Publication III

Jouni Mäenpää and Jaime Jiménez Bolonio. Performance of REsource LOcation And Discovery (RELOAD) on Mobile Phones. In *2010 IEEE Wireless Communications and Networking Conference (WCNC)*, Sydney, Australia, pp. 1-6, April 2010.

Performance of REsource LOcation And Discovery (RELOAD) on Mobile Phones

Jouni Mäenpää and Jaime Jiménez Bolonio
Ericsson
Finland
{jouni.maenpaa, jaime.jimenez}@ericsson.com

***Abstract*—REsource LOcation And Discovery (RELOAD) is a peer-to-peer signaling protocol that can be used to maintain an overlay network, and to store data in and retrieve data from the overlay. RELOAD is currently being standardized in the Internet Engineering Task Force (IETF). The main application using RELOAD is Peer-to-Peer Session Initiation Protocol (P2PSIP). In this paper, we study the performance of RELOAD on a mobile phone participating in a P2PSIP overlay. We focus on memory consumption, CPU load, battery consumption, and bandwidth usage. The goal is to find out whether mobile phones can act as full peers in a P2PSIP overlay.**

I. Introduction

REsource LOcation And Discovery (RELOAD) [1] is a peer-to-peer (P2P) signaling protocol that is used in Peer-to-Peer Session Initiation Protocol (P2PSIP) overlay networks. P2PSIP is not a protocol by itself but rather a decentralized person-to-person communication system that uses the RELOAD protocol and the Session Initiation Protocol (SIP) [2] to enable real-time communication in a P2P environment. In a P2PSIP system, the centralized proxy-registrar and message routing functions of client/server SIP are replaced by a P2P overlay network. The overlay is organized using a Distributed Hash Table (DHT) algorithm. P2PSIP uses the RELOAD protocol to maintain the overlay network, and to store data in and retrieve data from the overlay. RELOAD is currently being standardized in the P2PSIP working group of the Internet Engineering Task Force (IETF). In addition to using RELOAD to maintain the overlay, P2PSIP uses SIP as the call control protocol. In the P2PSIP system, the overlay network is used as a lookup mechanism to map SIP address-of-record values to contact Uniform Resource Identifiers (URIs).

RELOAD is a binary protocol that uses type, length, value (TLV) fields to allow for extensibility. Each message has three parts: forwarding header, message contents, and security block. The forwarding header is used to forward the message between peers. The message contents include the actual information being delivered. The security block contains certificates and a digital signature over the message. RELOAD provides several features, including a security framework, usage model, Network Address Translator (NAT) traversal, high performance routing, and topology plugins. RELOAD's security framework supports two certificate issuance models. In the first model, a central enrollment server allocates certificates to peers. In the second model, self-signed and self-generated certificates are used. In the experiements conducted for this paper, the latter certificate issuance model is used. The usage model allows the definition of new application usages that use the overlay network service that RELOAD provides. NAT traversal is provided using the standard IETF NAT traversal mechanisms. High performance routing is made possible by RELOAD's lightweight forwarding header. Finally, new overlay algorithms can be implemented for RELOAD as topology plugins.

Two types of nodes can join a RELOAD-based P2PSIP overlay: clients and peers. The difference between clients and peers is that client nodes do not need to route traffic or store data for other nodes, whereas peers do. Clients use the services provided by the overlay through peers. A node may not be able to act as a full peer for a number of reasons, including a low-bandwidth network connection and limited resources such as computing power, memory, or battery power.

The goal of this paper is to study whether mobile phones can act as full peers in a RELOAD-based P2PSIP overlay. Peers in a P2PSIP overlay need to exchange frequent messages between each other to maintain the topology of the overlay. RELOAD also makes heavy use of cryptographic operations. In the paper, we study the costs of participating in a P2PSIP overlay from the viewpoint of mobile terminals. Our focus is on battery consumption, Central Processing Unit (CPU) load, memory consumption, and the amount of traffic exchanged. The paper is structured as follows. Section II gives an introduction to our RELOAD prototype. Section III describes the experiments and the traffic model. Section IV presents the results of the experiments. Finally, Section V concludes the paper.

II. P2PSIP Prototypes

The experiments were conducted using two P2PSIP prototypes: a mobile prototype and a fixed prototype.

