Volume 3, Issue 11, November 2013 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

Controlling Overload in Networks with SIP Redirect Servers

Kiran Kumar Guduru*
Senior Software Engineer
Samsung R&D Institute-Bangalore (SRI-B)
Bengaluru, India

Dr. Usha J
Department of Master of Computer Applications
R.V College of Engineering
Visvesvaraya Technological University, India

Abstract— The Session Initiation Protocol is an application level signalling protocol used for locating end users, establishing, modifying and closing the multimedia sessions between two or more users. It enables features like inviting new participants to existing sessions, providing facility to add or remove media from the session. With the evolution of telecommunication networks, this protocol has been adopted as the basis for IP-Multimedia Subsystem (IMS) networks. Internet Telephony is seeing rapid growth in recent times and Session Initiation Protocol is most widely used for signalling purpose. However, existing networks are not well equipped to handle overload conditions. This paper discusses the behaviour and controlling of overload conditions in SIP networks with redirect networks in detail and proposes a solution to control overload situation while using redirect servers. Simulation studies have revealed that more attention needs to be given to support and implement standardized overload control techniques to redirect servers. Different approaches to implement a standardized way of controlling overload while using redirect servers in SIP signalling networks is discussed.

Keywords— SIP, redirect servers, overload, congestion, transactions.

I. INTRODUCTION

The Session Initiation Protocol (SIP) [1] is an application level signalling protocol. It enables features like inviting new participants to existing sessions, providing facility to add or remove media from the session. SIP does not specify any information about the session description. It is used to carry other multimedia and transport protocols like Session Description Protocol (SDP) in the body of the SIP message. With the evolution of telecommunication networks from 1st generation networks to 4th generation networks, 3GPP has identified and adopted SIP as the basis for IMS signalling networks. SIP signalling networks can be interconnected with SS7 signalling networks through gateways to provide backward compatibility.

SIP make use of elements called proxy servers, redirect servers, back-to-back user agents and presence servers for registering users and their locations, routing requests from Callers location to Callees location, authenticating and authorizing users and updating the user's availability state. SIP provides the flexibility of combining one or more entities as individual component servers or can be implemented in a single server that can perform all the required services.

A Location service, considered to be the database of user's information, stores the user's information and location details during the time of user's registration through SIP redirect servers or SIP proxy servers. A user can register and or update one or more possible locations in the Location service. SIP redirect or SIP proxy servers interact with Location service to update and or obtain the information about the SIP Callee's possible location and availability state. A redirect server receives the request from the outgoing proxy server, queries the location information of the Callee from the Location service and conveys the single or multiple possible destination location of the Callee through SIP redirect response. Based on the location information in the redirect response, the core proxy will route the SIP signals to the possible destination locations of the Callee. Querying about the location information and validating the possible locations takes more processing time and is so termed as costly operation. So redirect servers are more preferred instead of implementing the same mechanism in the outgoing proxy server to increase the throughput of proxy servers. Some implementations may optimize this process by querying the location information of the Callee for the first request, and the same routing information will be used for the other transactions for the rest of the session.

SIP prefers UDP for singling transport. Since UDP is non-reliable transport protocol, SIP provides reliability of messages from its application layer by monitoring and retransmitting the messages, if no response is received within the specified time period. If the complete network or any node/hop in the network is congested and not able to respond within the specified time period, it is assumed that those packets were missed or corrupted and will be retransmitted. Because of these unnecessary retransmissions, the whole system will get more congested resulting in the decrease of throughput and increases the response time.

