Computer Engineering and Applications Vol. 5, No. 2, June 2016

High Load Control Mechanism for SIP Servers

Ahmadreza Montazerolghaem[1], Seyed-Amin Hosseini-Seno[2], Mohammad Hossein Yaghmaee[3] and Rahmat Budiarto[4]

1,2,3 Department of Computer Engineering, Ferdowsi University of Mashhad, Mashhad, Iran,
4Dept. Of Computer Information System, Al Baha University, Kingdom of Saudi Arabia
Ahmadreza.montazerolghaem@stu-mail.um.ac.ir, hosseini@um.ac.ir, h.yaghmaee@um.ac.ir, rahmat.budiarto@surya.ac.id

ABSTRACT

To start voice, image, instant messaging, and generally multimedia communication, session communication must begin between two participants. SIP (session initiation protocol) that is an application layer control induces management and terminates this kind of sessions. As far as the independence of SIP from transport layer protocols is concerned, SIP messages can be transferred on a variety of transport layer protocols including TCP or UDP. Mechanism of Retransmission that is embedded in SIP could compensate for the missing packet loss, in case of need. This mechanism is applied when SIP messages are transmitted on an unreliable transmission layer protocol like UDP. Also, while facing SIP proxy with overload, it could cause excessive filling of proxy queue, postpone increase of other contacts, and add to the amount of the proxy overload. In the present work, while using UDP as transport layer protocol, invite retransmission timer (T1) was appropriately regulated and SIP functionality was improved. Therefore, by proposing an adaptive timer of invite message retransmission, attempts were made to improve the time of session initiation and consequently improve the performance. Performance of the proposed SIP was implemented and evaluated by SIPP software in a real network environment and its accuracy and performance were demonstrated.

Keywords*:* Load Control, TCP, SIP, VoIP, Regulating Timers.

1. INTRODUCTION

In recent years, Internet-based networks can be seen everywhere that makes dial-up calls via Internet Protocol (IP) network became popular. Besides, the following factors have more welcomed such communication. First is about economic problems; Internet phone calls, especially for international calls, are much cheaper than typical phone calls. The second factor is the development of IP communication in a variety of applied equipment. Personal computers are easily connected to the Internet using different modes. Existence of IP in cell phones is another factor that could cause further development of the Internet than other technologies. Another factor is further use of packet switched architecture, instead of circuit switched one that can itself make optimum use of the resources possible.

In simultaneous and two-way communication such as audio and video ones, file transfer, exchanging instant message, and generally multimedia sessions, in which communication is online, first, a session must initiate among the participants. The most important point about these types of communication, especially the Internet phone call contact, is signaling, which is responsible for the task of initiating and managing each session. One of the very suitable protocols in this field is SIP, the task of which is making, modifying, and terminating the session. One of the key components in delay-sensitive applicable programs like voice and image transmission by the Internet is the time required for the session start-up, which is highly effective in protocol efficiency. Reduction of this period of time leads to increase in the SIP server transmission and consequently improves its efficiency. On the other hand, it will cause more user acceptability. Due to the importance of this issue, some researchers have focused on reducing the necessary time for creating the session in progress, some examples of which are mentioned below.

This work aims to improve the session initiation time in UDP protocol by using an appropriate regulation of timer and sending invite message retransmission in such a way that the session initiation time will be improved and the number of missed calls was reduced.

The rest of this paper is organized as follows: Section 2 provides related works. Section 3 gives an overview to improve SIP efficiency while using UDP. Section 4 presents the adaptive timer T1 including its estimation. Section 5 presents the implementation of the proposed adaptive timer T1, followed by the experimental results and discussion. Lastly Section 6 concludes this paper.

