Middle-East Journal of Scientific Research 6 (1): 88-92, 2010
ISSN 1990-9233
© IDOSI Publications, 2010

Two Stage Architecture for Load Balancing and Failover in SIP Networks*

Alireza Karimi, MehdiAgha Sarram and Mohammad Ghasemzadeh

Department of Electrical and Computer Engineering, Yazd University, Yazd, Iran

Abstract: Session Initiation Protocol (SIP) proxy server is the most important component in SIP networks. In large SIP-based networks overload on this server can cause problems. Also if this server fails, it will be impossible to make new calls. To prevent overload problem and have failover capability, we proposed and implemented a two stage architecture. At the first stage there is a cluster of dispatchers and at the second stage there is a cluster of SIP proxy servers. We implemented our algorithm for load balancing in dispatchers and we used a probing mechanism for failover. We used Opensips, an open source SIP server as our dispatchers and our back-end SIP servers. We also used SIPp, an open source SIP traffic generator to test our network with high rate of calls. In our algorithm we considered both round trip time and number of connections to SIP servers to do load balancing. We probed back-end SIP servers every 30 seconds to see if they are alive and to measure round trip times. The comparison of our algorithm with Opensips algorithm for load balancing showed an improvement about 20% in throughput.

Key words: SIP (Session Initiation Protocol) · Load balancing · Dispatcher · SIP proxy server · Failover

INTRODUCTION

In compare with PSTN (Public Switched Telephone Network), voice over IP availability and scalability is too low. Some of the problems in this way come from the nature of IP networks. For example "up time" in PSTN is about 99.999% that is so difficult for IP networks to reach this [1]. The Session Initiation Protocol (SIP) [2] is a distributed signaling protocol for IP telephony. SIP-based telephony services have been proposed as an alternative to the classical PSTN and offer a number of advantages over the PSTN. SIP as the most important protocol for multimedia and voice over IP has some components that have important roles in SIP networks availability and scalability. The most important component in SIP networks is SIP proxy server. Although SIP proxy server is only responsible for signaling and voice traffic doesn't pass through this server, but in large SIP networks even signaling load can cause overload problem for this server. While individual servers may be able to support hundreds or even thousands of users, large-scale ISPs need to support customers in the millions, it is text-based and it is similar in many aspects to the well-known protocols such as HTTP and DNS [1]. For example it identifiers are similar to email addresses in the form of *user@domain*.

What makes SIP easy to understand comparing to previous multimedia protocols like H.323 and more powerful is that SIP proxy server of a domain is responsible for forwarding the incoming requests destined for the logical address of the form *user@domain* to the current transport address of the device used by this logical entity and forwarding the responses back to the request sender. SIP is a transaction-based protocol designed to establish and tear down media sessions, mostly referred as calls [3]. When two SIP phones want to communicate, all signaling messages pass through SIP proxy server as shown in Fig. 1.

If for any reason SIP proxy server fails, it will be impossible to start new calls and to do billing and accounting for current calls, because at the end of a conversation, phones generate some messages to inform

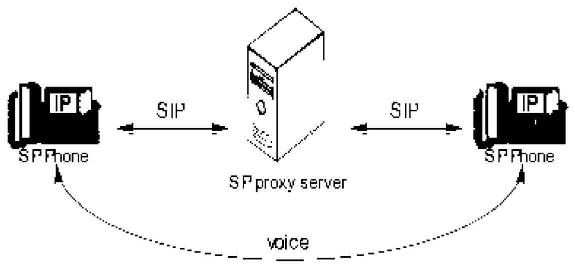

Fig. 1: SIP proxy server role

Corresponding Author: Mehdi Sarram, Department of Electrical and Computer Engineering, Yazd University, Yazd, Iran.
E-mail: mehdi.sarram@yazduni.ac.ir.