A. Mobile Prototype

Our mobile prototype consists of two applications, a prototype for Java-enabled mobile phones and an application simulating a RELOAD-based P2PSIP overlay network that runs on a server. Both applications were implemented in the Java programming language. The prototype running on the terminal side was implemented as a Java Micro Edition (J2ME) application whereas the server side simulating the overlay uses the Java Standard Edition (J2SE).

Cryptographic operations are implemented using the BouncyCastle lightweight cryptographic Application Programming Interface (API). Peers in the overlay use self-signed X.509 certificates. RSA is used as the public key algorithm. The RSA key length is 1024 bits. Secure Hash Algorithm (SHA-1) with RSA encryption is used as the signature algorithm. RELOAD messages are exchanged over UDP. Use of a secure transport protocol such as Transport Layer Security (TLS) or Datagram TLS (DTLS) was not possible due to the following reasons. First, J2ME does not include a DTLS implementation. Second, although J2ME has a client-side TLS implementation, phones cannot act as TLS servers. This prevented the use of TLS since a mobile terminal participating in a RELOAD-based P2PSIP overlay needs to act as both a TLS client and server.

B. Fixed Prototype

Our fixed prototype is implemented as a J2SE application. Instead of using a simulated P2PSIP overlay, the experiments with the fixed prototype were carried out in PlanetLab [3], which is a global testbed for computer networking and distributed systems research. The difference between the mobile and fixed prototypes is that while the mobile prototype uses RELOAD as the peer protocol, the fixed prototype uses the Peer-to-Peer Protocol (P2PP) [4]. RELOAD is based on P2PP; the Internet draft defining P2PP preceded the current RELOAD specification. It was merged with the RELOAD specification during the P2PSIP standardization process in the IETF. The purpose of the experiments with the fixed prototype is to get information on message hop counts and delays in a P2PSIP overlay running in the real Internet. This information is then used to estimate the delays that a mobile peer would experience in a real-world P2PSIP overlay. The fact which peer protocol is used does not affect hop counts and thus also not the delays that messages experience.

C. Chord

Both in the fixed prototype and in the traffic model used with the mobile prototype, the Chord [5] DHT is used to organize the topology of interconnections amongst the peers in the overlay. Since Chord is used, the overlay has a ring topology. Chord is a structured P2P algorithm that 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 consisting of a finger table, and successor and predecessor lists. In an N-node network, each node maintains information about $O(\log N)$ other nodes in its finger table. The successor list contains the peer's immediate successors on the Chord ring. It has been shown that a successor list size of $\log N$ is sufficient to ensure that lookup performance is not affected even by massive simultaneous peer failures [5]. The predecessor list contains the peer's immediate predecessors on the Chord ring.

To ensure that the contents of the routing table stay up-to-date with the constantly changing topology of the overlay, each peer runs a stabilization routine periodically [6]. The stabilization routine consists of four operations: predecessor stabilization, successor stabilization, finger stabilization, and strong stabilization. In the predecessor stabilization routine, a peer synchronizes its predecessor list with its first predecessor. In successor stabilization, a peer synchronizes its successor list with its first successor. During each finger stabilization operation, a peer tries to refresh one of its finger table entries. Finally, the purpose of the strong stabilization routine is to detect loops. In strong stabilization, a peer searches for itself from the Chord ring by routing a lookup request for its own identifier via its successor.

III. Experiments

A. Mobile Prototype

In the experiments, a phone running the mobile prototype joined a simulated Chord-based P2PSIP overlay network. Both the server simulating the P2PSIP overlay and the mobile phone were located in Helsinki. The phone model used in the experiments was Sony-Ericsson C905, which is a feature phone (in contrast to being a smart phone). The phone was connected to the Internet using a Third Generation (3G) High Speed Downlink Packet Access (HSDPA) connection with 2048 kbit/s downlink and 384 kbit/s uplink bandwidth. The 3G network of the Finnish operator DNA was used. The DNA network allocates public IP addresses to mobile phones. Other wireless interfaces of the phone were switched off during the measurements. After the application was started, the screen of the phone was allowed to power off and it was not activated during the measurement period. Before starting the measurements, the battery was fully charged. A brand new battery was used. The experiments were run in conditions in which the phone reported the maximum 3G signal strength.

During the measurements in which CPU load, memory consumption, and bandwidth usage were monitored, the phone was connected via a Universal Serial Bus (USB) cable to a laptop that collected measurement data from the phone using Sony-Ericsson Resource Monitor software. In the battery consumption related measurements, the phone was not connected to the laptop. The battery charge information was obtained using Java Standardization Request (JSR) 256, also known as the Mobile Sensor API. JSR 256 allows J2ME applications to fetch data from sensors on a mobile phone.