2. RELATED WORKS

As certain servers which are specified for SIP can be used for a session, increasing their number, load distribution among them, or their processing capability can help reduce the time for session initiate [1]. It is worth knowing that these changes are costly and sometimes difficult. Another way is to apply stateless mode instead of stateful in servers [2]. However, it has a major problem; a complete history of communication will not be available [3]. Another method which can be used for reducing the time for session initiation is to remove user authentication confirmation, which has the highest positive impact on reducing the time of the session initiation; however, due to the nature of the Internet network and the necessity of using authentication confirmation in some cases, this method has some considerable problems as well. On the other hand, another important component, which must be considered, is the percentage of missed sessions, the reduction of which evidently influences the protocol.

Generally, with regard to compatibility in interoperability, UDP is considered the standard transmission protocol for SIP; however, it should be considered that other protocols such as TCP and SCTP are also reliable and applicable [4] (in some cases, use of connection-oriented protocols are required: for example, in the circumstances in which the length of SIP message transmission is more than its MTU [5]). In contrast, according to the standard, all the implementations of SIP must back-up TCP and UDP [10]. Accordingly, by appropriately selecting transmission protocol and UDP as transmission layer protocol and by regulating SIP timers, some conditions can be provided for increasing the protocol effectiveness in various network conditions considering the amount of packet loss and delay and the amount of traffic load of SIP. Since the session initiator can determine the transmission protocol, the responder complies with it, and this selection is decided in each node of the SIP network, therefore, some measures can be done at each point of the network considering the network conditions for selection [6]. In this case, selection is done dynamically; without spending certain expense, transmission can be increased and as a result high efficiency is obtained.

3. IMPROVING SIP EFFICIENCY USING UDP

In this work, a situation was considered in which, due to some reasons, UDP was assumed as transmission layer protocol for SIP. In this situation, the regulations were so that the efficiency of SIP is improved. Moreover, two important parameters were considered as the assessment criteria for the efficiency of SIP. The first parameter was the time required for session initiation and the next parameter was the ratio of the sessions' loss to their start-up. Now, attempts were made to take actions in order to increase the protocol efficiency by accurate regulation of the T_1 retransmission timer on SIP, compared to the increased efficiency of the protocol action. As is known, UDP protocol is an unreliable protocol and no mechanism exists for guaranteeing the delivery of messages to the destination. On the one hand, sometimes due to some of its advantages, it is selected as the transmission layer protocol; in such cases, if the guaranteed delivery of packets to the destination is important, the management mechanism must be implemented in the application layer.

3.1. APPLICATION OF TIMER T

In SIP, some timers are placed to manage the messages' retransmission in the message loss cases, among which timer T_1 can be named as the most important one. This timer specifies the initial amount of timer A as well as the amount of timer B while the default amount of SIP timer is 500 msec. Timer A is the invite message retransmission and has a relation with timer T_1 as in Equation 1.

$$Timer\ A = 2^{J-1} \times T_1 \qquad (1)$$

J is the times of invite message retransmission. As can be seen, message retransmission is in the form of a symbol of the initial amount of timer T_1; i.e. when the invite message is sent, User Agent Client (UAC) waits for a period of A= T_1 to receive a temporary reply (for example, 180 ringing) or final reply (200 OK) from UAS; if it does not receive any reply up to the end of timer A, the mentioned formula is used to reset timer A and transmit invite message again. As can be observed, at every stage of invite retransmission, the amount of timer A will be doubled in order to increase the probability of receiving the reply, which will continue (for 7 times) so that timer B will be terminated; this timer is equivalent to 64 times of timer T_1 according to formula in Equation 2.

$$Timer\ B = 64 \times T_1 \qquad (2)$$

As demonstrated in this formula, the amount of timer B is 64 times of initial timer T_1, which means that, after this period of time and not receiving a final reply, time-out will occur in UAC and contact will totally drop out. Now, T_1 has a very important impact on the message retransmission as well as contact drop-outs. Since its accurate regulation can considerably help improve efficiency, this timer was regulated in the direction of improving the average time of the session initiation as well as reducing the session loss.

3.2. REGULATING TIMER T1

T_1 specifies the initial values of timers A and B, which are respectively responsible for regulating the retransmission of invite message and time-out of the sessions. Therefore, in order to specify the function of timer T_1, the following points can be mentioned regarding the regulation of timer T_1.