II. SIP NETWORKS WITH REDIRECT SERVERS

The SIP user profiles, received through registration requests, are stored in databases. Information in these databases is used for identifying the end user location, routing a mobile originating or a mobile terminating call and user specific features like call blocking, call forwarding etc. This process of registration and processing of user specific information

can be done either through proxy servers or through redirect servers. This process takes more processing time, so redirect servers are used to reduce this burden on proxy servers. SIP redirect servers provides the routing path information and destination address for the proxy servers for routing the call. If there is more than one routing path to a destination address, then redirect servers will provide the routing path information to the outgoing proxy, such that the load will be distributed among those proxies and reduces the load on each proxy server. It is expected that the load from redirected server will be distributed equally or with operator specific predefined preferences among all the available paths to avoid overload situation. This approach has the following drawbacks. In peak hours, the load distribution is good and acceptable. But for the normal loads, where all the load can be processed by less number of proxy servers than the load is being distributed, such type of default distribution of load is not encouraged. It consumes more power and system resources unnecessarily. Since routing algorithms has more number of options, it takes more processing time to identify and allocate the correct path and increases the average response time. This may result in reduced throughput and increases the number of unnecessary retransmissions. In real scenarios there are chances that some proxy servers can process more calls than the others. The reasons for such type of behaviour are, state-less proxies can process more calls than that of state-full proxies. If the next hops of routing path are more overloaded, or a particular path experiencing more congestion, then the number of calls processed by a particular proxy in that routing path will be reduced. So it is not expected that either all proxy servers in different paths can process equal number of calls or can process a predefined number of calls all the times.

So distribution of load should be dynamic among different paths in different times and with different loads. Dynamic overload control mechanism increases the system capacity, throughput, and reduces the unnecessary power consumption in times of fewer loads on the network. The following section explains the hop-by-hop dynamic overload control mechanism in SIP signalling networks.

III. SIP OVERLOAD CONTROL

SIP Overload Control (SOC) is a hop-by-hop overload control method [1, 2, 3], which is being standardized by the IETF SOC working group. SOC specifies a standardized method for communicating overload information of a SIP server, to its previous SIP server. The previous SIP server is termed as client server in this perspective, and that of next hop server as simply SIP server or server node.

At the time of session establishment, the client server initiates the overload control handshake through SIP VIA header as specified in [2], which includes the parameters like overload control algorithm, validity, and tolerable/controlled load and sequence number. Server processes this request and sends its negotiated possible response to this request in the earliest possible SIP response that is being sent to that particular client server to complete the handshake. Once the handshake is completed, server identifies the overload condition using the mutually agreed overload algorithm and indicates the overload information to the client server, in the SIP responses that are being sent to that client server. This message includes the number of requests that are tolerable at the server, time period for that controlled load and the sequence number. Based on the information, client server reduces the overload on the next hop server, and redirects the remaining load to other paths or drops processing the excess load, depending on the routing schemes.

In SOC, many client servers can be connected to a single server node. Server node identifies the load received from each client node, and sends the overload control information dynamically to the corresponding client server from which more loads is being received. If all the clients are sending equal amount loads, then the server node sends the overload control information to one or more client servers based on the priorities or predefined client servers or configured client servers. The overload control algorithm negotiated with each client node is not mandatory to be same; it may be same or different for each of the client nodes connected. In such situations, the server node has to run many overload control algorithms.

It is not mandatory for all the servers in a SIP network to implement SIP Overload control mechanism. Any server in the network is free to implement its own customized overload control algorithm, to support backward compatibility for the SIP servers that are not implementing SOC. If the next/previous server hop is not supporting SOC, the client/server node, which may act as the server/client node for other servers, can implement a self overload control algorithm. The same method is applicable for any of the SIP server supporting the SOC. The client server nodes are also capable of waking up a sleeping server node, which has established a connection with that client server node and is not receiving load from its corresponding client server nodes, to receive the load in the overload conditions. The mechanism described in this method does not require any extra messages for indicating the overload condition to its pervious hop. It uses very few extra parameters in the existing headers to covey the overload information. Next section explains the limitations of SIP Overload Control.

IV. LIMITATIONS OF SIP OVERLOAD CONTROL

SIP Overload Control (SOC) specifies the standard for communicating hop-by-hop overload information in a single direction, from server node to client node. It does not specify mechanism to convey the overload state of a client server to its immediate server node. It does not specify how to communicate the overload state of one server node to another next hop server node via a client server.