*This research work was partially supported by Iranian Telecommunication Research Center (ITRC).

the SIP proxy server of the end of conversation [1]. To increase service availability and scalability and to do load balancing and failover we proposed and tested a two stage architecture. In this architecture there is a cluster of dispatchers at the first stage and there is a cluster of SIP proxy servers at the second stage. We implemented our load balancing algorithm in dispatchers and we used probing mechanism for health checking of back-end SIP proxy servers. In a clustered architecture, there are three issues need to be concerned:

- Single point of failure problem
- Health monitoring of back-end SIP proxy servers
- Load balancing method for SIP proxy servers

In our approach we considered all these issues. In section 2 we described some of the failover and load balancing techniques for SIP servers. In section 3 we described our architecture and our algorithm for load balancing and failover. Experimental results are shown in section 4 and conclusion is placed in section 5.

Related Works: Failover and load balancing for web servers is a well studied problem. TCP connection migration, IP address takeover and MAC address takeover have been propos-ed for high availability. Load sharing via connection dispatchers [4] and HTTP content or session-based request redirection are available for web servers [5]. Although SIP is similar to HTTP but there are basic differences between these two protocols for example in SIP the call requests and responses are not usually bandwidth intensive, caching of responses is not useful. Here we describe some of the existing load balancing techniques in SIP.

Dns-based Load Balancing: It is possible to use DNS to do load balancing by means of DNS SRV [6] and NAPTR [7] records by using the priority and weight fields in these resource records, as shown in Fig. 2:

Priority 0 indicates that a server is a primary server so a, b and c in above are primary servers. Priority 1 indicates that that a server is a backup server. The weight column indicates that servers a, b and c will receive 60%, 20% and 20% of load respectively. Clients can use weighted randomization to achieve this distribution.

Dns-based Load Balancing with Probing Mechanism: In this method client queries a DNS server to resolve SIP proxy server's FQDN (Fully Qualified Domain Name) to an IP address. DNS server has a probing mechanism to see

Test.com	priority	weight	
_sip._udp	0	60	a.test.com
	0	20	b.test.com
	0	20	c.test.com
	1	0	backup

Fig. 2: DNS SRV records

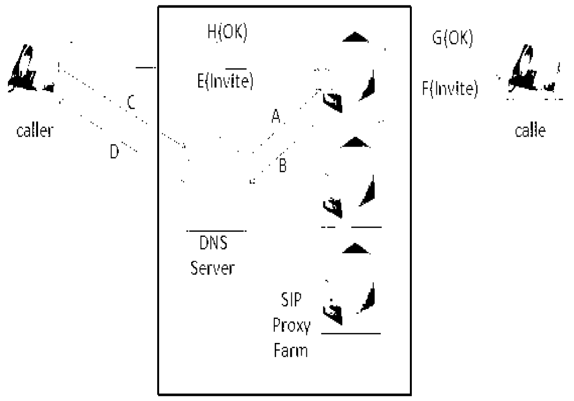

Fig. 3: DNS-based load balancing with *probing mechanism*

if SIP proxies in the cluster are fine or not. DNS server does health checking periodically, if the probed SIP proxy server is ok so DNS server chooses the IP address of this server for DNS replies and clients will use this server as their SIP proxy server until next probing [8].

The problem of this method is that it is changing DNS server's role and it is wasting resources because it is possible to have heavy loads in a short period and it is sending the entire load to one of the SIP proxy servers and rest of the proxy servers have a small amount of load.

Identifier-based Load Balancing: For identifier-based load balancing (Fig. 4), the user space is divided into multiple non-overlapping groups. A hash function maps the destination user identifier to the particular group that handles the user record, e.g., based on the first letter of the user identifier [5]. For example, s1 is responsible for the calls for users who their ID begins with a letter from 'a' to 'm' and s2 handles calls for users who their ID begins with a letter from 'n' to 'z'. A high speed dispatcher proxy the call requests to s1 and s2 based on the destination user identifier. There is no guarantee that the load on servers is equal and single point of failure still exists.