- If delay is low in the network, the amount of timer T_1 can be reduced to prevent waiting in vain, in case of the deterioration of the transmitted invite message from UAC or temporary transmitted reply from UAS.
- If the network delay is higher than timer T_1's default (which is 500 msec) (such as satellite transmission environment or due to sectional density), the amount of timer must be added to prevent retransmission and vain invite messages, the replies of which are on the track, due to the network's natural delay.
- If the delay in receiving the reply from UAS is due to CPU's high load in the mentioned node, the time of timer T_1 can be increased to reduce the working load of UAS.
- In case of the existence of loss in the environment and loss or temporary reply of the invite message, it is better to reduce the speed of invite message retransmission through increasing the time of timer T_1 so that congestion can be reduced.
- If invite messages or temporary reply reache the destination defectively, it is better to take action to retransmit the invite message immediately by reducing time T_1 in order to prevent waiting for a reply that will never be received.

4. ADAPTIVE TIMER T_1

In order to have the best condition of the protocol performance, invite message retransmission should be performed at the best time, when the reply is not received from UAS. If the retransmission is done later than its time, it will result in the increased average time required for initiating a session and, if it is done sooner than that, it might be in vain. This work refers to the works in [7-10] to come up with the proposed adaptive timer T_1

4.1. DETECTING RETRANSMISSION TIME OF INVITE MESSAGE

The time that is required for waiting to get replies is analyzed. The time for receiving the reply after transmitting the invite message (called delay (invite→ok)) was evaluated. Hence, the time for receiving reply can be obtained through the following total periods of time.

- Period of time for getting the invite message from UAC to UAS (called $\text{delay}_{(\text{invite-prop})}$),
- Period of time for analyzing the invite message in UAS and initiating temporary or final reply messages. This time is the sum of all waiting periods of time in the queue and UAS processing (called $\text{delay}_{(\text{Queu/Proc})}$).
- Period of time for the reply message (temporary or final) to get from UAS to UAC (called $\text{delay}_{(\text{Ok-Prop})}$).

$$Delay_{(invite\rightarrow ok)} = Delay_{(invite\text{-}prop)} + Delay_{(Queu/Proc)} + Delay_{(Ok\text{-}Prop)} \qquad (3)$$

To determine the values Equation 3, the following procedure was followed.

(1) To determine $Delay_{(invite\text{-}prop)} + Delay_{(OK\text{-}Prop)}$, Ping was used. With this estimation, the transmission and receiving time in the network or Round Trip Time (RTT) was obtained. As is known, the length of the transmitted packages in the network affects their sending and receiving and larger length of the packages will increase the RTT time. To more appropriately estimate the invite message transmission time and receiving the reply message, Ping packages with a suitable length was used. According to [11], on average, the length of invite message was 728 Bytes and that of the OK message was 573 Bytes. Therefore, the length of transmitted Ping packages was selected to fit the length of SIP message as much as possible.

$$RTT = Delay_{(invite\text{-}prop)} + Delay_{(OK\text{-}prop)} \qquad (4)$$

(2) To determine UAS processing time, the following method is suggested:
In the online form, the UAC node will be aware of the time of queuing and processing in the UAS node. This amount is called $Delay_{(Queue/Proc)}$.
Generally, three conditions can be predicted for the transmitted invite message:
- Retransmission despite the existence of delay in the network,
- Retransmission when the message is corrupted, and
- Retransmission when loss occurs.

Each of them was separately analyzed as follows; also, in each case, the appropriate value of timer T_1 was obtained:

4.2. RETRANSMISSION DESPITE THE DELAY IN THE NETWORK

According to Equation 3 and Equation 4, the delay in receiving the reply and proper retransmission time are estimated by Equation 5 and Equation 6.

$$Delay_{(invite \rightarrow ok)} = RTT + Delay_{(Queue/Proc)} \qquad (5)$$

$$T_1 = RTT + Delay_{(Queue/Proc)} \qquad (6)$$

Hence, RTT and the time required for queuing and processing in UAS node are the determiners for the invite message retransmission time.

