A NEW SCHEME TO REDUCE SESSION ESTABLISHMENT TIME IN SESSION INITIATION PROTOCOL (SIP)

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Technology
in
Computer Science & Engineering

BY
K.KALPANA

Department of Computer Science & Engineering

National Institute of Technology

Rourkela

May 2007

A NEW SCHEME TO REDUCE SESSION ESTABLISHMENT TIME IN SESSION INITIATION PROTOCOL (SIP)

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Technology
in
Computer Science & Engineering

BY
K.KALPANA

Under the Guidance of

Prof. A.K.TURUK

Department of Computer Science & Engineering

National Institute of Technology

Rourkela

May 2007

National Institute of Technology

Rourkela

CERTIFICATE

This is to certify that the thesis entitled,"**A New Scheme to Reduce Session Establishment Time in Session Initiation Protocol (SIP)**" submitted by Ms. **K.Kalpana** in partial fulfillment of the requirements for the award of Master of Technology in "Computer Science & Engineering "at the National Institute of Technology, Rourkela (Deemed University) is an authentic work carried out by her under my supervision and guidance.

To the best of my knowledge, the matter embodied in the thesis has not been submitted to any other University / Institute for the award of any Degree or Diploma.

Place : NIT Rourkela
Date :

Dr. A.K.TURUK
Supervisor
Asst. Professor
Dept. of Comp. Sci. & Engg.
National Institute of Techonology
Rourkela - 769008.

ACKNOWLEDGEMENT

I take this opportunity to express my deep regards and sincere gratitude for this valuable guidance rendered to me by guide Dr. A.K.TURUK, Asst. Professor, Department of Computer Science & Engineering, National Institute of Technology, Rourkela, for his kind and valuable guidance for the completion of my thesis. His consistent support and intellectual guidance made me energize and innovate new ideas.

I am thankful to Dr. S.K.Jena, Professor, H.O.D of Computer Science & Engineering, for his support and providing necessary facility to carry out the work. I am thankful to all my Professors, Lecturers and members of the department for their generous help in various ways for the completion of the thesis work.

Finally, I am also thankful to my parents, my brother and my friends who have have helped me a lot in completion of my thesis work.

K.Kalpana

Contents

Abstract

The session Initiation Protocol (SIP) has been developed by Internet Engineering Taskforce standard (IETF) with the main purpose of establishing and managing sessions between two or more parties wishing to communicate. SIP is a signaling protocol which is used for the current and future Internet Protocol (IP) telephony services, video services, and integrated web and multimedia services.

SIP is an application layer protcol, thus it can run over Transmission Control Protocol(TCP) or User Datagram Protocol (UDP). When the packets are sent over the network , a form of congestion control mechanism is necessary to prevent from network collapse. TCP is a reliable protocl and provides the congestion control by adjusting the size of the congestion windows. UDP is an unreliable protocol and no flow control mechanism is provided.

Many applications of the Internet require the establishment and management of sessions. The purpose of the thesis is to study the session establishnment procedure in SIP and try to reduce the time taken for the session setup in two different conditions.One, when there is no congestion in the network, and the other is when there is a network congestion.

We have simulated the behaviour of session establishment in SIP using Network Simulator (NS2). UDP is used as the transport protocol. We have created different network topologies. In the topology we had created SIP user agents who wants to communicte, proxy servers for forwarding the requests on behalf of the user agents, and a Domain Name Server (DNS) which maintains the location information of all proxy servers. We tried to reduce the time taken for the session establishment. As UDP does not provide any congestion control mechanisms, we used the binary exponential backoff (BEB) algorithm to set the timers. In our network topolgy when there is no packet loss in the network, the time taken for the session establishment is reduced from 0.86 sec to 0.574 sec. In case of network congestion the setup time is reduced from 4.55 sec to 2.86 sec.

From the simulation, we conclude that the session establishment time can be reduced by reducing the number of message exchanges required for session setup.

List of Figures

List of Tables

CHAPTER 1

INTRODUCTION

Session Initiation Protocol (SIP) is an application layer control (signaling) protocol for creating, modifying and terminating sessions with one or more participants [3]. SIP has been developed by IETF standard for IP telephony. The session is considered as an exchange of data between an association of participants. SIP enables the creation of infrastructure of network hosts (proxy servers) for locating session participants. SIP makes use of Session Description Protocol (SDP) which is used for carrying the session descriptions. SIP works independently of the underlying transport protocols.

1.1 Motivation

Many applications of the Internet require the establishment and management of sessions. The next generation of enterprise networks is undergoing major changes in new architectures, applications, and services begin to rollout within businesses. In general, the world of voice/telephony, video, and data are “converging ”into a global communications network. Most of the communication is in digital form and data is transported via packet networks such as IP (Internet Protocol), ATM (Asynchronous Transfer Mode), and Frame Relay. Since data traffic is growing much faster than telephone traffic, there has been considerable interest in transporting voice over data networks. Organizations have been working on the solutions, which would allow them to use the excess capacity on broadband networks for voice and data transmission, as well as to utilize the Internet and company Intranets as alternatives to more expensive systems.

A large number of factors are involved in creating a robust network capable of delivering multimedia services. Some of these factors include better voice and video codecs, packetization, packet loss, packet delay, jitter variation, directory services, resource integration, and reliable network architecture. Also, critical are the choices of call signaling protocols, security concerns, the ability to integrate seamlessly with existing Internet Services and the need to traverse network address translator (NAT) and firewalls.

The main motivation for transporting voice over data networks is the potential cost saving achievable by eliminating or bypassing the circuit switched telephony infrastructure. Further, the universal presence of IP and associated protocols in user and network equipment is the key reason for the Voice over IP.

The beauty of VoIP lies in its efficient use of bandwidth. Unlike circuit-switched PSTN network, where in VoIP employs packet switching that performs statistical multiplexing. In circuit switching an end-to-end path must be set up before any data can be sent i.e. it statically

reserves the required bandwidth in advance. Where as in packet switching, bandwidth is dynamically allocated to various links based on their transmission activity. With circuit switching, any unused bandwidth on an allocated circuit is just wasted. With packet switching it may be utilized by other packets from unrelated sources going to unrelated destinations. Along with bandwidth saving techniques such as voice compression, silence suppression VoIP offers more capacity that is normally impossible for PSTN.

The evolution of the core enterprise network to IP is enabling the migration of the traditional circuit-switched voice and call signaling message traffic over the Internet using voice-over-IP (VOIP) technology, and SIP is used as the call signaling protocol. We first analyze the benefits offered by such a unified end-to-end IP-based multimedia network solution [2].

1. Cost reduction:Moving voice calls over the Internet eliminates the notion of long-distance. Further convergence of voice, data, and video traffic can improve network efficiency and reduce operation cost.

2. Utilization: Digitized voice calls require less bandwidth than the traditional 64 kbps circuit calls and, hence, more calls can be made over the existing bandwidth.

3. Simpler integration: An integrated infrastructure allows more standardization and is simpler to manage. It is now possible to have tighter integration with web-based applications and supply chains.

4. Enhanced services: Richer and enhanced services that integrates existing enterprise applications with VOIP, video, or presence technologies is now possible.

5. Consolidation: In a network the most significant cost elements are users. So it would be beneficial if we a get any opportunity to combine operations, to eliminate points of failure, and to consolidate accounting systems to track usage of resources.

While enterprise customers clearly see the benefit of migrating to such converged networks, even the service providers have optimism to support such convergence.

- More revenue: While tradition voice business is down, data is growing. Hence, digitized voice and video services will provide them with more revenue models.

- Efficiency: It has been proven that it is more efficient and cheaper to provision a packet-switched network than a circuit-switched network [16]. Hence, the migration toward packet architecture is inevitable.

- Ubiquitous service: The service providers will now be in a position to offer any service (voice, video, or data) to any customer through their converged network.

SIP, which is an IETF standard for IP telephony, has received much attention recently and seems to be the most promising candidate as a signaling protocol for current and future IP telephony services, video services, and integrated web and multimedia services. In particular, IP subsystem of 3G networks, known as IP Multimedia System (IMS), is conceived according to the functionalities of the SIP; subsequently, the 3GPP consortium adopted SIP as the signaaling framework for Universal Mobile Telecommunication Networks (UMTS) [4]. SIP is an ASCII-based application layer signaling protocol that can used to establish, maintain and terminate calls between two or more end points. While SIP is new and actual deployment experiences are fewer, it is widely expected that future enterprise networks will incorporate SIP for its simplicity, flexibility, and built in security features.

1.2 Objective

To make Internet multimedia (audio or video) calls, a caller must know the IP address and port number, where the callee wants to receive the audio/video packets, as well as the audio and video codecs the callee supports. SIP uses high level addresses of the form userID@domain, and this facilitates users mobility. SIP makes use of proxy servers to help in routing requests to the user's current location and to which users can send registrations, invitations to sessions and other requests. The goal of the thesis is to discuss the issues in the establishment of sessions in SIP and present different solutions for reducing the session setup time.

1.3 Contribution

we have simulated the behaviour of SIP using network simulator (NS2) [10, 11] where UDP is used as the transport protocol. SIP user agents, proxy servers and domain name server were created in the network. SIP user high level addresses of the form user@domain. SIP user agent(UA) who wants to communicate with another person whose IP address is not known, will send an INVITE request to the proxy in its domain. If the destination is in its domain then proxy will send it to the destination. Otherwise it will request the DNS for the location of the UA. After getting the address from DNS, the proxy will send the request to the proxy in the destination domain, which in turn forwards the request to the destination.

In session establishment, the response from the destination will follow the reverse path. But once the destination UA recieves the request, it will get the IP address of the source UA, so it can directly send the response to the source UA. The INVITE packet may lost because of congestion or the packet may be delayed. As UDP provides no flow control mechanism, SIP uses retransmission timers for congestion problems. We used the timers at UAs so that when the response is not received in time, the request will be retransmitted. We had also maintained timers at proxy servers, and they will send the requests on behalf of UAs if the respose is not received within in that timeout period. In the first case, we tried to reduce the number of message exchanges and in later case, the number of message exchanges and the distance traversed for sending the request message are reduced, so that the session setup time is reduced.