1) Traffic Model: In our traffic model, the size of the overlay is N=10000 peers and the average session time of peers is eight hours. A session time of eight hours was selected since it corresponds to a full working day. The size of the successor list is 13 peers, i.e., $O(\log N)$. The size of the finger table is also 13 peers. The overlay uses the successor replication strategy [7] to improve reliability. A replication factor of three is used. Because of the way successor replication works, this means that peers in the overlay need to maintain a predecessor list with four entries. Thus, in total, the routing table of each peer contains 30 entries. Peer-IDs of peers participating in the overlay are distributed uniformly at random. In the traffic model, the recursive overlay response routing mode [8] (also known as symmetric recursive routing) is used.

TABLE I
TRAFFIC MODEL

Parameter	Value
DHT algorithm	Chord
Chord stabilization interval	85s
Finger table size	13
Successor list size	13
Predecessor list size	4
Replication factor	3
Average session time	8h
Network size	10000
Measurement duration	3600s

TABLE II
RELOAD MESSAGES

RELOAD Message	Mean interval [s]
Update (successor stabilization)	85
Update (predecessor stabilization)	85
Find (finger stabilization)	12.8
Find (strong stabilization)	11.1
Join	more than 3600
Leave	more than 3600
Attach	61.9
Store	1609

In a Chord-based overlay, roughly $\Omega(\log^2 N)$ rounds of stabilization should occur in the time it takes N new peers to join the overlay or $N/2$ peers to leave the overlay [9] to keep the routing tables of peers consistent with the constantly changing topology of the overlay. Using this formula, a value of 85s can be calculated for the stabilization interval. The traffic model is summarized in Table I. We verified the model against results obtained by running the fixed P2PSIP prototype in the PlanetLab and found it to be accurate.

In [10], it has been shown that in a P2PSIP overlay, traffic generated by the stabilization routines clearly dominates over lookup traffic; up to 95% of the total traffic exchanged between peers can consist of stabilization traffic. Therefore, in our experiments, we chose to focus only on stabilization traffic. In the case of the Chord DHT used in the experiments, stabilization traffic consists of predecessor, successor, and finger stabilization operations as was discussed in Section II-C.

The forwarding header included in every RELOAD message includes among other things a destination list and a via list. The destination list is used for source and return path routing, whereas the via list is filled by peers forwarding the message. As specified in [1], if an intermediate peer is willing to store state, it can choose to truncate the via list of a request and save the information internally. In our experiments, peers always truncate the via list to reduce the size of RELOAD messages.

Certificates are only included in RELOAD messages routed across the overlay. There is no need to include them in single-hop messages exchanged between two neighbors on a direct connection; these peers already know each other and thus have already exchanged certificates. In practice, this means that Update messages do not include certificates, whereas all other messages do. Further, as was explained above, each RELOAD message needs to be signed. Messages are signed by their initiator. The signature as well as the sender's certificate is verified by every peer receiving the message.

To implement the successor and predecessor stabilization routines, a peer sends one Update request to its first predecessor and another Update request to its first successor on the Chord ring. The peer executes the finger stabilization routine by sending a Find request to a peer selected from its finger table. The Find request is also used to implement the strong stabilization routine. Find requests are routed across the overlay. Whenever a new peer needs to be included in the routing table as a result of the stabilization routines, an Attach request is routed across the overlay to the peer to establish a direct connection to it. As peers come and go, the contents of the successor and predecessor lists change. When this happens, Store requests are sent to transfer the ownership of resource records and replicas. Store requests are sent on a direct connection. The RELOAD messages that are sent and received by the phone are listed in Table II together with the mean interval at which the phone receives and sends each message. Since the overlay is large, the probability that the phone receives a Join or Leave message is relatively low.

B. Fixed Prototype

To gain an understanding of delays and message hop counts that peers having a wired connection to the Internet experience in a P2PSIP overlay, we also carried out experiments using the fixed P2PSIP prototype. In the experiments, a set of PlanetLab nodes created a 1000-peer P2PSIP overlay. Ideally, we would have used an even larger overlay, but the number of simultaneously online PlanetLab nodes was a limiting factor. We used 250 dedicated PlanetLab nodes. Thus, on average, four instances of the prototype were running simultaneously on each PlanetLab node. In the experiments, we collected information on average hop count values and Round-Trip Times (RTTs). In these experiments, the session time of peers was eight hours. A total of 20 measurements were carried out. The delay and hop count values are averages over the 20 measurements. Data collection was started when the size of the overlay reached 1000 peers and was continued for one hour. During the measurement period, the size of the overlay stayed at 1000 peers. The Chord stabilization interval was 135s.