4.3. RETRANSMISSION WHEN THE MESSAGE IS CORRUPTED

As already mentioned, in case any error exists in the invite message (or temporary reply), it must be transmitted immediately; i.e., in the event of a message malfunction, the message should be transmitted again as soon as possible (possible minimum time). Therefore, to estimate the minimum time required for retransmission, the following procedure should be followed.

When the reply is not transmitted, the session will be terminated until the termination of timer T_1, and timer B will be 64 times more than Timer T_1. So, timer T_1 can be 1/64 of the time required for receiving the reply. In other words, if timer

T_1 is considered to be smaller than this amount, the session will be expired before receiving the reply. Thus, the minimum time of the timer is shown in Equation 7.

$$T_{1(Minimum)}=(RTT + Delay_{(Queue/Proc)})/64 \qquad (7)$$

4.4. RETRANSMISSION IN THE EVENT OF LOSS

In the case of the existence of congestion in the network, loss will happen, which results in the loss of the packages. If the invite message is lost (or the reply loss), the invite message retransmission should be delayed and the congestion and negative effects will be reduced through reducing the retransmission rate. For this reason, to estimate the time required for retransmission, a coefficient K (K > 1) was considered for retransmission:

$$T_1 = K \times (RTT + Delay_{(Queue/Proc)}) \qquad (8)$$

4.5. ESTIMATING ADAPTIVE TIMER T_1

For each condition mentioned in Section 4.1, an appropriate amount of timer T1 was estimated. In order to suggest the most appropriate timer, suitable weight had to be related to each of the delay components. Final amount of the timer was the total of the weight components. Now, to have the percentage of the corrupted messages, the following procedure is followed.

$$R = [P + a(1-P)] + [P + a(1-P)][P + a(1-P)] + [P + a(1-P)] [P + a(1-P)] [P + a(1-P)] + \dots$$

So,

$$R = [P + a(1-P)][1+ [P + a(1-P)]+ [P + a(1-P)]+ [P + a(1-P)]+\dots]$$
$$R = [P + a (1-P)][1+ R]$$
$$a (1-P) = [R / (1+R)] - P$$

So, the percentage of the defective messages was obtained from Equation 9.

$$a = \{(R / (1+R)) - P\} / (1-P) \qquad (9)$$

Where P is the network packet loss rate, ($1-P$) is the probability of the messages being obtained, a is the percentage of the defective messages, and R is the percentage of the retransmitted messages.

Considering Equations 6, 7, 8, and 9 relations, the proposed adaptive timer can be expressed as follows.

$$Adaptive\ T1 = (1-P)\ (1-a)\ (RTT + Delay_{(Queue/Proc)}) + a\ (1-P)\ ((RTT + Delay_{(Queue/Proc)})/64) + P\times (K\times (RTT + Delay_{(Queue/Proc)}))$$
$$Adaptive\ T1 = (RTT + Delay_{(Queue/Proc)})\ [(1-P)\ (1-a) + a\ (1-P)/64 + PK]$$
$$Adaptive\ T1 = (RTT + Delay_{(Queue/Proc)})\ [(1-P)\ (1-a+ a/64) + PK]$$

Hence,

$$Adaptive\ T1 = (RTT + Delay_{(Queue/Proc)})\ [(1-P)\ (1-63/64\ a) + PK] \qquad (10)$$

Where K ($1<K$), the invite message retransmission rate reduction coefficient.

5. EXPERIMENTING THE PROPOSED TIMER, RESULTS AND ANALYSIS

In this section, the proposed accuracy performance of the timer is studied by representing three different scenarios, as follows.