1.4 Organization of the Thesis

In chapter 2, we presented a brief overview of SIP, the structure of the protocol, the request/response model of SIP. The working of SIP, and behaviour of the logical entities involved in a SIP session is presented in chapter 3. The simulation of SIP using network simulator (NS2) is presented in chapter 4, in which the REGISTER request is presented in chapter 4.1. Sending the INVITE request and reducing the session establshment delay by different cases is discussed in 4.2, the BYE request is presented in 4.3. The final results are presented in chapter 4.4. Finally, in chapter 5 we draw up the conclusions and the future work.

CHAPTER 2

SESSION INITIATION PROTOCOL

2.1 Protocol Definition

The SIP protocol is an application layer control signaling protocol created with the purpose of establishing, modifying, and terminating voice, video, and multimedia sessions. These sessions include Internet telephone calls, multimedia distribution, and multimedia conferences. Session is considered as an exchange of data between an association of participants. SIP works on top of several different transport protocols.

Many applications of the Internet require the management and creation of sessions. The implementation of these applications is complicated by the following reasons:

- Users may move between end points.
- The users may be addressable by multiple names.
- The users may communicate in serveral different media sometimes simultaneously.

SIP provides a registration function that allows users to upload their current location for use by proxy servers. A user is not limited to registering from a single device. SIP invitations are used to create sessions, carry session descriptions that allow participants to agree on a set of compatible media types. SIP makes use of proxy servers to help in routing requests to the user's current location and to which users can send registrations, invitations to sessions and other requests.

2.2 Overview

Many applications such as Unified Messaging (UM) systems and Interactive Voice Recognition (IVR) systems have been developed out of traditional telephony. They can be used for storing and interacting with voice, video, faxes, email, and instant messaging services. Users often use SIP to initiate communications with these applications [15].

2.2.1 Functions of SIP

1. SIP can establish, modify and terminate multimedia sessions (conferences) such as Internet telephony calls.
2. Invite participants to an already existing session, such as multicast conferences.

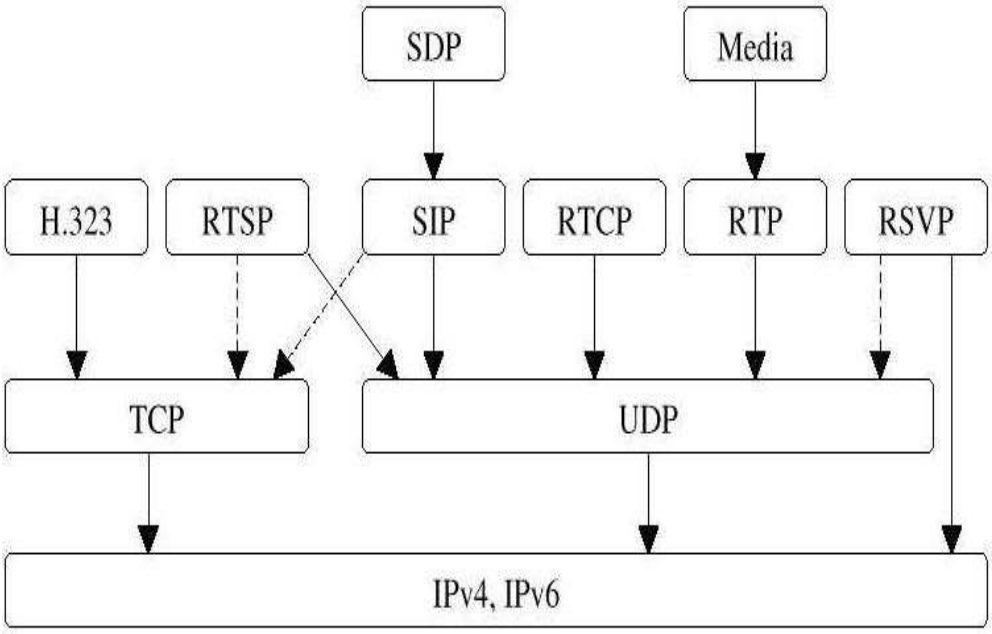


Fig. 2.1: IP Telephony Protocol Architecture

3. Media can be added to (and removed from) an existing session.

4. SIP supports name mapping and redirection services which supports personal mobility. i.e. users can maintain a single externally visible identifier regardless of their network location.

SIP works independently of the underlying transport protocols and without dependency on the type of session that is being established. SIP is rather a component that can be used with other IETF protocols to build complete multimedia architecture. The Internet Protocol (IP) telpehony protocol stack is shown in fig.2.1. SIP is used with the following protocols to build a complete multimedia architecture and provide completee services to the users [2]:

- Real-time Transport Protocol (RTP) and Real-time Transport Control Protocol (RTCP) for transporting real-time data and providing QOS feedback.
- Real-time Streaming Protocol (RTSP) for controlling delivery of streaming media.
- Media Gateway Control Protocol (MEGACO) for controlling gateways to the public switched telephone network (PSTN).
- Session Description Protocol (SDP) [5] for delivering multimedia services.

Therefore, SIP should be used in conjunction with other protocols in order to provide complete services to the users. SIP (RFC 3261 [3]) works with both IPv4 and IPv6. SIP provides

a set of security services, which include denial of service prevention, authentication, integrity protection, encryption and privacy services.

2.2.2 SIP Overview

The Session Initiation Protocol (SIP) is an application-layer control protocol that can establish, modify and terminate multimedia sessions or calls [3]. These multimedia sessions include multimedia conferences, distance learning, Internet telephony, Voice mail, Instant Messaging (IM), and similar applications. SIP can invite both persons and ”robots”, such as a media storage service. SIP can invite parties to both unicast and multicast sessions; the initiator does not necessarily have to be a member of the session to which it is inviting. Media and participants can be added to an existing session. SIP can be used to initiate sessions as well as invite members to sessions that have been advertised and established by other means. Sessions can be advertised using multicast protocols such as SAP, electronic mail, news groups, web pages or directories among others.

To make Internet multimedia (audio or video) calls, a caller must know the IP address and port number, where the callee wants to receive the audio/video packets, as well as the audio and video codecs the callee supports [2]. IP addresses are hard to remember and can easily change with mobility when they receive dynamic addresses through Dynamic Host Control Protocol (DHCP) [**?**] servers [13]. Users in a SIP network are identified by unique SIP addresses. SIP uses high level addresses of the form userID@domain, which is called SIP Uniform Resource Identifier (URI) and this facilitates users mobility. The user ID can be either a username or an E.164 address.

SIP transparently supports name mapping and redirection services, allowing the implementation of ISDN and Intelligent Network telephony subscriber services [9]. These facilities also enable personal mobility. In the parlance of telecommunications intelligent network services, this is defined as: ”Personal mobility is the ability of end users to originate and receive calls and access subscribed telecommunication services on any terminal in any location, and the ability of the network to identify end users as they move. Personal mobility is based on the use of a unique personal identity (i.e., personal number).” [3]. Personal mobility complements terminal mobility, i.e., the ability to maintain communications when moving a single end system from one subnet to another.

2.2.3 Capabilities of SIP

Like other VOIP protocols, SIP is designed to address the functions of signaling and session management within a packet switching network. Signaling allows call information to be carried across network boundaries. Session management provides the ability to control the attributes of an end-to-end call. SIP supports four facets of establishing and terminating multimedia communications:

- **User location:** determination of the end system to be used for communication.
- **User capabilities:** determination of the media and media parameters to be used.
- **User availability:** determination of the willingness of the called party to engage in communications.
- **Call setup:** "ringing ", establishment of call parameters at both called and calling party.
- **Session Management:** includes transfer and termination of sessions, modifying session parameters, and invoking services.

SIP Elements

There are basically four different entities involved in a SIP session :user agents, registrars, proxy servers, and redirect servers [1]. The differentiation among proxy servers is logical, not physical; tht ts, the same SIP server can act as registrar and proxy at the same time, for example. User agnets can be client or server user agents, and agian the distinction is logical. A user agent represents an end system. It contains a user agent client (UAC) , and a user agent server (UAS). A client user agent is the entity initiating a session, by sending a request, and a server user agent is the entity incharge to process such a request and send accordingly the responses required. Thus, the difference is in the "direction "in which a user agent is used. Registrars are entities where SIP users are registered and that make the information necessary to localize a SIP user available to other SIP entities, like proxy or redirect servers, that is, they provide the location service. Proxy servers are intermediary entities that accept and forward SIP messages on behalf of user agents. Redirect servers perform similar tasks, but they provide the information necessary to locate another SIP entity where a user can be contacted. In other words, they redirect the user to an alternative set of SIP addresses (also known as Uniform Resource Identifiers, URI) where the callee can be contacted.

An important feature of SIP, missing in other signaling protocols, is the possibility of forwarding single request calls to multiple destination addresses during session establishment [1]. This function, called forking, is important because it permits to provide serveral advanced telephony services, such as automatic call forwarding to Voice Mail or simpler user location. The SIP protocol allows both parallel forking, where calls are forwarded to multiple destination addresses at the same time, or sequential forking, where each address is tried sequentially, if the user was not contacted at the previous address.

2.3 Structure of the Protocol

SIP is structured as a layered protocol, which means that its behavior is described in terms of a set of fairly independent processing stages with only a loose coupling between each stage.

1. The lowest layer of SIP is its syntax and encoding. Its encoding is specified using an augmented Backus-Naur Form grammar (BNF).

2. The second layer is the transport layer. It defines how a client sends requests and receives responses and how a server receives requests and sends responses over the network. All SIP elements contain a transport layer.

3. The third layer is the transaction layer. Transactions are a fundamental component of SIP. A transaction is a request sent by a client transaction (using the transport layer) to a server transaction, along with all responses to that request sent from the server transaction back to the client. The transaction layer handles

   - Application layer retransmissions.
   - Matching of responses to requests.
   - Application layer timeouts.

   Any task a UAC accomplishes takes palce using a series of transactions. Stateless proxies donot contain a transaction layer. The transaction layer has client component and a server component, each of which are represented by a finite state machine that is constructed to process a particular request.

4. The layer above the transaction layer is called the transaction user (TU).

Each of the SIP entities, except the stateless proxy, is a transaction user. When a TU wishes to send a request, it creates a client transaction instance and passes the request along with the destination IP address, port, and transport to which to send the request.

SIP is a text-based protocol and uses the UTF-8 char set. A SIP message is either a request from a client to a server, or a response from a server to a client.

Two main things are present in this protocol implementation are Session and transaction.

1. **Session:**

   From the SDP specification "A multimedia session is a set of multimedia senders and receivers and the data streams flowing from senders to receivers. A multimedia conference is an example of a multimedia session." (RFC 2327) (A session as defined for SDP can comprise one or more RTP sessions.) A callee can be invited several times, by different calls, to the same session. If SDP is used, a session is defined by the concatenation of the SDP user name, session id, network type, address type, and address elements in the origin field.