In some network architectures, overload controlling or load balancing intelligence is present in redirect servers. Redirect servers always acts as server nodes. When the overload control or load controlling intelligence is present in SIP redirect server, the client server interacting with both redirect server and that of a server node in SOC method, has no mechanism to convey the overload information of the other server node to the redirect server. So there is a need for communicating the overload state of a server node to a redirect server via the client server. Otherwise the client server

has to drop the calls that are redirected by the SIP redirect server to the overloaded server after exceeding the overload limit. Next section proposes the method for communicating the overload information of an overloaded server node to the redirect server.

V. SIP OVERLOAD CONTROL WITH REDIRECT SERVERS

This section proposes a method for communicating the overload control information from a client server to a redirect server. The conveyed overload information may be the overload state of the client server node itself or it may the overload state of the next hop server node of the client server in the network. After receiving the overload information from the client server, redirect server will redirect the load to the destination through appropriate path, by following the specifications specified in [5].

In the proposed system, the client server identifies the redirect server by following the specifications defined in [10], using SIP Request-Disposition header. After identifying the redirect server, client server initiates the overload control handshake specifying that the client server will send the overload information to the SIP redirect server by specifying the overload control information received from the particular server and the corresponding SERVER_ID of the SIP overload SIP server. This kind of handshake should not be initiated with the servers other than redirect servers to avoid the security breaches. Once the handshake is completed, the client server communicates the overload information received from the server node to the redirect server. When the redirect server receives the overload information of any server node in the network path from the client server node, it redirects the excess load beyond the overload limit for the specified time mentioned in the overload information through the alternate path. After completing the time interval, redirect server again starts sending the entire load through the same old server node path. This method will decrease the load on the overloaded servers, when the redirection intelligence is present in the SIP redirect server.

From the simulation environment it is observed that, by regulating the load on the network at redirect server according to overload information received from the client server, the response time of SIP transactions is very much reduced with the increase in the throughput. Resource utilization at original server node is also increased.

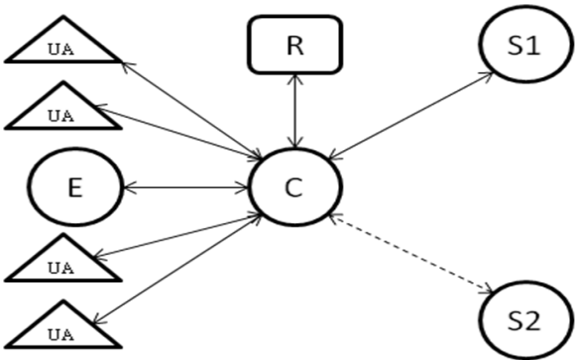

Fig 1 SIP Network Topology

Fig 1 shows the network setup of the simulation model of the proposed system, overload control with SIP redirect server. In the figure E and UA represents the edge server and the individual users or User Agents respectively. Edge servers are the SIP servers which receive the load from individual user agents and forward it to client server or core server represented by C. In this perspective client server and core server are used alternatively. C redirects all the requests from E to redirect server R and receives the SIP 3xx, redirect, response, based on which it routs the load to Gateway servers S1, S2. In the existing system, client server receives the requests from User Agents and Edge servers and forwards it to the redirect server to enquire about the destination address and or path. Redirect server gives all the available destination address in a SIP 3xx response. In this system, redirect server has no information about the overload states of the Gateway servers. So to avoid the overload on the servers, it distributes the load between the servers. The load distribution method assumes that all the calls will be redirected to a particular server, until it reaches the fixed configured value. In this simulation it is assumed to be hundred transactions per second. Beyond this limit, excess calls are redirected to the other available server. In our simulation model, the load is distributed among the two Gateway servers.

In the proposed system client server receives the requests similar to that in the existing method and forwards it to the redirect server. Redirect server indicates to forward all the requests to Gateway server S1. In the first request to the redirect server, client server sends the required parameters to identify, whether the server is redirect server or a normal server node, as specified in [10], to identify whether the server is a SIP redirect server or other kind of SIP server. Redirect server answers to this query in its response with its type and domain to which the server belongs to. After identifying that the server node is a redirect server and after confirming that the redirect server belongs to the same domain, client server initiates the overload control handshake, by sending the required parameters defined in [2], in the subsequent request. Redirect server processes the request and sends the acceptance confirmation for overload control in the subsequent response. Simultaneously client server initiates a similar kind of overload control handshake specified in [2] with the Gateway servers in its first request. Identification of server type and the domain to which that server belongs to are not required for Gateway servers. It follows the same rules as that for the normal overload control method. Now client server completed the overload control handshake step with the redirect server and the Gateway server.