IV. RESULTS

This section presents the results of the experiments.

A. Memory Consumption

The Java memory usage on the phone is shown in Figure 1. The amounts of used and free Java memory were collected once per second. The figure plots the size of the Java heap, which is used to store all the data needed by the application. From the figure, we can see that the size of the Java heap varies between 547 and 767 kB. The average size of the heap (i.e., average memory usage) is 661 kB. Since the maximum Java heap size available to applications on the phone model used in the experiments is 30 MB, we can conclude that the memory usage of the mobile prototype does not pose a problem. We

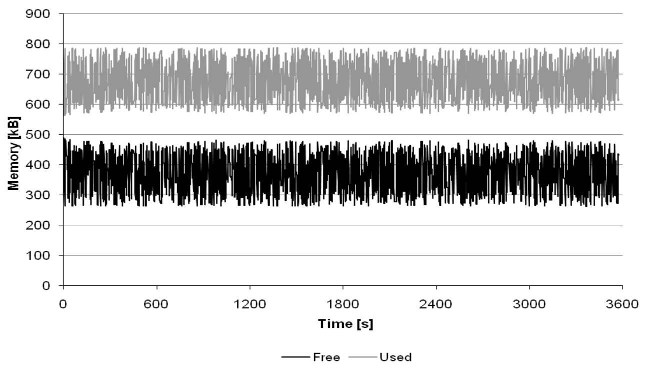

Fig. 1. Java Memory Consumption

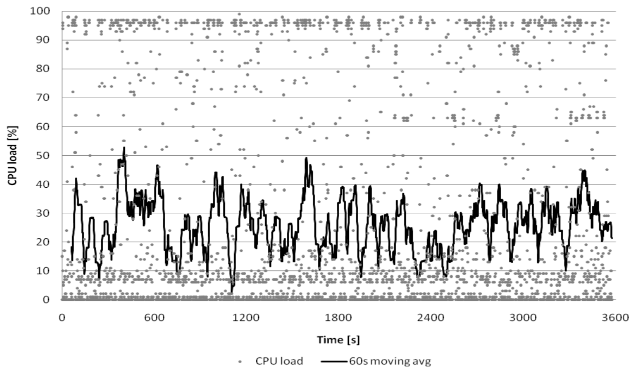

Fig. 2. CPU Load

also measured the memory consumption of a J2ME-based non-P2P instant messaging application. The memory consumption was found to vary between 350 and 850 kB depending on the amount of instant messages sent and received during a chat session. Thus, we can conclude that the memory consumption of the mobile P2PSIP prototype is not excessive. Figure 1 also shows the free Java memory on the phone. The phone platform increases the heap size dynamically as needed. However, since the RELOAD prototype does not at any point fill the initially allocated 1.0 MB Java heap area, there is no need to increase the heap size during the experiment.

In contrast to traditional P2P applications such as file sharing, in a P2PSIP communication system, the resource records peers store in the overlay are rather small. This reduces the amount of storage and memory capacity required at each peer. The resource records P2PSIP peers store in the overlay are contact records containing SIP Address of Record (AoR) to node-ID mappings. These contact records contain a RELOAD SipRegistration Protocol Data Unit (PDU) defined in [11], the size of which was 190 bytes in our experiments. A typical peer only stores one such contact record in the overlay. Since the size of the overlay was 10000 peers and a replication factor of three was used, the overlay contains on average 40000 contact records. In a system in which node-IDs are distributed uniformly at random, this means that each peer is storing on average only four contact records.

B. CPU load

The CPU load of the phone is shown in Figure 2. Information on CPU load was collected once per second. The figure shows each individual CPU load value and in addition, the 60s running average of the CPU load. The average CPU load was 25.7% with a standard deviation of 36.9. The 95th and 85th percentile CPU loads were 96% and 94%, respectively, meaning that in 15% of the cases, the CPU load was higher than 94%. By looking at the 60 moving average of the CPU load, we can see that the CPU load varies greatly during the measurement period depending on the traffic load the peer running on the phone experiences. The main reason behind the high CPU load are the cryptographic operations needed to execute for each incoming and outgoing RELOAD message, including verification of certificates and signatures, and generation of signatures.