Single Controller for Proxy Servers: F5's BIG-IP [1] system is an application traffic management solution and it tries to provide high availability and reliability for SIP

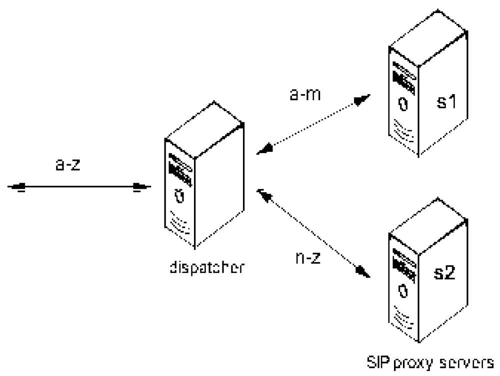

Fig. 4: Identifier-based load balancing

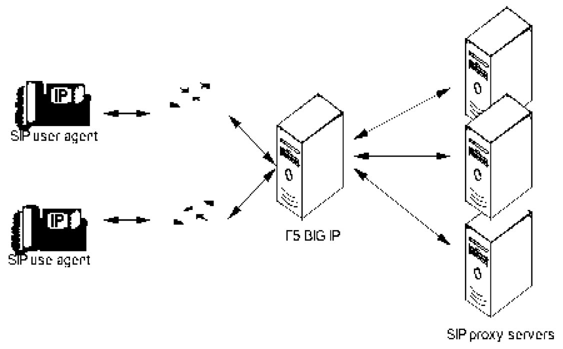

Fig. 5: F5's BIG-IP

networks. Fig. 5 illustrates F5's BIG-IP component in SIP network architecture. The BIG-IP does the advanced health check by sending SIP OPTIONS requests to SIP proxy servers. F5's BIG-IP has an important role in this SIP network architecture because it is responsible for health checking and also proxies all SIP requests to back-end SIP proxy servers. It prevents from service unavailability when a SIP server fails. However there is still a single point of failure problem in this architecture. If F5's BIG-IP itself fails the service will become unavailable.

Network Address Translation: Network Address Translation (NAT) is a network service that is responsible for translating internal IP addresses from machines inside the network to a public address used by the NAT service-essentially hiding your internal network addresses. For incoming requests NAT server can choose one of the internal SIP proxy servers. Eventually, the NAT itself becomes the bottleneck making the architecture inefficient.

Hybrid Architecture: In this architecture there are two sets of proxy servers the first set of proxy servers selected via DNS NAPTR and SRV records, performs request routing to the particular second-stage cluster based on the hash of the destination user identifier. In fact it is a combination of DNS based and identifier based load balancing. The second-stage server performs the actual request processing. Adding an additional stage does not increase the voice delay, since the media path (usually directly between the SIP phones) is independent of the signaling path. Use of DNS does not require the servers to be co-located, thus allowing geographic diversity [5].

System Design Approach: In our architecture we used two stages of servers, the first stage servers, act as dispatchers (load balancers) and the second stage servers are SIP proxy servers. SIP user agents select one of the dispatchers via DNS SRV records. Each dispatcher makes a list of SIP proxy servers every 30 seconds by probing SIP proxy servers and if a SIP proxy server doesn't respond to the probe, it will be considered dead and the dispatcher will not route requests to this server. The list is ordered based on two criteria: round trip time and number of connections to each SIP proxy server. Based on these two criteria each SIP proxy server has a grade for 30 seconds. We used number of connections because call durations aren't the same for all calls. We also used round trip time to take network status into account. It is especially useful when SIP proxy servers are in different networks. Each dispatcher has a counter for every SIP proxy server to count the number of connections to each SIP proxy server. When an "invite" message passes through dispatcher it increases the counter by one and when final "200 ok" message passes through the dispatcher -off course in reverse direction- it decreases the counter by one.

SIP dispatchers also do health checking in constant intervals, for example every 5 seconds. If any of the SIP proxy servers doesn't respond to health checking messages, dispatcher will delete that server from the list. The architecture is illustrated in Fig. 6.