According to the overload control algorithm negotiated in the overload control handshake, Gateway server, monitors for its overload state. Gateway server S1, after reaching the overload state, intimates its state, including the sequence number, the number of transactions up to which it can process during the overload state and the time interval up to what it likes to receive the controlled load to the client server in the responses. Client server forwards the overload state of the server node to the redirect server, in the next request intended to redirect server, with an extra parameter, SERVER_ID, to indicate the overloaded server. After receiving the overload information of the server S1, it forwards the requests to S1 until it reaches the limited number of transactions specified in the overload information. Beyond that limit it forwards the new calls and their corresponding transactions to the other Gateway server S2. Client server may negotiate for the overload control handshake with server S2, once it starts sending the requests to S2, similar to that of Gateway server S1. After completing the time period specified in the Gateway server S1, redirect server again start sending the entire load to server S1.

In the simulation model, it is assumed that, when the queue length per second exceeds two hundred transactions per second, Gateway server detects that, it reached the overload state and sends the overload information to the client server. The overload information contains the sequence number, tolerable controlled load, and hundred transactions per second for time interval of one second. If the number of transactions exceeds four hundred transactions per second, then the overload time interval is increased to two seconds. For simulating both the existing method and the proposed method, a simulation environment is created which can generate a random distribution of load to the client server from twenty five transactions per second to three hundred transactions per second. The call length is distributed randomly varying from zero seconds to greater than thirty seconds. The distribution of these calls and call lengths follows the normal distribution algorithm.

$$f(x) = 1/\sigma\sqrt{(2\pi)}\ \exp\text{-}[(x-\mu)^2/2\sigma^2]$$

Where μ represents the mean and σ represents the standard deviation. In the simulation model, SIPP simulator is configured to generate calls at a mean (μ) call rate of seventy calls (two hundred and ten transactions) per second with a standard deviation (σ) of seven calls (twenty transactions) per second, and with a variance (σ^2) of fifty calls (one hundred and fifty transactions) per second.

The method specified in the proposed system will reduce processing time at redirect server for identifying the correct server path. For most of the time, when the Gateway server is not experiencing the overload state, redirect server can forward the entire load to the same path, without any extra processing for identifying or distributing the load.