Scenario 1. Loss rate= 5% and RTT = 600 msec. At the beginning, based on the default value of the timer R, R = 110%. Meanwhile, the coefficient *K* of 2 was assumed. In these circumstances and based on the proposed formula, the best time of timer T_1 was specified to be approximately 700 msec. Figure 1 and Figure 2, respectively represent the SST (Session Start up Time) changes and the percentage of retransmission packages to the timer changes.

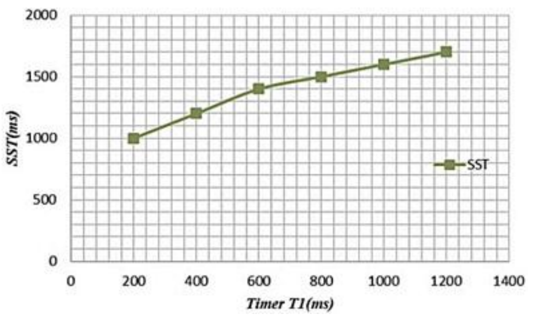

FIGURE 1. Average changes of session initiation time to session timer T1 in msec.

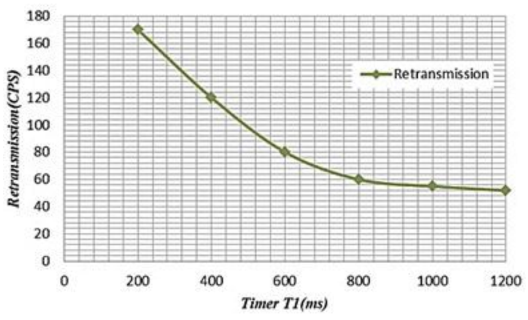

FIGURE 2. The invite message retransmission changes to timer T_1 changes in msec.

As can be observed, in Scenario 1 where timer T1 was more than about 700 msec, increased SST was resulted and no great recovery would occur; if it was less than 700 msec, SST might be reduced; however, the invite message retransmission would be greatly increased.

Scenario 2. The proposed adaptive timer is compared with the fixed timer and default T1 as equivalent to 500 msec, with condition that the network delay was 100 msec and the amount of the network loss was variable. Figure 3 and Figure 4 demonstrate the SST changes and the invite message retransmission in terms of loss changes for the proposed fixed and adaptive timer, respectively. Accordingly, the proposed formula showed appropriate improvement in SST. In fact, as could be anticipated, it also increased the retransmission amount of the message, which was very minor. According to the rate of improvement, SST seemed to have an appropriate effect on the efficiency of SIP protocol.

Scenario 3. The proposed adaptive timer is compared with fixed timer and timer T1's default, as equivalent to 500 msec in the circumstances where rate of conversation production started from a low amount and continued to heavy conversation rate up to 1800 contacts per second. Figures 5, 6, 7, 8 and 9 show the passage in the form of a function of the rate of incoming call requests in both cases, indicating that despite the proposed adaptive timer, the proxy passage could be kept at the maximum capacity; i.e. improving performance while facing the overload.

As can be seen in Figure 7, proxy's queue is approximately empty before occurring overload, since every message is drawn out of queue and processed upon reaching to proxy. Although in overload conditions many packages are consistently waiting in the queue to receive service. Nevertheless, applying overload control method, could decrease memory usage considerably and delay its exponential growth rate up to about 1500 cps.

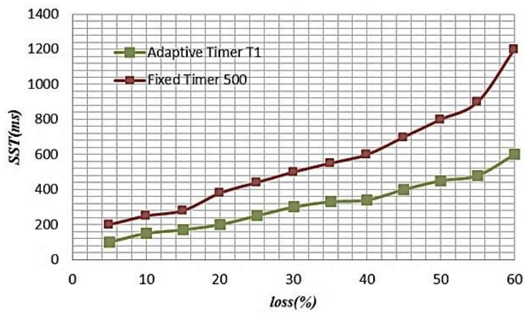

FIGURE 3. Comparing average time changes of the session initiation with loss changes in adaptive timer T_1 and 500 msec fixed timer