2. **SIP Transaction:**

   A SIP transaction occurs between a client and a server and comprises all messages from the first request sent from the client to the server up to a final (non-1xx) response sent from the server to the client. If the request is INVITE and the final response is a non-2xx, the transaction also includes an ACK to the response. The ACK for a 2xx response to an INVITE request is a separate transaction.

2.4 SIP Messages

SIP is a text-based protocol. This makes a SIP header largely self-describing and minimizes the cost of entry. Since most values are textual, the space penalty is limited to the parameter names. SIP is based on request/response transaction model. Each transaction consists of a request that invokes a particular method or function on the server and at least one response. These, requests and responses, include different headers to describe the details of the communication.

2.4.1 Requests

The request is characterized by the Start-Line, called Request-Line. It starts with a method token followed by a Request-URI and the protocol version. Requests are referred to as methods. There are six different request methods in SIP whose funcationality is discribed as follows:

- **INVITE:** The INVITE method indicates that the user or service is being invited to participate in a session. For a two-party call, the caller indicates the type of media it is able to receive as well as their parameters such as network destination. A success response indicates in its message body the port number where the callee wants to receive audio/video packets, as well as the audio and video codecs the callee supports.
- **ACK:** The ACK request confirms that the client has received a final response to an INVITE. It may contain a message body with the final session description to be used by the callee. If the message body is empty, the callee uses the session description in the INVITE request. This method is only used with the INVITE request.
- **BYE:** The user agent client uses BYE to indicate to the server that it wishes to release the call.
- **CANCEL:** The CANCEL request cancels a pending request, but does not affect a completed request. (A request is considered completed if the server has returned a final response).
- **OPTIONS:** The OPTIONS method allows a user agent (UA) to query another UA or a proxy server as to its capabilities. This allows a client to discover information about the supported methods, content types, extensions, codecs, etc... without ringing the other party. If no response is returned for the OPTIONS, then timeout error must be returned. that is, the target is unreachable and hence unavailable.
- **REGISTER:** The REGISTER method stores the contact information of UA's and conveys information about a user's location to a SIP server.

2.4.2 Responses

After receiving and interpreting a request message, the recipient responds with a SIP response message, indicating the status of the server, success or failure. The responses can be of different kinds and the type of response is identified by a status code, a 3-digit integer. The first digit defines the class of the response. The other two have no categorization role. The six different classes that are allowed in SIP are here listed with their meaning in Table 2.1. These classes can

Table 2.1: SIP Response status codes and description

Response Class	**R**esponse Code	**D**escription
1xx	Provisional	Request received, continuing to process the request.
2xx	Success	The action was successfully received, understood and accepted.
3xx	Redirection	Further action needs to be taken in order to complete the request.
4xx	Request Failure	The request contains bad syntax or cannot be fulfilled at this server.
5xx	Server Failure	The server failed to fulfill an apparently valid request.
6xx	Global Failure	The request cannot be fulfilled at any server.

be categorized by provisional and final responses. A provisional response is used by the server to indicate progress, but does not terminate a SIP transaction. 1xx responses are provisional, other responses are considered final.

A response is said to be final which terminates a SIP transaction, as opposed to a provisional response that does not. All 2xx, 3xx, 4xx, 5xx and 6xx responses are final.

SIP applications are not required to understand the meaning of all registered response codes, though it is desirable. However applications must be able to recognize the class of the response and treat any unrecognized response as being the x00 response code of the class. Some of the responses of different class are shown in Fig 2.2.

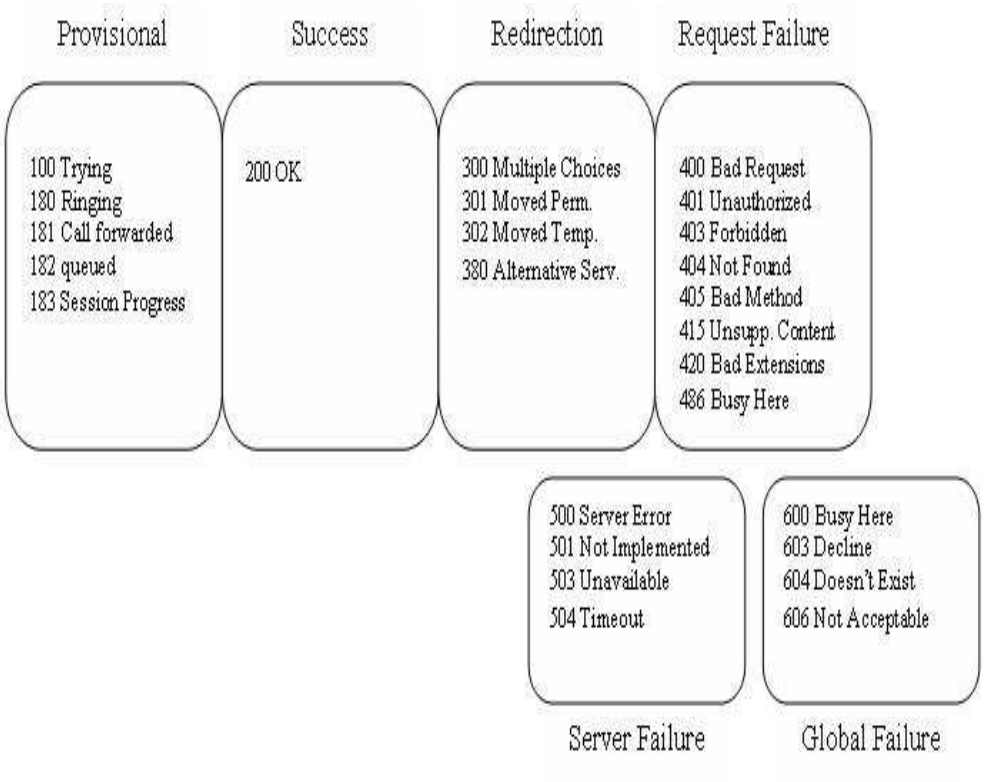


Fig. 2.2: SIP Responses of differene Classes

CHAPTER 3

WORKING OF SIP

The request handling in SIP is often classified as INVITE or Non-INVITE. SIP can run over TCP or UDP. UDP is preferred for real time data transfer as it avoids the phase of TCP connection set-up and teardown. A form of congestion control is necessary to prevent network collapse, as UDP provides no flow control mechanism. SIP itself doesnot define a congestion control mechanism. The only effort SIP itself poses in facing congestion problems is using exponential retransmission back-off timers.

To be able to locate and invite participants there has to be a way the called party could be addressed. The entities addressed by SIP are users at hosts, identified by a SIP URL. The SIP URL has an email-like identifier of the form user@host. Where the user part can be a user name, a telephone number. The host part is either a domain name or a numeric network address. In many cases a user's SIP URL could be guessed from the users email address.

Examples of SIP URLs could be:

sip:xxx@example.com

sip:abc@176.7.6.1

This URL may well be placed in a web page, so that clicking on the link, as in the case mail URLs, initiates a call to that address.

3.1 UAC Behaviour

The SIP message format is given as follows:

generic message = start-line

*message header

CRLF

Start-line = Request-line/status-line

Request-line = Method sp Request-URI sp SIP-version CRLF

Status-line = SIP-version sp status-code sp Reason-phrase CRLF

sp - Single space.

CRLF - Carriage Return Line Feed sequence.

Every SIP request/response must contain the Request-line/Status-line.
A valid SIP request generated by a UAC must contain at minimum, the following header fields:

- **To:** The To header field contains the logical recipient if the request, or the address of the user or resource that is the target of the request.
- **From:** The From header feild specifies the logical identity or address of the requestor. The From header field is used to determine which processing rules to apply to a request. Ex : Automatic call rejection
- **Call-ID:** The Call-ID field is a unique identifier to group together a series of messages. It must be same for all requests and responses sent by either user agent in a dialog. It should be same in each registration from a UA.
- **CSeq:** The CSeq field is used for identifying and to order the transactions. It consists of sequence number and a method. The method name must match that of the request. Ex : CSeq : 4711 INVITE
- **Max-Forwards:** The Max-Forwards field serves to limit the number of hops a request can transit on the way to its destination. It consists of an integer value that is decremented by one at each hop. If the Max-Forwards value reaches zero before the request reaches its destination, it will be rejected with a 483 (Too Many hops) error response.
- **Via:** The Via header feild indicates the transport used for the transaction and identifies the location where the response is to be sent.
- **Contact:** The Contact header feild provides a SIP URI that can be used to contact that specific instance of the UA for subsequent requests.

The request is sent to the destination either by applying Domain Name Server (DNS) procedures to identify the destination, or if there is a local policy , it may specify an alternative set of destinations to attempt.

3.2 UAS Behaviour

The request processing is atomic.

- **Method Inspection:** After the UAS receives the request it may be authenticated or authentication may be skipped. Then, the UASmust inspect the method of the request. If the UAS recognizes but doesnot support the method of the request, it must generate a 405 (Method Not Allowed) response. The UAS must also ass an Allow header field to the 405 response which lists the set of methods supported by UAS. If the method is supported by server , processing continues.

- **Header Inspection:** The UAS must inspect To, Request-URI, Require the header fields. The Require header field is used by a UAC to tell a UAS about SIP extensions that the UAC expects the UAS to support in order to process the request.

- **Content Processing:** If UASunderstands the extensions then it examines the body of the message, and then the header fields that describe it. If UAS does not understand the content-type, content language, content-encoding and the body part is not optional (Content-Disposition header field), the UAS must reject the request with a 415 (Unsupported media type) response. The response must contain the Accept, Accept-encoding, Accept-language header fields to indicate UAC about the list of media supported by UAS.

- **processing the request:** UAS should send a final response to a non-INVITErequest. When 100 Trying response is generated, any Timestamp header field present in the request must be copied into this 100 (Trying) response. If there is a delay in generating the response, this delay value is added to the time-stamp value in the response.

  delay = Sending response - request received . (seconds)

3.3 Registrar

Each user wishing to communicate must register with a registrar server using their assigned SIP addresses. The registrar server writes this information, known as ”binding”, into a database, called location service. The location service contains the information that allows a proxy to input a URI and receive a set of zero or more URIs that tell the proxy where to send the request. A user is not limited to registering from a single device. More than one user can be registered on a single device at the same time.