The average loads that different applications cause on the phone are compared in Table III. These applications include the mobile P2PSIP prototype, a chess application, a 3D racing game, and a non-P2P instant messaging application. From the table, we can see that the average load caused by the mobile P2PSIP prototype is only 1.4 times higher than for the non-P2P instant messaging application and 2.2 times higher than for the chess application. Further, it is considerably (3.7 times) lower than the load caused by the 3D racing game. However, when compared to the other applications, the CPU load of the mobile P2PSIP prototype varies much more over time. This is because of the peaks in CPU load occurring when the mobile phone carries out cryptographic operations.

TABLE III
CPU LOADS OF DIFFERENT APPLICATIONS

Application	Avg [%]	Stdev [%]	95th percentile [%]
P2PSIP prototype	25.7	36.9	96
Chess application	11.9	9.4	17
3D racing game	92.1	5.9	96
Instant messaging	18.0	14.0	31

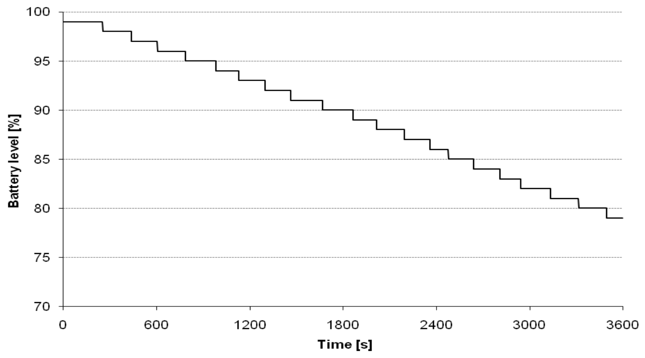

Fig. 3. Battery Consumption

C. Battery Consumption

The battery consumption of the phone is depicted in Figure 3. During the one hour measurement period, the battery

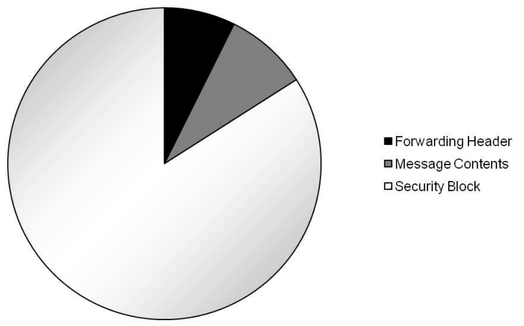

Fig. 4. Sizes of RELOAD Message Parts

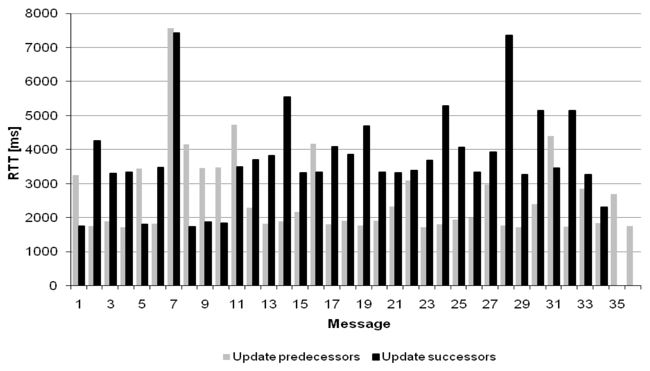

Fig. 5. Delay between Mobile Phone and Server

charge dropped from 99% to 79%. We also carried out an additional experiment in which the phone joined the overlay and stayed online until it ran out of battery. The battery was drained in 4h 50min. Therefore, from the viewpoint of battery consumption, the cost of participating as a full peer in a P2PSIP overlay seems rather high for a mobile phone.

