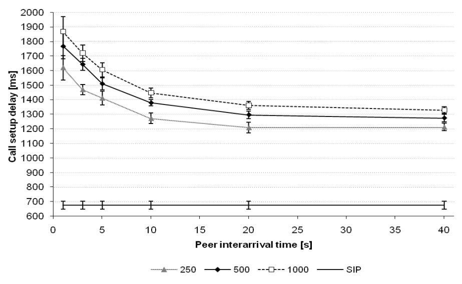

Fig. 4. Call setup delay

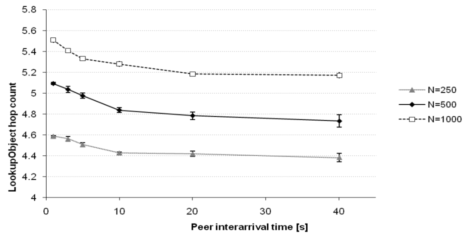

Fig. 5. Lookup request hop count

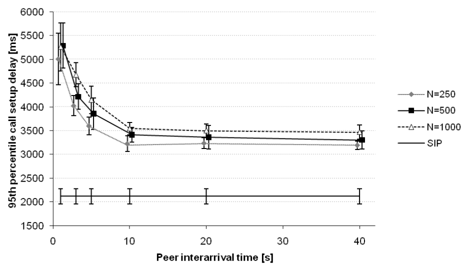

Fig. 6. 95th percentile call setup delay

impact of network size and churn rate on hop counts of lookup requests is depicted in Figure 5. From the figure, we can see that the hop count is indeed higher in larger networks. The hop count also grows significantly as churn increases.

From Figure 4, we can also observe that regardless of network size, the call setup delay seems to grow only logarithmically as churn increases.

Figure 4 includes also the call setup delay of client/server SIP. For client/server SIP, the call setup delay consists of only the INVITE transaction delay. The SIP UA sends the INVITE request to its outbound SIP proxy-registrar, which forwards the INVITE request to the contact address of the called user fetched from the proxy-registrar's database. In contrast, in P2PSIP, the INVITE request is sent directly to the called user after that user's contact address has been fetched from the overlay. The call setup delay of P2PSIP is 80-176% higher than the call setup delay of client/server SIP depending on the churn rate and network size. However, it should be noted that even the highest observed average P2PSIP call setup delay, 1866ms, does not seem particularly high.

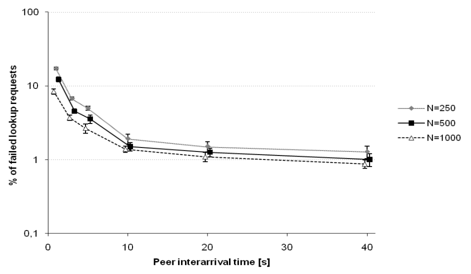

Fig. 7. Percentage of failed lookup requests

The 95th percentile call setup delays of P2PSIP and client/server SIP are depicted in Figure 6. In the figure, the delays of the 250 and 500 peer networks have been shifted to left since some of the error bars overlap. The 95th call setup delay is 3194-5288ms for P2PSIP depending on the churn rate and network size. For client/server SIP, it is 2118ms. From the figure, we can observe that differences between the smallest and largest network sizes are statistically significant for all other churn rates except for the highest one. In addition, for each network size, we can see that the 95th percentile call setup delay grows significantly as churn increases. The 95th percentile call setup delay is 51-150% higher for P2PSIP than for client/server SIP. Even in the worst case, 95% of P2PSIP calls experience an average call setup delay less than 5.3s.

C. *Failed calls*

Figure 7 shows the percentage of failed lookup requests. A lookup was considered to have failed if it experienced a timeout, exceeded the maximum number of hops, was rejected, or encountered a path error. Failed lookups were not retransmitted. In the figure, the Y axis uses logarithmic scale. The values for N=250 have been shifted to left and those of N=500 to right since some of the confidence intervals overlap. From the figure, we can observe that at the highest churn rate, 17.2%, 12.3%, and 8.4% of lookup requests fail when the network size is 250, 500, and 1000, respectively. At the lowest churn rate, the corresponding values are 1.3%, 1.0%, and 0.87%. For the three highest churn rates, the differences between the three network sizes are statistically significant. For the rest of the churn rates, only the differences between the largest and smallest network sizes are significant. The percentage of failed lookups is inversely proportional to network size; when the churn rate is the same, more lookups fail in smaller networks.