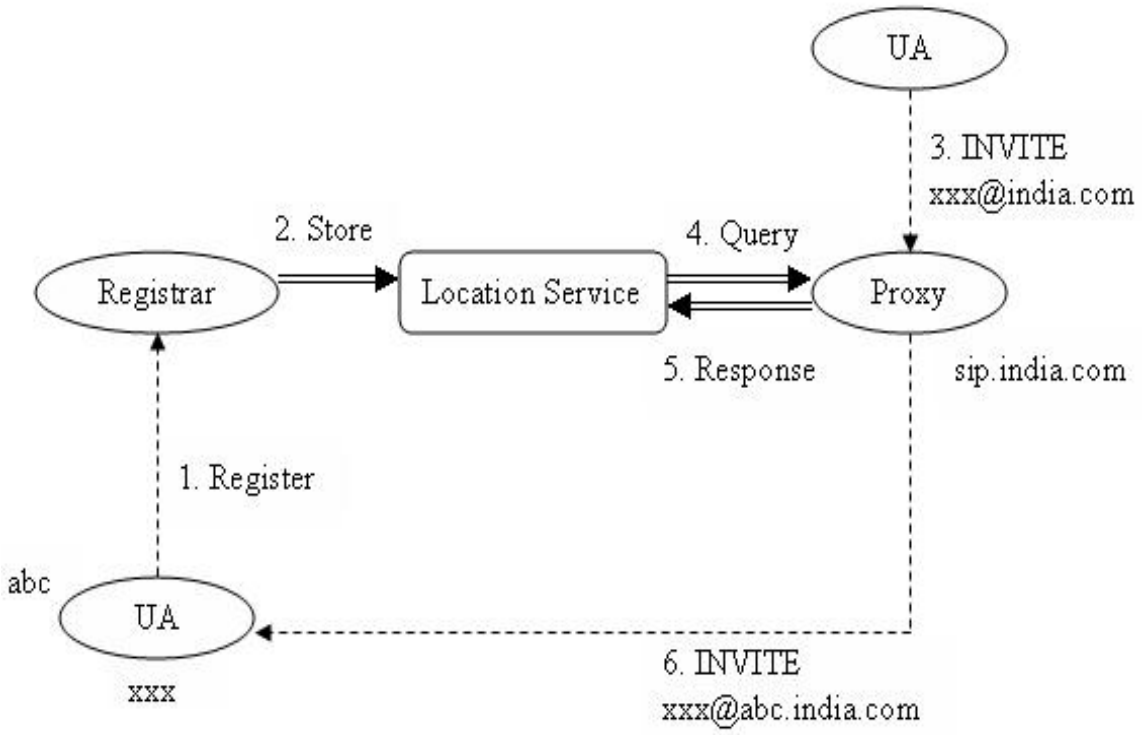


Fig. 3.1: REGISTER example

3.3.1 Constructing the REGISTER request

REGISTERrequests add, remove, and query for bindings. The following header fields will be included in REGISTER request.

- **Request-URI:** The Request-URI names the domain of the location service for which the registration is meant.
  ex : sip: chicago.com
- **To:** The To field contains the address of whose registration is to be created, queried , or modified.
- **From:** The From field values is same as To.
- **Call-ID:** The Call-ID is a unique identifier. All registrations from a UAC should use the same call-ID header field value for registrations sent to particular registrar.
- **CSeq:** The CSeq value generates proper ordering of REGISTER requests. A UA must increment this value by one for each REGISTER request with the same call-ID.

- **Contact:** The Contact field contains zero or more address bindings. The 2xx response to the REGISTER request will contain , in a contact header field, a complete list of bindings that have been registered for thisaddress at this registrar.

- **Expires:** The value in the Expires field indicates how long the UA would like the binding to be valid.

The procedure of sending REGISTER request is shown in Fig. 3.1. SIP UA will send a REGISTER request to the registrar in its domain. The registrar stores this information in the location service. The registrar may be a proxy server or redirect server. In the Fig. 3.1 we can observe that when the UA sends an INVITE message (3) to the proxy server, the proxy will contact the location service to get the address of destination. When the proxy gets the response (5) from the location service, will forward the request to the destined UA.

UAs must not send a new registration until they have received a final response for the previous one from the registrar or the previous REGISTER request has timed out.

3.4 Proxy Server

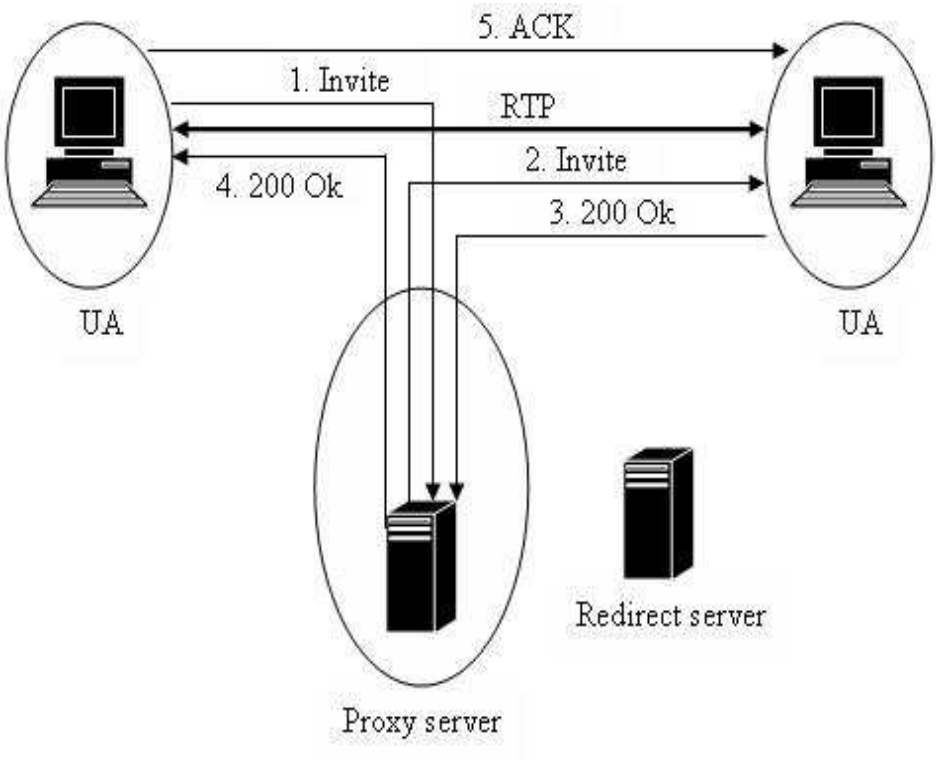


Fig. 3.2: SIP session through a Proxy Server

SIP makes use of proxy servers for locating session participants and to which users can send their register requests, invitations to sessions and other requests. Proxy servers make use of DNS and location service lookups for locating the end user. Proxy servers can also make flexible ”routing decisions” to decide where to send a request.

If a proxy server is used, the caller UA sends an INVITE request to the proxy server. If the callee UA is present in the same domain then proxy server determines the path, and then forwards the request to the callee as shown in Fig. 4.2. The callee responds to the proxy server by sending 200 Ok response. The proxy server in turn forwards the 200 Ok response to the caller. The caller UA sends an ACK to the callee to indicate the receipt of the response. After the callee receives the ACK the session is established between the caller and callee. Real-time Transport Protocol (RTP) is used for the communication between the caller and callee.

3.5 Redirect Server

In some architectures it may be desirable to reduce the Processing load on proxy servers that are responsible for routing requests, and improve signaling path robustness, by relying on redirection.

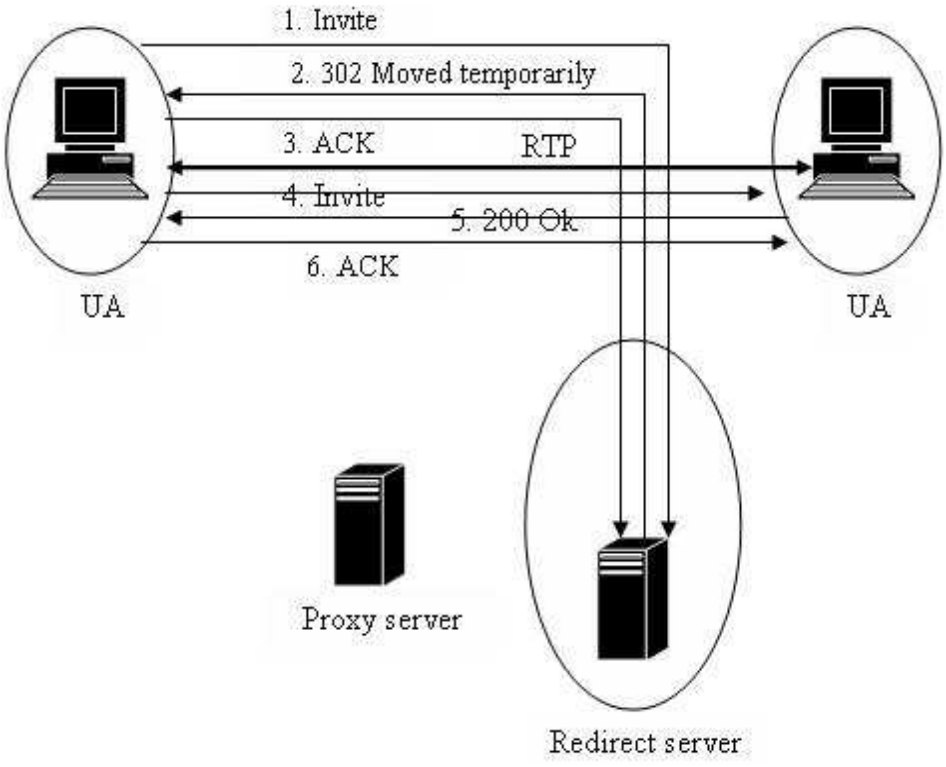


Fig. 3.3: SIP session through a Redirect Server

Redirection allows servers to push routing information for a request back in a response to the clients, thereby taking themselves out of the loop of further messaging for this transaction while still aiding in locating the target of the request.

When the originator of the request receives the redirection, it will send a new request based on the URI(s) it has received. By propagating URIs from the core of the network to its edges, redirection allows for considerable network scalability. For CANCEL requests the redirect server should return a 2xx response.