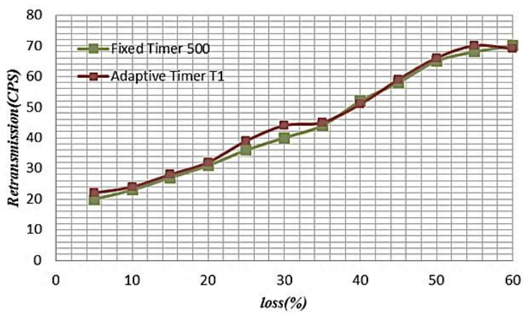

FIGURE 4. Comparing the invite message retransmission changes with loss changes in adaptive and fixed timer T_1

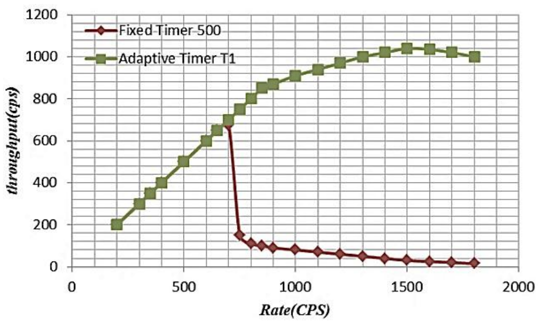

FIGURE 5. Throughput with adaptive and fixed timer T_1

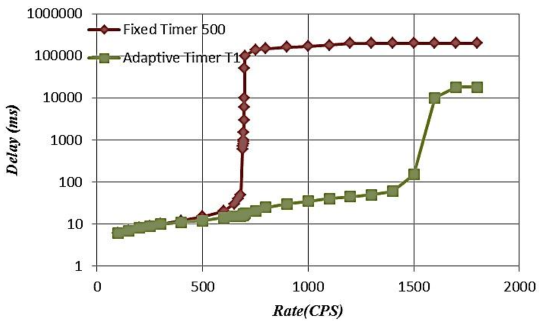

FIGURE 6. Average delay with adaptive and fixed timer T1

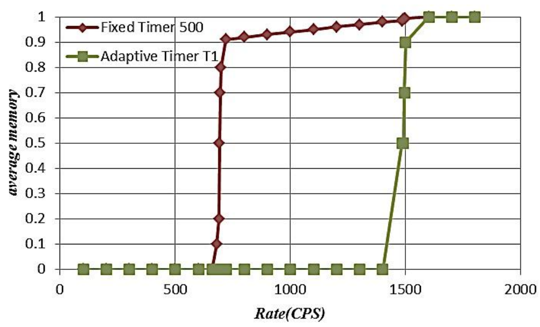

FIGURE 7. Average memory utilization with adaptive and fixed timer T1

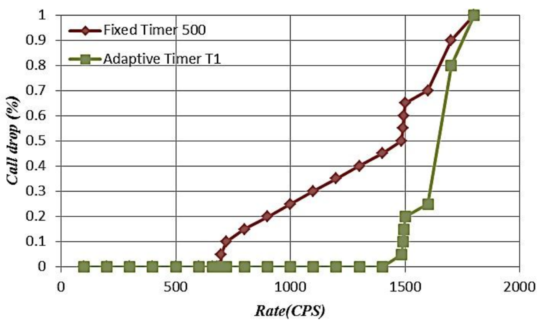

FIGURE 8. Call drop with adaptive and fixed timer T_1

ISSN: 2252-4274 (Print)
ISSN: 2252-5459 (Online)