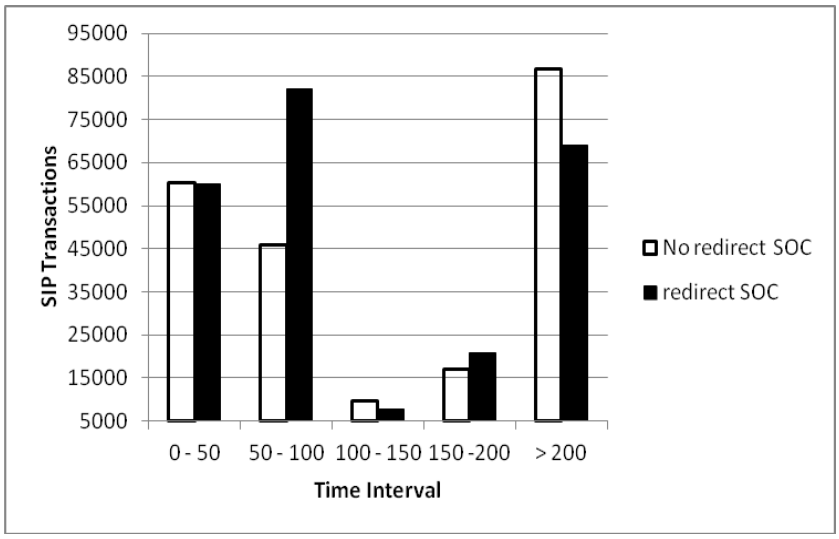

Fig 2 Request Response Round Trip Times

In Fig 2 X-axis represents the time intervals for the response time of the SIP transactions and Y-axis represents the number of transactions processed within the respective time interval. Empty bars represents the response times of the existing system, while the dark and filled bars represents the response times of the proposed system. From Figure 2 it is observed that the response time of most of the transactions in the proposed system lies in the interval of 50-100ms where as the response time in the existing system for most of the transactions is greater than 200ms. According to SIP [] the value of T1 timer is 500ms and that of window timer for reliability is 200ms. For SIP non-invite transactions, if the response time is greater than 500ms, timer T1 will expire, then the server assumes that the request is either dropped or corrupted in the network and the application of client server retransmits the requests. These unnecessary retransmissions that arise due to congestion in an overloaded server again increase the load on the server and reduce the through put.

For SIP Invite transactions, since it is the first request which has to identify the routing path, the processing time is expected to be more when compared to other transactions. Any server which receives the SIP Invite transaction, if it is not able to send the response with in 200ms, window timer, the server node will send a SIP 100 Trying response, indicating the client that it has received the request properly and to stop further retransmissions from the client server. For the proposed system, since most of the responses are received before 200ms, server will not send any 100 trying

responses. For the existing system, since the response time for most of the requests is greater than 200ms, it is expected that most of the invite transactions might have received the 100 trying responses, which again increases the server nodes processing time, to generate and send these responses and the extra network traffic due to these 100 trying responses.

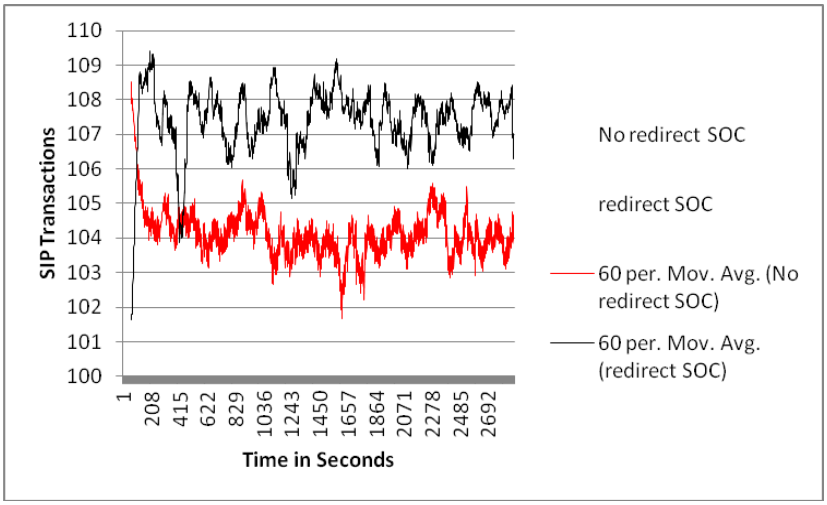

Fig 3 Minute Averaged Transactions on Primary Server

Fig 3 represents the minute averaged transactions of load redirected to the primary Gateway server S1 by using the dynamic overload control method and static overload control method. In figure 3 Y-axis represents the averaged number of transactions redirected to the primary server. Black coloured line indicates the load distribution of the proposed system. Red coloured line indicates the load distribution of the existing system. From figure 3 it is observed that the primary server is used more efficiently than that in the existing system by redirecting more number of number of transactions to the primary server. In proposed system for less number of transactions received in particular time periods, the secondary server is not used and will change to sleeping mode. When the Gateway server S1 receives more number of calls than that it can process, then only server S2 is activated to receive the excess load. But in the existing system, load is distributed irrespective of the overload of Gateway server S1. So it is always required to make all the servers active for all the times. The proposed method reduces the power usage and utilizes more system resources.

From the analysis of the simulation model, it is observed that throughput of the proposed system has increased by 20% when compared to that of the existing system. Call failure rate has also been decreased from 2% in the existing method to 0.5% in the proposed method. Number of retransmissions of SIP transactions in the proposed method has been decreased by 61% in the proposed method than that of in existing method. These retransmission of requests will utilize more network bandwith and processing time of server nodes.