The session establishment through redirect server is shown in Fig.4.3. If a redirect server is used, the caller UA sends an INVITE request to the redirect server, the redirect server contacts the location server to determine the path to the callee, an then the redirect server sends that information back to the caller. The caller then acknowledges receipt of the information.

The caller then sends a request to the device indicated in the redirection information (which could be the callee or another server that will forward the request). Once the INVITE request reaches the callee, it sends back a response and the caller acknowledges the response. After the callee receives the ACK, the session is established. RTP is used for the communication between the caller and the callee.

3.6 Initiating a session

3.6.1 UAC Processing

Creating the Initial INVITE:

- An allow header field should be present in the INVITE request. It indicates what methods can be invoked within the dialog, on the UA sending the INVITE, for the duration of the dialog.

- A supported header field which enumerates all the extensions understood by UAC, should be present in the INVITE.

- An Accept header field may be present in the INVITE, which indicates the contents acceptable by the UA. It is useful for indicating support of various session description formats.

- The UAC may add an Expires header field to limit the validity of the invitation. If there is no final response for this request, then the UA should generate a CANCEL request for the INVITE.

he UAC may add a message body to the INVITE. SIP uses an offer/answer model [8] where one UA sends a session description, called offer, which contains the description of the session. i.e. it indicates the desired communication means (audio, video, games), parameters of those means (such as codec types) and addresses of receiving media from the answerer.

The other UA responds with an answer, which indicates which communication means are accepted, the parameters that apply to those means, and addresses for receiving media from the offerer.

If a 2xx response contains an offer, the ACK must carry on answer in its body. If the offer in the 2xx response is not acceptable, the UAC core must generate a valid answer in the ACK and then send a BYE immediately. The UAC core considers the INVITE transaction completed in 64 T1 seconds after the reception of the first 2xx response. After acknowledging any 2xx response to an INVITE, the UAC core can terminate the dialog by sending a BYE request. A dialog is a peer-to-peer SIP relationship between two UAs that persists for some time. A dialog is established by SIP messages, such as a 2xx response to an INVITE request. Core designates the functions specific to a particular type of SIP entity, i.e., specific to either a stateful or stateless proxy, a user agent or registrar. All cores, except those for the stateless proxy, are transaction users.

3.6.2 UAS processing

Processing of the INVITE

The UAS performs the following:
If the request is an INVITE that contains an Expires header field, the UAS sets a timer for the number of seconds indicated in the header field value. If the invitation expires before the UAS has generated a final response, a 487 (Request terminated) response should be generated.

The INVITE may contain a session description. It is possible that the user is invited and the user is already a participant in that session, even though the INVITE is ouside of a dialog. The UAS can detect this duplication within the session description (SDP) using the session id, version number in the origin field.

If the INVITE does not contain a session description, then the UAC is asking the UAS to provide the offer of the session. The UAS can indicate progress, accept, redirect, or reject the invitation.

Progress

If the UAS desires an extended period of time to answer the INVITE, it will need to ask for an ”extension” in order to prevent proxies from canceling the transaction.

A proxy has the option of canceling a transaction when there is a gap of 3 minutes between responses in a transaction. To prevent cancellation, the UAS must send a non-100 provisional response at every minute, to handle the possibility of lost provisional responses.

The INVITE is Redirected

If the UAS decides to redirect a call, a 3xx response is sent. A 300 (Multiple choices), 301 (Moved permanently), or 302 (Moved temporarily) response should contain a contact header field containing one or more URIs of new addresses to be tried.

The INVITE is Rejected

When a UAS is not willing or able to take additional call it will send a 486 (Busy Here) response. When no end system will be able to accept this call, a 600 (Busy Everywhere) repose should be sent.

A UAS rejecting an offer containe in an INVITE should return a 488 (Not Acceptable Here) response. Such a response should include a warning header field value explaining why the offer was rejected.

The INVITE is accpeted

The UAS generates a 2xx response, which establishes a dialog.

3.7 Modifying an Existing Session

A successful INVITE request establishes both a dialog between two UAs and a session using the offer-answer model. Modifying the session involves changing the addresses or ports, adding a media stream, deleting a media stream, and so on.

Either the caller or callee can modify an existing session by sending a re-INVITE request. If a UA receives a non-2xx response to a re-INVITE, the session parameters must be unchanged. If the non-2xx final response is a 481 (Call/Transaction does not Exist), or a 408 (Request Timeout), or no response at all is received, then the UAC will terminate the dialog.

If a UAC receives a 491 response to a re-INVITE, it should start a timer with a value T choosen as follows :

1. If the UAC is the owner of the call-ID of dialog-ID, T has a randomly chosen value between 2.1 and 4 sec in units of 10 ms.

2. If the UAC is not the owner of the call-ID of the dialog-ID, T has a randomly chosen value of between 0 and 2 sec in units of 10 ms.

When the timer fires, the UAC should attempt the re-INVITE once more, if it still desires for that session modification to take place.

UAS Behaviour

A UAS that receives a second INVITE before it sends the final response to a first INVITE with a lower CSeq sequence number on the same dialog must return a 500 (Server Internal Error) response to the second INVITE and must include a Retry-After header field with a randomly choosen value of between 0 and 10 sec.

A UAS that receives an INVITE on a dialog while an INVITE it had sent on that dialog is in progress must send a 491 (Request Pending) response to the received INVITE. If a UA receives a re-INVITE for an existing dialog, it must check any version identifiers in the session description, the content of the session description. If there is a change then the UAS must adjust the session parameters accordingly, but after asking the user for confirmation.

If the new session desription is not acceptable, the UAS can reject it by returning a 488 (Not Acceptable Here) response for the re-INVITE, it should include a warning header field. If a UAS generates a 2xx response and never receives an ACK, it should terminate the dialog by BYE.

3.8 Terminating a Session

The UA can terminate the session with a BYE request.

3.8.1 UAC Behaviour

If the response for the BYE is a 481 (Call/Transaction does not Exist) or a 408 (Request Time-out) or no response at all is received for the BYE, the UAC must consider the session and will terminate the dialog.

3.8.2 UAS Behaviour

If the BYErequest does not match with existing dialog, the UAS should send a 481 (Call/Transaction does not Exist) response. For any pending requests a 487 (Request terminated) response should be generated.

3.9 Canceling a Request

A UAS that receives a CANCEL request for an INVITE, but has not yet sent a final response should stop ringing, and then respond to the INVITE with a specific error response (a 487 response).

3.9.1 UAC Behaviour

The Request-URI, call-ID, To, the numeric part of CSeq, and From header fields in the CANCEL request must be identical to those in the request being cancelled, including tags.

3.9.2 UAS Behaviour

The CANCEL method requests that the TU at the server side cancel a pending transaction. If the UAS did not find a matching transaction for the CANCEL, it should respond with a 481 response.

If the UAS had issued a final response to the original request, then the cancel request has no effect on the processing of original request, no effect on session state, no effect on the responses generated for the original request.

If the UAS had not issued a final response

1. if the original request was an INVITE, the UAS should respond with a 487 (Request terminated).

2. Otherwise CANCEL request has no impact on processing of transactions.

3.10 Querying for Capabilities

The SIP method OPTIONS allows a UA to query another UA or a proxy server as to its capabilities. This allows a client to discover information about the supported methods, content types, extensions, codecs, etc... without ringing the other party.

The target of the OPTIONS request is identified by the Request-URI, which could identify another UA or a SIP server. If no response is returned for the OPTIONS, then timeout error must be returned. i.e. the target is unreliable and hence unavailable. An OPTIONS request may be sent as part of an established dialog to query the peer on its capabilities that may be utilized later in the dialog.

An Accept header field should be included to indicate the type of message body the UAC wishes to receive in the response. This is set to the media capabilities of a UA, such as SDP. The response code will be same as the INVITE. Allow, Accept, Accept-Encoding, Accept-languages and supported header fields should be present in a 200 ok response to an OPTIONS request. Contact header fields containing alternative names and methods of reaching the user may be present in 200 ok response.

CHAPTER 4

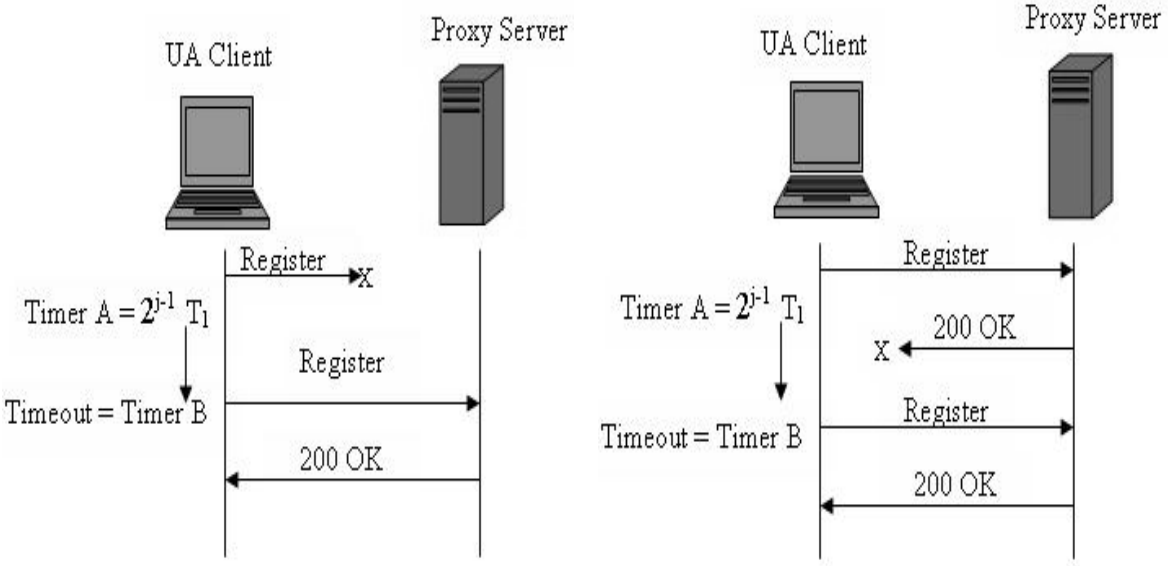
CONTRIBUTION

We had simulated the Session Initiation Protocol using Network Simulator (NS2). SIP can run over TCP or UDP. UDP is preferred for real time data transfer as it avoids the phase of TCP connection set-up and teardown. A form of congestion control is necessary to prevent network collapse, as UDP provides no flow control mechanism. SIP itself doesnot define a congestion control mechanism. The only effort SIP itself poses in facing congestion problems is using exponential retransmission back-off timers.