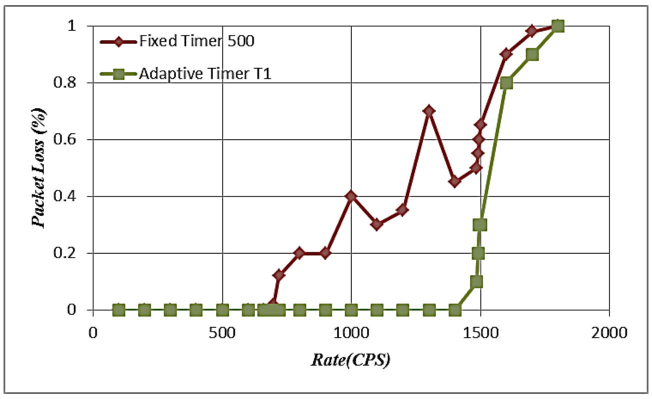

FIGURE 9. Packet Loss with adaptive and fixed timer T_1

6. CONCLUSION

According to the amount of loss and network delay as well as SIP traffic, it is concluded that a proper selection of layer protocol could be effective in the time required for the session initiation and consequently efficiency. Using TCP could prevent any call drop; however, it would result in more SST and utilization of the bandwidth and CPU in the server while the cost of using UDP will be the existence of a slight call drop. In the presence of loss, using TCP was also optimal both in terms of SST and call drop. Moreover, considering these experiments, great effect of appropriately regulating timer T1 was noticed. As the smaller the amount of timer T1, the more the reduction of SST. However, its cost would increase with the increase of the invite message retransmission with a negative impact on the use of bandwidth and CPU in the server.

It is worth mentioning that timer T1 could not be considered smaller than 1/64 time required for sending the invite message and receiving a reply, because it would cause the session timeout. To make the SST time optimal (considering the amount of invite message retransmission), timer T1 must become larger; in the case of RTT increase, the time required for processing on the server along with loss would increase and, in case of the increased percentage of the message malfunction, timer T1 must become smaller. Meanwhile, the proposed adaptive timer, which was suggested considering the amount of loss and delay and message malfunction, was evaluated and it was shown that it reduced the SST to the proper extent and did not have any considerable negative impact on the message retransmission.

REFERENCES

[1] J. Rosenberg, H. Schulzrinne, "SIP: Session Initiation Protocol," RFC 3261, 2002.
[2] M. Cortes, J. O. Esteban, H. Jun, "ISE03-3: Towards stateless core: Improving SIP proxy scalability," Proceedings of IEEE Global Telecommunications Conference, GLOBECOM '06, 2006, pp. 1-6.
[3] E. Nahum, J. Tracey, C. P. Wright, "Evaluating SIP server performance," Proceedings of SIGMETRICS '7 Conference, June 2007, vol. 35, Issue 1, pp.349-350.

[4] J. Rosenberg, H. Schulzrinne, "Session Initiation Protocol (SIP): Locating SIP servers," RFC: 3263, 2002.

[5] R. Jain,V. K.Gurbani , R. Jain, "Transport protocol considerations for session initiation protocol networks," Bell Lab Technical Journal, 2004, vol. 9, no. 1, pp. 83-97.

[6] B. Campbell, 1. Rosenberg, H. Schulzrinne, C. Huitema, D. Gurle, "Session Initiation Protocol (SIP) Extension for Instant Messaging," RFC 3428, 2002.

[7] G. De Marco, G. Lacovoni, "A technique to analyse session initiation protocol traffic," Proceedings of the 11th International Conference on Parallel and Distributed Systems - Workshops (ICPADS'05), 2005, vol. 2, pp. 595-599.

[8] Y. Wang, "SIP overload control: a backpressure-based approach," ACM SIGCOMM Computer Communication Review, 2010, vol. 40, pp. 399-400.

[9] Y. Hong, C. Huang, and J. Yan, "Impact of Retransmission Mechanism on SIP Overload: Stability Condition and Overload Control," Journal of Networks, 2012, vol. 7, pp. 52-62.

[10] A. Abdelal and W. Matragi, "Signal-based overload control for SIP servers," presented at the 7th IEEE Consumer Communications and Networking Conference (CCNC), 2010, pp. 1-7.

[11] H. Fathi, S. S. Chakraborty, R. Prasad," Optimization of SIP Session Setup Delay for VoIP in 3G Wireless Networks," IEEE Transaction on mobile computing, September 2006, vol 5, no. 9, pp.1121-1132.