For load balancing dispatcher uses ordered list and forwards a percentage of load to each SIP proxy server. The percentage of load that is forwarded to each SIP proxy server is proportional to each SIP server's grade. If number of connections to ith SIP proxy server is N_i, round trip time is RTT_i, SIP proxy server grade is G_i, a and b are two constant numbers, each SIP proxy server grade will be as follow:

$$G_i = a*(1/N_i) + b * (1/RTT_i) \qquad (1)$$

If ith SIP proxy server share is S_i and n is number of SIP proxy servers we have:

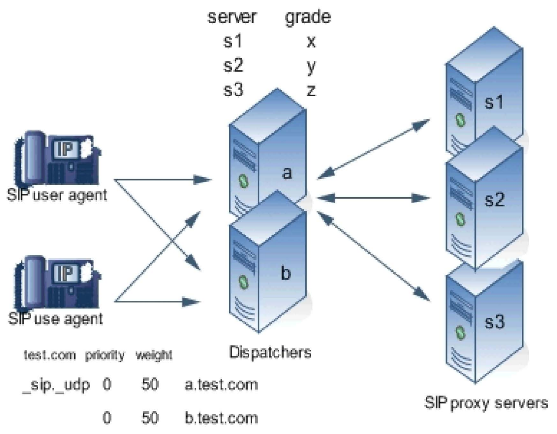

Fig. 6: System design approach

$$S_i = (100*G_i) / \Sigma (G_i) \ i=1,2,...,n \qquad (2)$$

We can adjust a and b in (1) to emphasis on N_i or RTT_i if one of them is more important for us or it is more suitable for current network condition.

Test Bed and Experimental Results

Traffic Generator: To make huge amount of calls we had to use a traffic generator. SIPp [9] was used as a SIP traffic generator in the performance evaluation. SIPp is a configurable packet generator, extensible via a simple XML configuration language. It uses an efficient event-driven architecture but is not fully RFC compliant. It includes some SIP user agent test scenarios, which were provided by SIPStone and it is a benchmark tool for evaluating SIP proxy server performance. We used version 3 of SIPp. We used one of the default UAC (user agent client) and UAS (user agent server) scenarios-"200 ok"- which have already been implemented in SIPp to test the proposed design approach. "200 ok" scenario is illustrated in Fig. 7. We also turned off the retransmission mechanism in SIPp.

Servers: For the dispatchers and back-end SIP proxy servers we used Openssips-an open source SIP proxy server. We installed Opensips on linux Debian 5. To evaluate the performance we set up a test bed like Fig. 6 but with one dispatcher and two SIP proxy servers.

Hardware: We used five computers for our test bed, two of them for SIPp traffic generator (one as UAS and one as UAC), with two virtual machines on each of them, another computer as dispatcher and two other computers as SIP proxy servers. They were connected via a fast Ethernet switch. All five computers are the same and their specifications are shown in Table 1.