So in the simulation SIP protocol was developed under UDP transport. We had created two different network topologoes which consits of two sip user agents, two proxy servers, and one DNS server which will act as the location service. Each user agent will be given a SIP address and the domain of the UA. The proxy servers were also assigend a unique address. We assumed the UAs as Alice and Bob. The proxy Servers are taken as www.nitrkl.ac.in, and www.jntu.ac.in. The proxy servers also act as registrar servers.

4.1 Registration

As each sip user wishing to communicate must register its sip address with the registrar, Alice and Bob will be registered in domains www.nitrkl.ac.in, and www.jntu.ac.in respectively. Here, the proxy servers themselves act as the registrar.



(a) When the REGISTER message is lost (b) When the Response is lost

Fig. 4.1: Registration in SIP using Timers

When the UA sends a REGISTER request to the proxy server, a timer set. If the UA does not get the response within that timeout period, the timer is fired and the REGISTER request is sent again as shown in Fig 4.1. Each time after sending the REGISTER request the timer value is set using the binary exponential back-off (BEB) algorithm. i.e. the Timer A value is incermented exponentially. But this timerA value was upper bounded by Timer B. When the Register request is received by the proxy server, a 200 OK response will be sent with a complete list of bindings in Contact header field that have been registered by this address-of-record at this registrar. The UA will also indicate how long he would like the binding to be valid. This value was set in the Expires header field.

4.2 INVITE Request

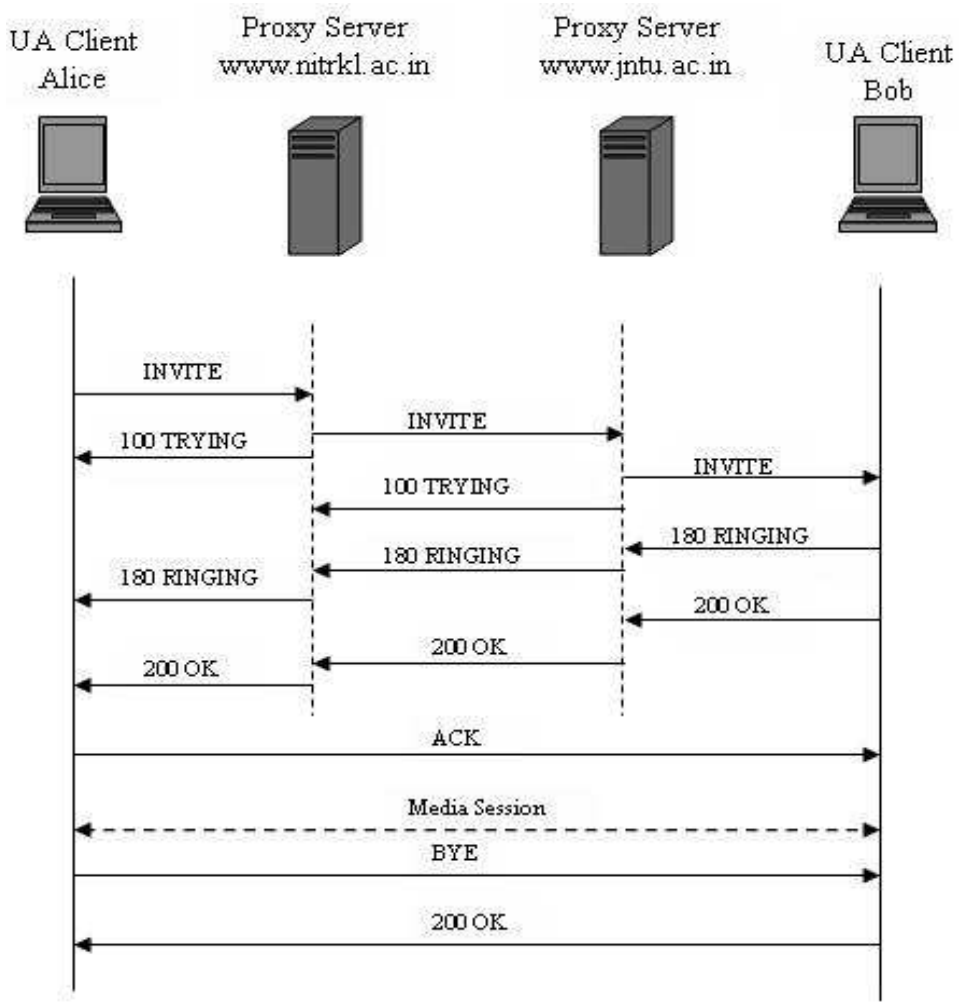


Fig. 4.2: SIP call flow example

When the UA wants to make a call it will send an INVITE request to the proxy in its domain. The SIP call flow for the INVITE request using two proxy servers is shown inf Fig 4.2.

In the simulation part two different network topologies are created with number of nodes 11 and 20 respectively. The UA Alice had registered in the domain www.nitrkl.ac.in (proxy server - p1), and the UA bob was registered in the domain www.jntu.ac.in (proxy server -p2). The proxy servers were registered with the DNS. we had implemented five different cases in session establishment. In each case we tried to reduce the session establishment time. In the INVITE request the max-forwards field is set by the time-to-live (ttl) value.

4.2.1 Case 1: SIP session establishment without congestion

In this case we assumed that no congestion was present in the network. The UA alice wants to communicate with bob@www.jntu.ac.in. Alice will send an INVITE request (1) to p1 which is in its domain, because the destination IP address is not known to alice. The proxy server p1 will immediately send a 100 Trying response to alice, and forwards the request on behalf of the UA. The proxy server p1 does not know the domain address of the destination. So p1 will send a request (2) to the DNS server (node 5 in Fig 4.3). DNS server will lookup in its database and if the proxy server was registered, it will send the response containing the address of p2 (3). If p2 is not registered then it will send a 404 (Not Found) error message.

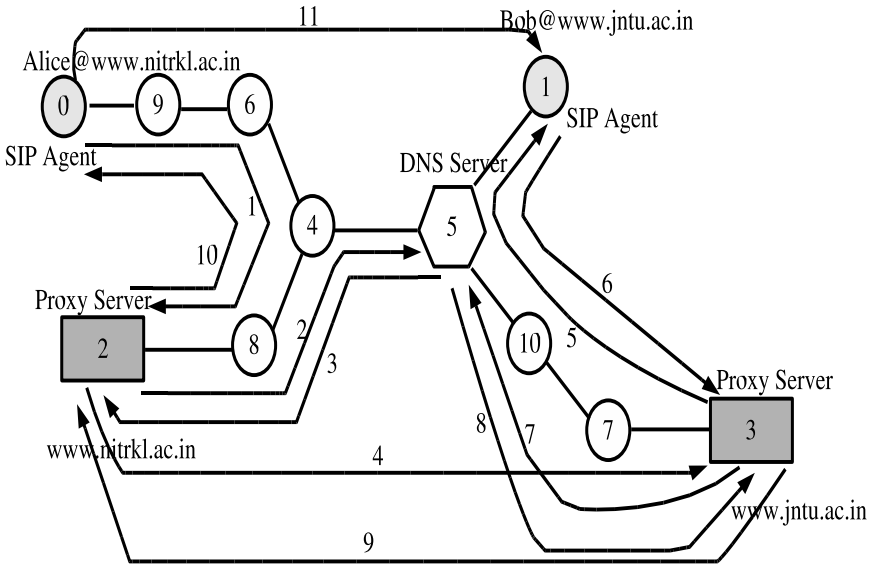


Fig. 4.3: Session establishment without congestion with 11 nodes

If p1 gets the address of p2 then p1 will send the INVITE request to p2 (4), which in turn

will send the request to Bob (5) if the life time of the UA Bob is not expired. If the time is expired then p2 will send a 301 (Moved Permanantly) response is sent. If Bob is not interested in receiving the call he will reject the call. If Bob is busy in attending an another call then a 486 (Busy Here) response will be sent. To accept the call Bob will send a 200 Ok response to p2 (6).

P2 will get the address of p1 from the DNS server (7,8), then p2 will forward the response to p1 (9), which in turn will forward it to alice (10). Now alice gets the IP address of bob, so it will send an ACK request (11) to indicate bob that the response is received. When bob receives the ACK, the session is established. The message flow for session establishment with number of nodes 11 , nodes 20 were shown in Fig 4.3, and Fig 4.4 respectively. In case of 20 nodes the time taken for the session establishment was more than with nodes 11, because the number of nodes to be traversed was more in case with 20 nodes.

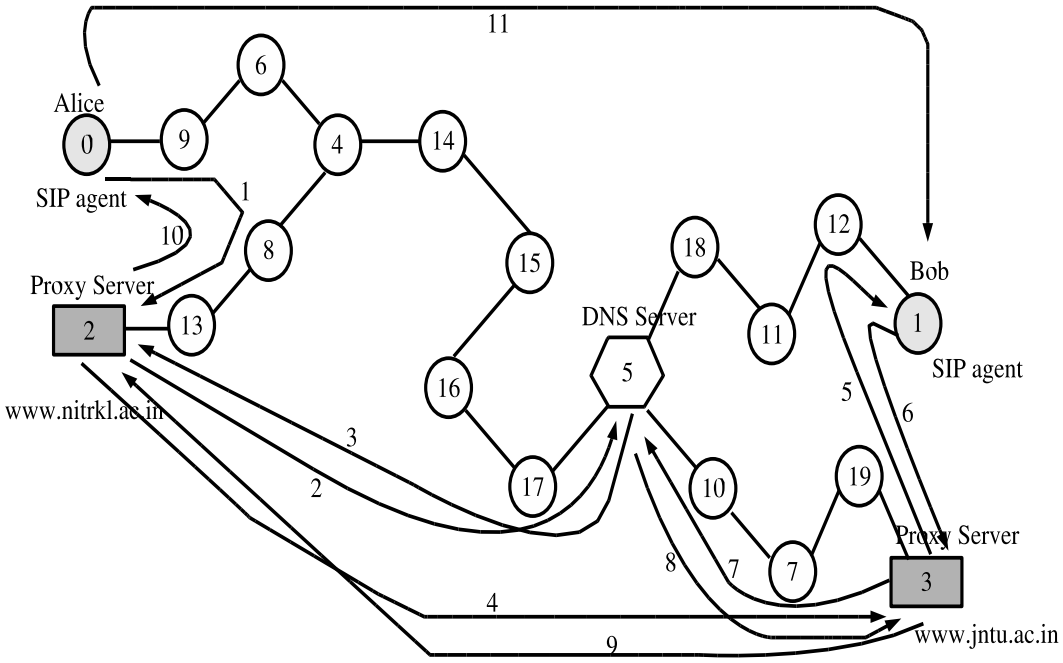


Fig. 4.4: Session establishment without congestion with 20 nodes

When Alice sends an INVITE request a timer was set. If the number of nodes to be travered was increased then the response from Bob may not be recieved within the timeout period because of delay. So alice will send the request after the timer fires. If Bob receives the INVITE message with the same sequence number then it will ignore the request.