The reasons behind failed lookup requests are shown in Figure 8. From the figure, we can observe that regardless of network size, at the two highest churn rates, the most common reason behind failures is that an intermediate peer receiving a lookup request rejects the request since it is either leaving the overlay or has not yet finished the join operation. When churn is high, there are a lot of peers that are in the middle of the join or leave process. Such peers may not be able to make a reliable routing decision and thus they may reject the lookup. When a 250-peer network is churning at the same rate as a 1000-peer network, the probability that a lookup is routed to

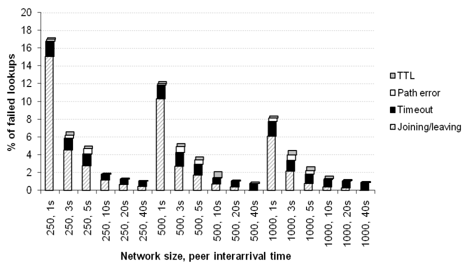

Fig. 8. Percentage of failed lookup operations

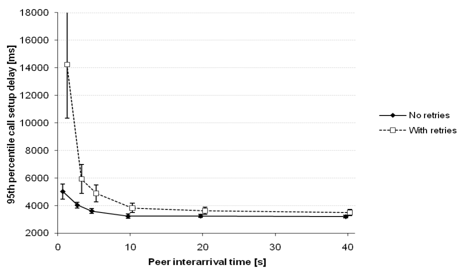

Fig. 10. 95th percentile call setup delay with and without retransmissions

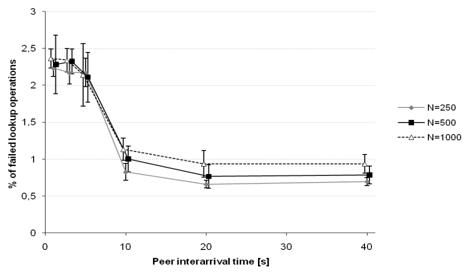

Fig. 9. Percentage of failed lookup requests with lookup retransmissions

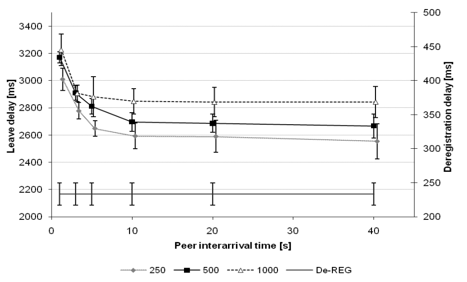

Fig. 11. Average leave delay

a joining or leaving peer is higher in the small network (in a 250-peer network churning at a rate of 1/s, a higher percentage of peers are in the middle of a join or leave operation than in a 1000-peer network). This explains why a higher percentage of lookups fail in smaller networks. However, the situation changes as the average peer interarrival and departure time grows. Now, lookup requests rejected due to join or leave no longer dominate. In the case of the two largest network sizes, lookups failing because of this reason become rather rare. Instead, nearly all lookup failures are caused by a timeout.

From Figure 7, we concluded that a rather large percentage of lookups fail when churn is high. Further, Figure 8 showed that most of these failures are caused by peers that cannot route messages since they are either joining or leaving the network. In Figures 7 and 8, lookups that were rejected by a joining or leaving peer were not retransmitted. Figure 9 shows the effect of retransmitting such lookups. From the figure, we can observe that at the three highest churn rates, the differences between the three network sizes are statistically insignificant. At the highest churn rate, roughly 2.3% of calls fail regardless of network size. This is a considerable improvement compared to Figure 7, in which between 8.4-17.2% of lookups failed depending on network size. For N=250, the drop in the number of failed lookups is 46-87%. For N=500, it is 22-81% and for N=1000, 0-72%. Starting from the interarrival time of 10s, the differences between the highest and lowest network sizes are statistically significant. When the interarrival time is 10s or larger, less than 0.82%, 1.0%, or 1.13% of lookups fail when the network size is 250, 500, or 1000, respectively.