To understand the reasons behind battery consumption, it is necessary to consider how often data is sent over the radio channel. During the one-hour measurement period, the phone received 1356 messages, meaning that the average time between incoming messages was only 2.65 seconds. The phone also sent roughly the same amount of messages. In a 3G Wideband Code Division Multiple Access (WCDMA) network, the radio channel stays allocated at least several seconds after a packet has been sent [12]. Thus, during the measurement period, the phone may not have many chances to transition to a low-power state in which the channel has been released. According to [13], the typical power consumption in a 3G WCDMA network is 200-400mA when a terminal has a dedicated channel. If the phone shares a channel with other phones (this Radio Resource Control (RRC) state is used if the phone does not have much data to transmit), the power consumption roughly halves [12]. The battery capacity of the phone we used in the experiments was 930 mAh. Since the bandwidth usage was rather low during our experiments, as will be described in Section IV.D, but still high enough to cause the phone to use a dedicated channel, the power consumption can be assumed to be closer to 200mA than to 400mA. Assuming a power consumption of 200mA, we can calculate that the battery should last roughly 4 hours 39 minutes. This value is almost exactly the same as the measured battery time. As a comparison, the maximum talk time of the phone model used in the experiments was up to four hours when using HSDPA, which is also rather close to the time in which the mobile prototype drained the battery.

D. RELOAD Messages

The average RELOAD message size was 819 bytes (with a standard deviation of 146). Figure 4 depicts the average size of different parts of the RELOAD message structure. The security block clearly takes the largest part of the message. It represents 84.1% of the total message size. The forwarding header takes up 7.3% and the message contents 8.6% of the total message size. Therefore, even in our reasonably large 10000-peer overlay, most of the bandwidth consumption is due to exchange of certificates and signatures. Another interesting finding is that the average size of the forwarding header is nearly equal to the average size of the message contents. Thus, even without the security block, the average size of the header is roughly as large as the average size of the actual payload.

The average incoming traffic received by the phone and the average outgoing traffic sent by the phone were roughly equal, 2.46 kbit/s. This figure includes only the bandwidth consumption of RELOAD messages; the overhead of lower layers in the protocol stack has been omitted. During the one hour measurement period, the phone sent and received roughly 1.1 megabytes of messages. Thus, bandwidth consumption does not seem like an issue for a mobile phone participating in a 10000-peer P2PSIP overlay.

E. Delays

Figure 5 shows the round trip delays of Update transactions between the mobile phone and its first successor and first predecessor on the simulated Chord ring. Only the delays of Update transactions are shown since their hop count is always one. The other messages are routed to their destination across the overlay via multiple hops. The Update delays were measured by sending a request from the server side to the phone and then waiting for the response.

The average round trip delay of Update transactions used in the predecessor stabilization routine was 2609 ms, whereas the average delay of Update transactions used in successor stabilization was 3732 ms. The standard deviations of predecessor stabilization and successor stabilization related Updates were rather high, 1219 and 1309 ms, respectively. The large variance in delay is caused by the radio interface. It should be noted that the delays include also the time it takes to sign the request and the response and to verify the signatures included in them. On the average, it took the mobile phone 1653 ms to sign a RELOAD message and 197 ms to verify the signature. Update transactions related to the successor stabilization routine have higher delays since their size is larger. The Update responses sent in predecessor stabilization include only the predecessor list whereas Update responses sent in successor stabilization include both the predecessor and successor lists.

These results can be compared against the data we obtained by running the fixed P2PSIP prototype in PlanetLab. The average round trip delay of predecessor stabilization related Update transactions between fixed peers in a global P2PSIP overlay running in PlanetLab was observed to be only 169 ms, which is roughly 15 times times less than for the mobile peer. Further, the average delay of multi-hop requests in the 1000-peer PlanetLab overlay was 846 ms with a standard deviation of 56 ms. The average hop count was 5.2. If substituting one of these hops with the observed delay between the mobile phone and the server, the average multi-hop request delay for a mobile phone participating in a 1000-peer overlay is 3292 ms. Thus, even if only the first hop is wireless, the delay becomes 3.9 times higher for a mobile peer than for the fixed peers. Of course, the delay is even higher if a message goes via multiple mobile peers on its way to its destination. Clearly, the presence of mobile peers can result in dramatic increases in the delays RELOAD messages experience.

To estimate the delays in a P2PSIP overlay consisting of only mobile peers, we performed an additional measurement to determine the average delay of RELOAD lookup transactions routed around the Chord ring through five wireless hops (which corresponds to the average hop count in a 1000-peer overlay) in an overlay created by five mobile phones. The sizes of the lookup request and response were set to the average message size observed in Section IV-D, that is, to 819 bytes. The phones were located in the same cell. The average delay over 70 such measurements was 12042 ms with a standard deviation of 3300 ms. This delay is very high since it is 14 times higher than the delay of routing through five hops in a PlanetLab overlay (which was 846 ms, as explained above).