Bob will also set a timer after sending a 200 Ok response which is upper bounded by Timer B. If the ACK was not received within that time then Bob will again send the 200 Ok response. The number of message exchanges in this case are 11. The time taken for the session establishment is shown in Table 4.1.

Table 4.1: Session seup time in case 1

	Invite	**200 OK**	**Ack**	**Setup time**
nodes 11				
Send time	2.0	2.41	2.78	
Recv time	2.38	2.77	2.86	0.86 sec
nodes 20				
Send time	2.0	2.78	3.393	
Recv time	2.75	3.383	3.6	1.6 sec

4.2.2 Case 2: Reducing message exchanges in case 1

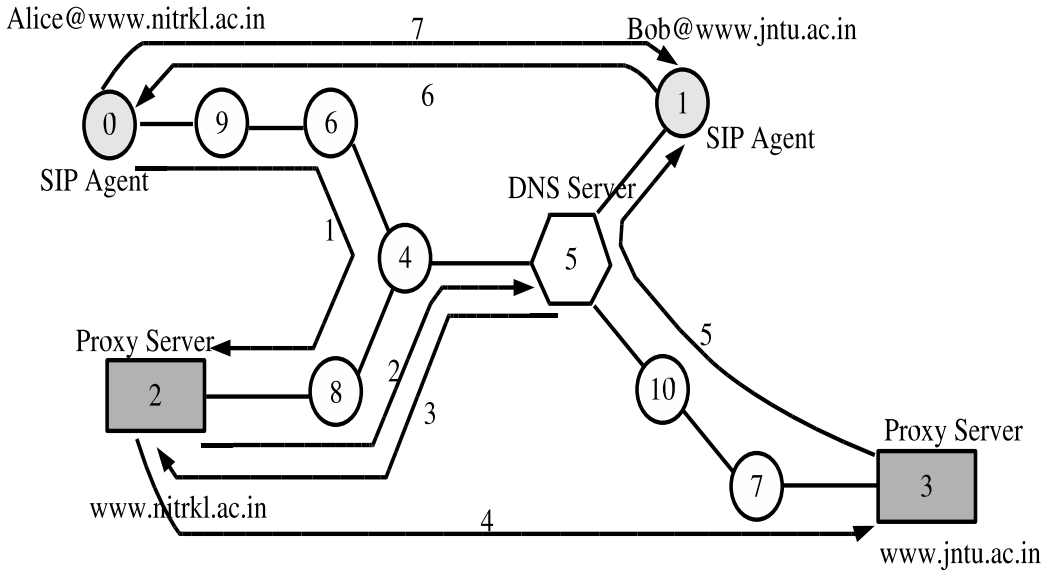


Fig. 4.5: Reducing message exchanges in case 1 with 11 nodes

In case 1 it was observed that alice will get the IP address of bob after receiving the 200 Ok response (after receiving message 5). But bob will get the IP address of alice by receiving the INVITE request. When bob receives the INVITE request, it will get alice's sip IP address, so bob can directly send the 200 Ok response to alice.

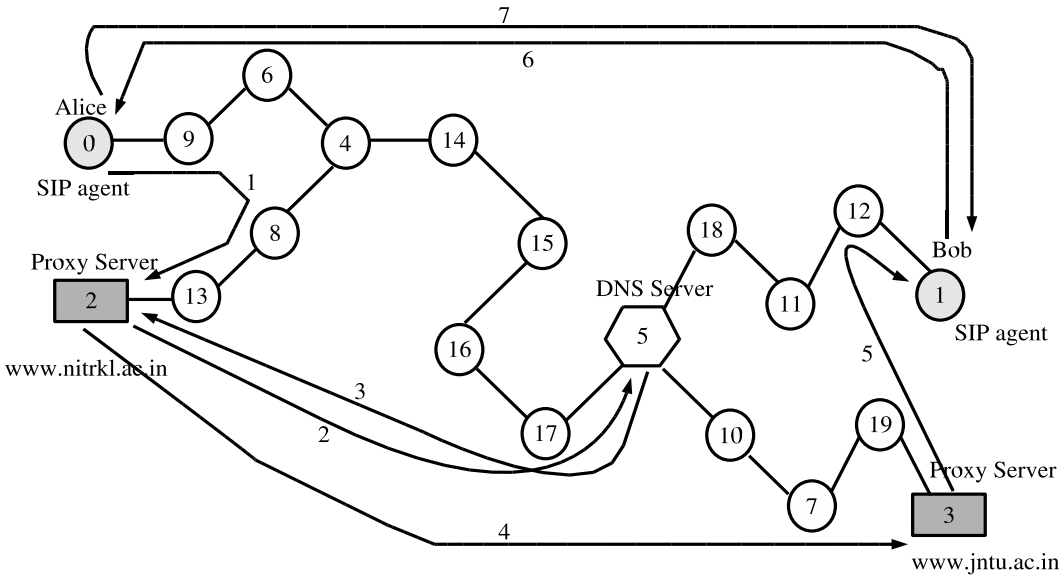


Fig. 4.6: Reducing message exchanges in case 1 with 20 nodes

In Fig 4.5 and Fig 4.6. it is shown that the number of message exchanges is reduced to 7. When alice receives the 200 Ok response, alice will obtain the ip address of bob, then the ACK message was directly sent to Bob. In this way the session is established by reducing the number of message exchanges. The time taken for session setup is shown in Table 4.2. The time taken in case 2 is less than in case 1. The delay in receiving the response from bob was reduced.

Table 4.2: Session seup time in case 2

	Invite	**200 OK**	**Ack**	**Setup time**
nodes 11				
Send time	2.0	2.41	2.494	
Recv time	2.38	2.483	2.574	0.574 sec
nodes 20				
Send time	2.0	2.78	2.99	
Recv time	2.75	2.98	3.2	1.2 sec

4.2.3 Case 3: SIP session setup during congestion

In this case we had created network traffic between the node 6 and node 4, and node 10 and node 7 as shown in Fig 4.7. Here, the message exchanges between UAs and proxy servers is same as in case 1. When the INVITE request (message number 1) the packet was lost at node 6 because the buffer was full at node 6. As the timer was set as soon as the INVITE request was sent from alice, the request will be retransmitted after the timeout period. Each time when INVITE request is retransmitted (2,3) the timer value was increased by using the BEB algorithm. In Fig 4.7 and Fig 4.8 it can be observed that the INVITE (6) was lost after p1

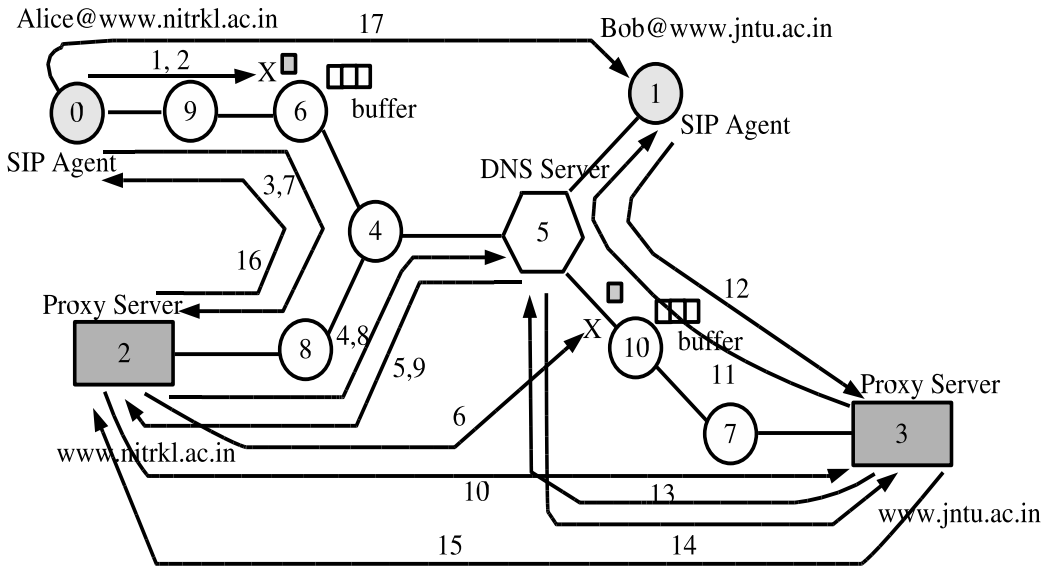


Fig. 4.7: Session setup during congestion with 11 nodes

gets reply from DNS and tring to send the request to p2. The INVITE is retransmitted after the timeout period (7). The number of message exchanges in Fig 4.7 is 17. In Fig 4.8 the INVITE