Of course, if a lookup needs to be retried, the total call setup delay increases. From Figure 8, one can see that the number of lookups that need to be retransmitted is highest in the smallest overlay, in which lookup retransmissions have the effect of increasing the average call setup delay by 17.7%, 7.4%, and 3.0% in the case of the three highest churn rates. The impact on the 95th percentile call setup delay in the smallest overlay is depicted in Figure 10. The increase in call setup delay is higher at higher churn rates that have more retransmissions.

We also performed an extra set of 15 measurements to determine the amount of failed calls in a P2PSIP overlay that does not experience any churn. In such a network, there are no lookup failures caused by intermediate peers in the middle of a join or leave operation. In the measurements, the network size was 1000. The average percentage of failed calls was found to be 0.78% with a standard deviation of 0.32. The average number of calls during the one hour period was 234 and on the average, 1.8 out of these calls failed. All of the failures were caused by lookup timeouts. As a comparison, the average number of failed calls for client/server SIP was zero in similar network conditions. Thus, we can conclude that the inherently longer delays in P2PSIP may prevent one from achieving a zero failure rate even if the network conditions are ideal.

D. Leave operation delay

Figure 11 shows the average P2PSIP leave operation and SIP de-registration delays. In the figure, the Y axis is different for the two delay types. As explained above, the leave operation consists of one RemoveObject and two Leave transactions. The RemoveObject request is routed across the overlay whereas the Leave requests are sent on a direct connection. From the figure, we can see that the average leave delay is 2555-3226ms depending on churn rate and network size. The SIP de-registration delay is considerably smaller, 230ms, because it consists of only a single de-REGISTER transaction.

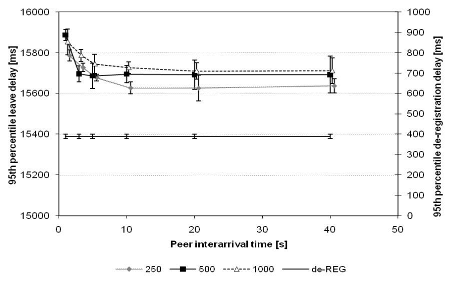

Fig. 12. 95th percentile leave delay

The 95th percentile leave operation and SIP de-registration delays are depicted in Figure 12. In the figure, the Y axis is different for the two delay types. The 95th percentile leave delay is 15.6-15.9s, depending on churn rate and network size. The 95th percentile SIP de-registration delay is considerably smaller, 390ms. The 95th percentile leave delay is rather high for all churn rates and network sizes. For instance, in a 1000-peer network churning at the lowest rate, 5% of the users need to wait for more than 15.7s after terminating the P2PSIP application for the leave operation to finish. The high 95th percentile leave operation delay is explained by P2PP Leave request timeouts; 5.3-6.8% of the requests experience a timeout depending on churn rate and network size. If a Leave request times out, it is not retransmitted. The high percentage of Leave request timeouts explains why the average leave delay (see Figure 11) is so high. The median leave delay is considerably smaller, 805-1052ms.

VI. Conclusion

In this paper, we studied various delays in a P2PSIP overlay through measurements carried out in PlanetLab. When compared to client/server SIP, P2PSIP clearly has higher delays. The average P2PSIP join operation delay is 11-20 times higher than SIP registration delay, depending on network size and churn rate. The average P2PSIP call setup delay is 2.4-5.0 times higher if taking lookup retransmissions into account. Finally, the average P2PSIP leave operation delay is 11-14 times higher than the SIP de-registration delay. However, in practice, the average delays of P2PSIP are still low enough so that they are likely to be acceptable to users. The average join, call setup, and leave delays were observed to grow logarithmically as churn or network size increases. This makes us think that delays will remain reasonable even in larger networks churning at higher rates.