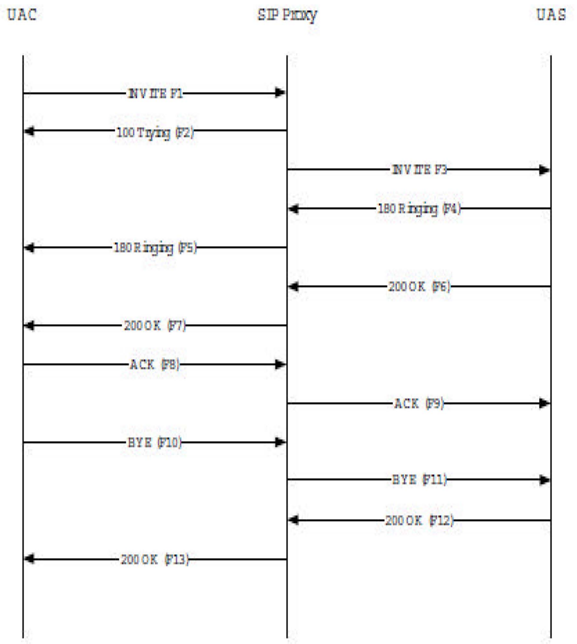

Fig. 7: Proxy 200 scenario

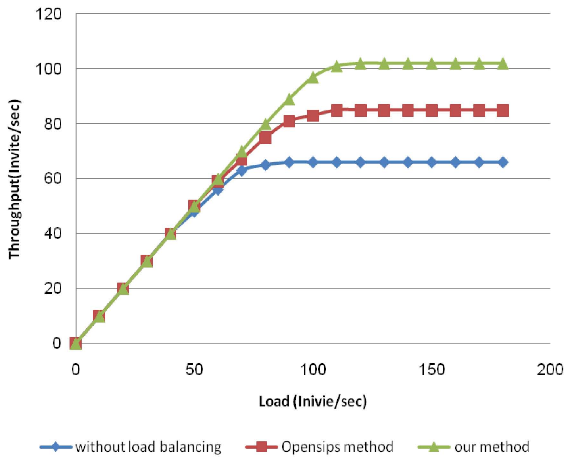

Fig. 8 Throughput of three configurations

Table 1: Computer hardware

CPU	Intel Pentium Dual 2.00 GHZ
RAM	1.00 GB
OS	Debian 5

We increased the load in SIPp UACs every 10 seconds by 10. We did our test for three different configur-ations: without load balancing (with one SIP proxy server), with Opensips load balancing method and with our method. We used "200 ok" scenario to know the throughput of network in each configuration. We configured SIPp UACs to make calls with different call durations. One of them made calls with 5 seconds as call duration and another made calls with 60 seconds as call duration. The results are shown in Fig. 8.

CONCLUSION

Voice over IP is becoming more and more popular every day. SIP as the most important protocol in voice over IP should provide scalability and reliability. One of the most important SIP network's components is SIP proxy server. To increase the amount of traffic that a SIP network can handle, one way is to do clustering for SIP proxy servers. We proposed a two stage architecture for SIP networks to increase scalability and reliability of SIP networks. We implemented our load balancing scheme in Opensips SIP server acting as dispatcher. Our scheme is based on response time of SIP proxy servers and number of connections to each SIP proxy server. We used number of connections because call durations aren't the same for all calls. We also used round trip time to take network status into account. The experimental results have shown that our method can handle more traffic load than Opensips method for load balancing.

REFERENCES

1. Wu, W.M., K. Wang, R.H. Jan and C.Y. Huang, 2007. "A Fast Failure Detection and Failover Scheme for SIP High Availability Networks," in 13th IEEE International Symposium on Pacific Rim Dependable Computing.
2. Rosenberg, J., H. Schulzrinnne, G. Camarillo, A.R. Johnston, J. Peterson, R. Sparks, M. Handley and E. Schooler, 2002. "SIP: session initiation protocol, " RFC 3261, Internet Engineering Task Force, June 2002.
3. Jiang, H., A. Iyengar, E. Nahum, W. Segmuller, A. Tantawi and C.P. Wright, 2009. "Load Balancing for SIP Server Clusters" In Proceedings of IEEE INFOCOM.
4. Hunt, G., G. Goldszmidt, R.P. King and R. Mukherjee, 1998. Network dispatcher: a connection router for scalable Internet services," Computer Networks, 30: 347-357, Apr. 1998.
5. Singh, K. And H. Schulzrinne, 2005. "Failover and load sharing in SIP telephony," In Proceedings of the 2005 International Symposium on Performance Evaluation of Computer and Telecommunication Systems SPECTS'05), July 2005.
6. Gulbrandsen, A., P. Vixie and L. Esibov, 2000. A DNS RR for specifying the location of services (DNS SRV)," RFC 2782, Internet Engineering Task Force, Feb. 2000.
7. Mealling, M. and R.W. Daniel, 2000. The naming authority pointer (NAPTR) DNS resource record," RFC 2915, Internet Engineering Task Force, Sept. 2000.
8. Leu, J.S., H.C. Hsieh, Y.C. Chen and Y.P. Chi, 2008. "Design and Implementation of a Low Cost DNS-based Load Balancing Solution for the SIP-based VoIP Service, " in IEEE Asia-Pacific Services Computing Conference, 2008.
9. "SIPp - Test Tool/Traffic Generator for the SIP Protocol " March 2006, [Online]. Available: http://sipp.sourceforge.net.