VI. CONCLUSIONS

Internet telephony is evolving with a rapid growth, with increased usage of data rates and by providing features like user to be always online. SIP is the widely used IP based signalling protocol. Because of the less transmission costs with internet telephony, SIP is considered as the IP based signalling protocol, and standardized by 3GPP for IMS networks. SIP maintains the user profiles.

SIP is used for routing the signalling information from caller to Callee. SIP network architectures can employ redirect servers for retrieving, updating and modifying the routing information, which is a time consuming and costly process and reduces the burden on Proxy servers. SIP offers backward compatibility with traditional PSTN networks.

SIP Overload Control is an IETF working group that is working for standardizing the overload control in SIP networks. It sends the server overload information to client server to reduce the load received for some duration of time, to avoid congestion. If redirect server is redirecting the calls to this overloaded server, through proxy server other than overloaded server, then it will result in call blocking. To avoid these call blocking scenarios, it is required to inform the overload state of the SIP server to SIP redirect server. This will reduce the load redirected to the overloaded server, by redirecting the extra calls to some other servers, which are not in overloaded state. This will reduce the processing time, number of retransmission of requests due to delayed responses, bandwidth usage and increases the throughput.

REFERENCES

[1] Masataka Ohta, "*Overload Control in SIP Signaling Network",* International Journal of Electrical and Electronics Engineering, 2009.

[2] Volker Hilt, Indra Widjaja, Holmdel, Murray Hill, "*Design Consideration for SIP Overload Control*", IETF RFC 6357, 2008.

[3] Volker Hilt, Indra Widjaja, Holmdel, Murray Hill, "*Controlling Overload in Networks of SIP Servers*", IEEE Journal, 2008.

[4] M. Lulling, J. Vaughan, "*Reliability and Congestion Control for VoIP Signalling Transport*", 2006.

[5] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnson, J. Peterson, R. Sparks, M. Handley, E. Schooler, "*SIP: Session Initiation Protocol*" IETF RFC 3261, 2002.

[6] Pavel O. Abaev, Yulia V. Gaidamaka, Alaxander V. Pechinkin, Rostislav V. Razumchik, "*Simulation of Overload Control In Sip Server Networks*", Proceedings 26th European Conference on Modelling and Simulation.

[7] Pragyan Verma, Preeti Sharma, Sattyam Kishore Mishra, "*Dropping of Call Due to Congestion in Mobile Network*", Journal of Computer Applications (JCA), Volume V, Issue I, 2012.

[8] D. Papadimitriou, M. Welzl, M. Scharf, B. Briscoe, "*Open Research Issues in Internet Congestion* Control", IETF RFC 6077, Febraury 2011.

[9] V. Gurbani, H. Schulzrinne, "*draft-ietf-soc-overload-control*-09", IETF Internet Draft, July 2012.

[10] J. Rosenberg, H. Schulzrine, P. Kyzivat, "*Caller Preferences for the Session Initiation Protocol (SIP)*", IETF RFC 3841, 2004.

[11] Zohair Chentouf, "*SIP Overload Control Using Automatic ClassificationI".*

[12] J. Postel, "*User Datagram Protocol*", IETF RFC 768, 1980.

[13] Charles Shen, Henning Schulzrinne "*On TCP-based SIP Server Overload Control*", Technical Report CUCS-048-09, 2009.

[14] J. Rosenberg, "*Requirements for Management of Overloading in the Session Initiation Protocol*", IETF RFC 5390, 2008.

[15] D. Willis, B. Campbell, "*Session Initiation Protocol Extension to Assure Congestion Safety draft-ietf-sip-congestsafe-02*", IETF DRAFT, 2004.

[16] H. Khartabil, "*Congestion safety and Content Indirection draft-khartabil-sip-congestionsafe-ci-02.txt*", IETF DRAFT, 2003.

[17] M. Whitehead, "*GOCAP-ones standardized overload control for next generation networks*", BT Technology Journal, Vol 23 No 1, 2005.