In a network experiencing high churn, the majority of lookup failures occur since the lookup is routed to a peer that is in an inappropriate state. To achieve a satisfactory lookup success ratio, rejected lookups need to be retransmitted. However, this has the effect of increasing the call setup delay; in the worst case, the 95th percentile call setup delay became almost threefold. Consequently, in a network churning at a high rate, either some users experience very high delays because retransmissions are needed or, if retransmissions are not used, the lookup failure rate becomes rather high.

Based on our results, some insights can be derived for P2PSIP protocol design. First, the lookup procedure may need to be optimized in order to achieve a call failure rate that is comparable to client/server SIP. One possible optimization is to send parallel lookups along different paths. Second, although call setup delays seem acceptable in P2PSIP when compared to client/server SIP, there are large differences between SIP registration and de-registration delays and P2PSIP join and leave operation delays. Thus, rather than focusing only on optimizing the lookup delay, P2PSIP should also focus on minimizing the rather long join and leave operation delays. Third, we also saw that at higher churn rates, the majority of lookup failures occur because the lookup request is routed to a peer that is joining or leaving the network. Preventing such situations or quickly recovering from them without the need for the originator to retry the lookup would improve the performance of P2PSIP considerably.

References

[1] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, "Sip: Session initiation protocol," 2002.

[2] S. Baset, H. Schulzrinne, and M. Matuszewski, "Peer-to-peer protocol (p2pp)," IETF, Internet Draft – work in progress, November 2007.

[3] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan, "Chord: a scalable peer-to-peer lookup protocol for internet applications," *IEEE/ACM Trans. Netw.*, vol. 11, no. 1, pp. 17–32, 2003.

[4] C. Jennings, B. Lowekamp, E. Rescorla, S. Baset, and H. Schulzrinne, "Resource location and discovery (reload) base protocol," IETF, Internet Draft – work in progress, March 2009.

[5] B. Meyer and M. Portmann, "Practical performance evaluation of peer-to-peer internet telephony using sip," in *Proc. of the 2008 IEEE 8th International Conference on Computer and Information Technology Workshops (CITWORKSHOPS '08).* Washington, DC, USA: IEEE Computer Society, 2008, pp. 204–209.

[6] M. Matuszewski and E. Kokkonen, "Mobile p2psip: Peer-to-peer sip communication in mobile communities," in *Proc. of the 5th IEEE Consumer Communications and Networking Conference (CCNC 2008)*, Las Vegas, Nevada, USA, 2008, pp. 1159–1165.

[7] E. Harjula, J. Ala-Kurikka, D. Howie, and M. Ylianttila, "Analysis of peer-to-peer sip in a distributed mobile middleware system," in *Proc. of IEEE GLOBECOM*, San Francisco, CA, USA, 2006.

[8] L. Peterson, A. Bavier, M. E. Fiuczynski, and S. Muir, "Experiences building planetlab," in *Proc. of the 7th symposium on Operating systems design and implementation (OSDI '06).* Berkeley, CA, USA: USENIX Association, 2006, pp. 351–366.

[9] B. Athwal, F. C. Harmatzis, and V. P. Tanguturi, "Replacing centric voice services with hosted voip services: an application of real options approach," in *Proc. of the 16th international telecommunications society (ITS) European regional conference*, 2006.

[10] "Traffic analysis for voice over ip," 2001. [Online]. Available: http://www.cisco.com

[11] C. Cheng, S. Tsao, and J. Chou, "Unstructured peer-to-peer session initiation protocol for mobile environment," in *Proc. of the 18th Annual IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC '07)*, 2007, pp. 1–5.

[12] B. A. Nardi, S. Whittaker, and E. Bradner, "Interaction and outeraction: instant messaging in action," in *Proc. of the 2000 ACM conference on Computer supported cooperative work (CSCW '00).* New York, NY, USA: ACM, 2000, pp. 79–88.

[13] D. Liben-Nowell, H. Balakrishnan, and D. Karger, "Observations on the dynamic evolution of peer-to-peer networks," in *Proc. of the First International Workshop on Peer-to-Peer Systems (IPTPS '01).* London, UK: Springer-Verlag, 2002, pp. 22–33.

[14] J. Maenpaa and G. Camarillo, "Study on maintenance operations in a peer-to-peer session initiation protocol overlay network," in *Proc. of IPDPS*, 2009.