V. Conclusions

In this paper, we studied the load a mobile phone experiences when participating as a full peer in a 10000-peer P2PSIP overlay that uses RELOAD as the peer protocol. The average memory consumption was found to be 661 kB, which is fairly low considering that the maximum available Java heap size is typically several megabytes even on low-end phones. The memory consumption is not excessive when compared to other J2ME applications. This makes us think that memory consumption should not pose a problem even for low-end feature phones acting as peers in a P2PSIP overlay.

The average bandwidh consumption of RELOAD messages was 2.46 kbit/s. This figure seems very low considering that current 3G radio technologies can provide broadband connections with speeds of several Mbit/s. Certificates and signatures form the largest part, 84%, of RELOAD messages. Thus, they cause the majority of bandwidth consumption in the overlay. The amount of actual information delivered between peers uses only 8.6% of the bandwidth. This makes RELOAD a rather expensive protocol for peers having only a narrowband connection to the Internet.

We also saw that the presence of mobile peers can increase average message delays considerably. The cost of exchanging a message with a mobile neighbor can be on average 15 times higher than for a fixed neighbor. Average lookup delay can be 14 times higher in an overlay consisting of only mobile peers than in an overlay consisting of fixed peers.

The average CPU load on the phone was 25.7%, which is reasonable when compared to other J2ME applications. The battery consumption of the phone was observed to be rather high. The main reason for this seems to be that the phone needs to send and receive RELOAD messages so frequently that the 3G radio channel stays allocated almost all the time.

As a summary, messaging delays and battery consumption can become bottlenecks when a mobile peer is participating in a P2PSIP overlay. However, memory usage, CPU load, and bandwidth consumption do not seem like issues.

Acknowledgment

This paper is partially based on a thesis titled ”Implementation and Performance of REsource LOcation And Discovery (RELOAD) Parser and Encoder” written by Jaime Jiménez Bolonio at Ericsson for Universidad Carlos III de Madrid.

References

[1] C. Jennings, B. Lowekamp, E. Rescorla, S. Baset, and H. Schulzrinne, “Resource location and discovery (reload) base protocol,” IETF, Internet Draft – work in progress 02, March 2009.

[2] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, “Sip: Session initiation protocol,” IETF, United States, Tech. Rep., 2002.

[3] L. Peterson, A. Bavier, M. E. Fiuczynski, and S. Muir, “Experiences building planetlab,” in *OSDI ’06: Proceedings of the 7th symposium on Operating systems design and implementation*. Berkeley, CA, USA: USENIX Association, 2006, pp. 351–366.

[4] S. Baset, H. Schulzrinne, and M. Matuszewski, “Peer-to-peer protocol (p2pp),” IETF, Internet Draft – work in progress 01, November 2007.

[5] 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.

[6] S. Rhea, D. Geels, T. Roscoe, and J. Kubiatowicz, “Handling churn in a dht,” in *ATEC ’04: Proceedings of the annual conference on USENIX Annual Technical Conference*. Berkeley, CA, USA: USENIX Association, 2004, pp. 10–10.

[7] S. Ktari, M. Zoubert, A. Hecker, and H. Labiod, “Performance evaluation of replication strategies in dhts under churn,” in *MUM ’07: Proceedings of the 6th international conference on Mobile and ubiquitous multimedia*. New York, NY, USA: ACM, 2007, pp. 90–97.

[8] C. O. William, D. A. Bryan, M. Zangrilli, and B. B. Lowekamp, “Challenges of dht design for a public communications system,” College of Willian and Mary, Technical Report WM-CS-2006-03, 2006.

[9] D. Liben-Nowell, H. Balakrishnan, and D. Karger, “Observations on the dynamic evolution of peer-to-peer networks,” in *IPTPS ’01: Revised Papers from the First International Workshop on Peer-to-Peer Systems*. London, UK: Springer-Verlag, 2002, pp. 22–33.

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

[11] C. Jennings, B. Lowekamp, E. Rescorla, S. Baset, and H. Schulzrinne, “A sip usage for reload,” IETF, Internet Draft – work in progress 01, March 2009.

[12] H. Haverinen, J. Siren, and P. Eronen, “Energy consumption of always-on applications in wcdma networks.” in *VTC Spring*. IEEE, 2007, pp. 964–968.

[13] H. Holma and A. Toskala, *WCDMA for UMTS: Radio Access for Third Generation Mobile Communications*. New York, NY, USA: John Wiley & Sons, Inc., 2004.