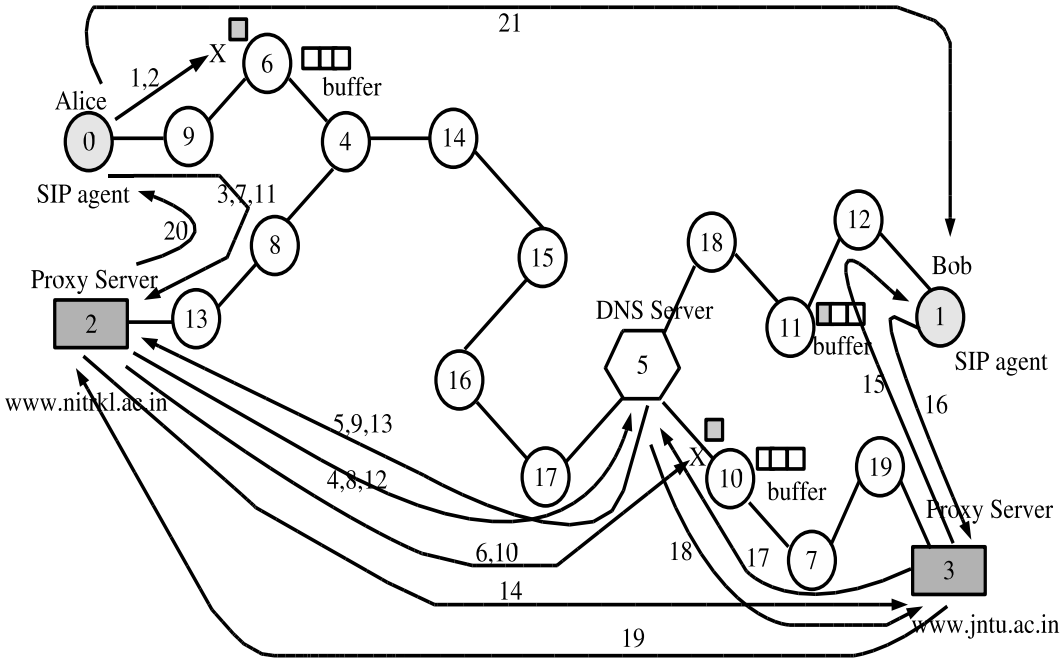


Fig. 4.8: Session setup during congestion with 20 nodes

is lost (10) at node 10 because the buffer is full. There is some traffic between nodes 11 and 12, but the INVITE packet(a packet in the buffer at node 11, message number 15) was transmitted through node 11 to bob. Bob will send the 200 Ok response to alice in the reverse path same as in case 1. Alice sends the ACK, and when bob receives the this ACK message (17 in Fig 4.7 or 21 in Fig 4.8) the session is established. The number of message exchanges in case of 20 nodes

Table 4.3: Session establishment time in case 3

	Invite	**200 OK**	**Ack**	**Setup time**
nodes 11				
Send time	2.0,2.5,3.5,5.7	6.09	6.47	
Recv time	6.06	6.46	6.55	4.55 sec
nodes 20				
Send time	2.0,2.5,3.5,5.7,9.97	10.73	11.33	
Recv time	10.7	11.32	11.54	9.54 sec

was 21 as the INVITE packet was lost one more time.

4.2.4 Case 4: Reducing messages in case 3

The main logic applied in this case was same as in case 2. In this case also network congestion was created between nodes 6 and 4, nodes 11 and 7 as shown in Fig 4.9 and Fig 4.10. In Fig 4.10 the traffic was created between nodes 11 and 12.

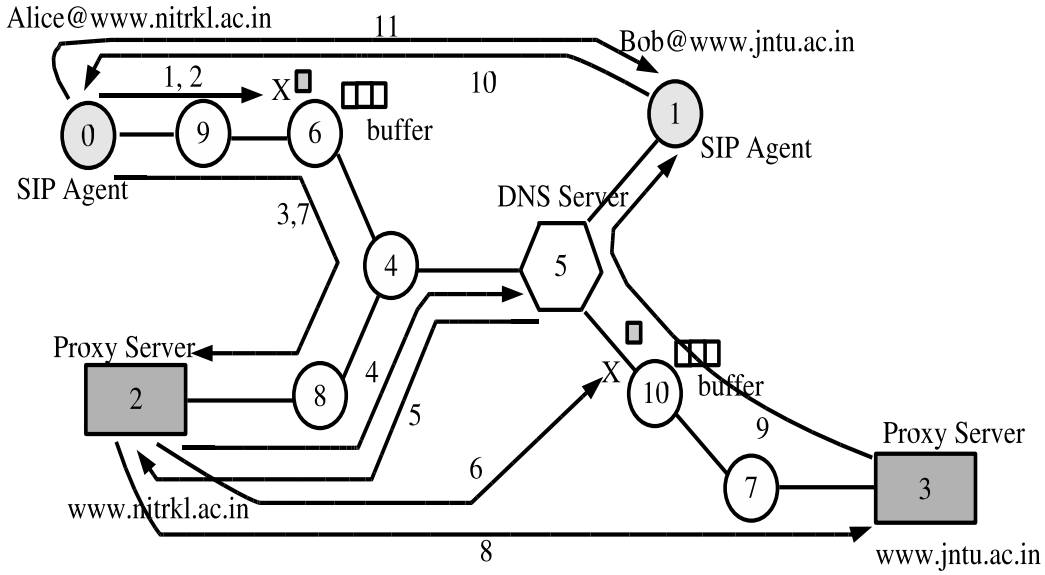


Fig. 4.9: Reducing message exchanges in case 3 with 11 nodes

When the packet (message 6) is lost at node 10, the proxy server p1 has obtained the ip address of p2. P1 will store p2 address in a database. So when the INVITE request is re-transmitted from alice, p1 will directly send the request to p2, without requesting DNS for p2 address. By this way two more messages are reduced. p2 sends request to bob, which will send the response to alice. Alice acknowledges to bob, the session is established. The number of message exchanges in this case are only 11. But the INVITE packet is retransmitted using the BEB algorithm, so each time when the INVITE packet was retransmitted the timer value

was increased exponentially. So the session establishment time was reduced by only a small amount in case of 11 nodes as shown in Table 4.4.

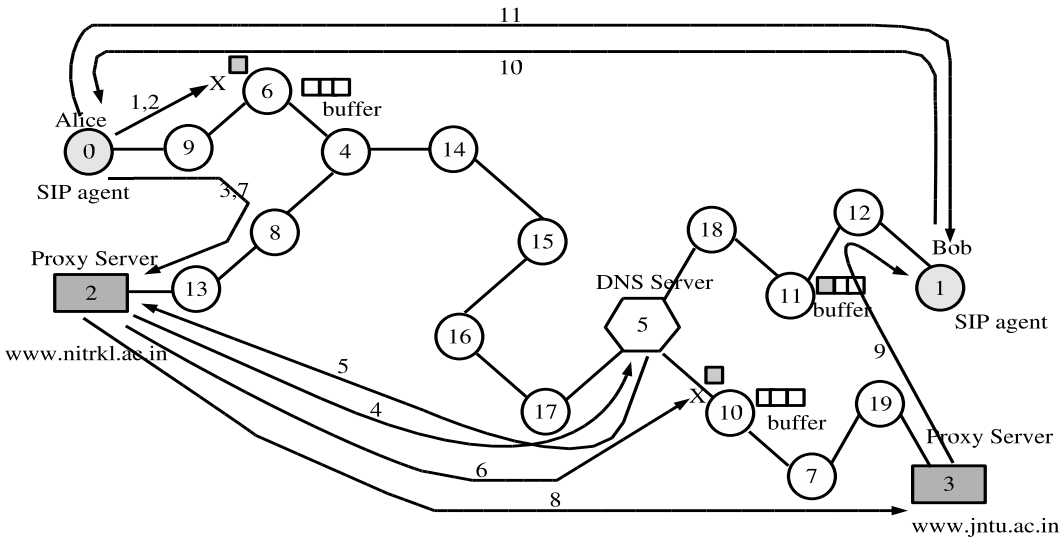


Fig. 4.10: Reducing message exchanges in case 3 with 20 nodes

Table 4.4: Session setup time in case 4

	Invite	**200 OK**	**Ack**	**Setup time**
nodes 11				
Send time	2.0,2.5,3.5,5.7	5.98	6.07	
Recv time	5.95	6.06	6.15	4.15 sec
nodes 20				
Send time	2.0,2.5,3.5,5.7	6.22	6.43	
Recv time	6.19	6.42	6.64	4.64 sec

4.2.5 Case 5: Sending INVITE thorugh proxies

In this case when alice send INVITE request the packet was lost at node6 (messages 1,2) as shown in Fig 4.11. The timer will be increased using the BEB algorithms, and INVITE (3) was retransmitted, this time it was received by proxy server p1. P1 will send a 100 Trying response to alice, send a request to DNS to obtain the IP address of p2 (4). When the reply comes from DNS (5), the address of p2 is stored in a database by p1. Now, p1 will send the INVITE request (6) to p2 and a timer was set. If the 100 Trying response from p2 was not received within that timeout period , p1 will retransmit the request to p2. In Fig 4.11 it is shown that a packet is lost at node 10 (message number 6). After a timeout period p1 will again retransmit the request(message 7) to p2, because p1 knows the address of p2. When p2 receives the INVITE

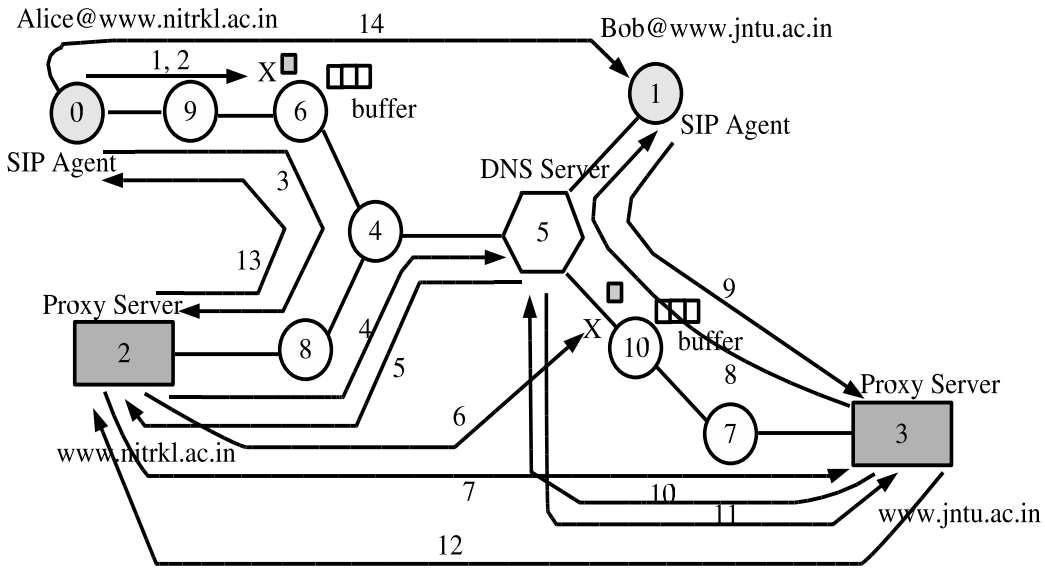


Fig. 4.11: Session establishment through proxies with 11 nodes

request , it will forward it to bob (8) and sets a timer which is upper bounded by Timer B. Bob will send a 200 Ok response (9). The response will folow the reverse path of the request.

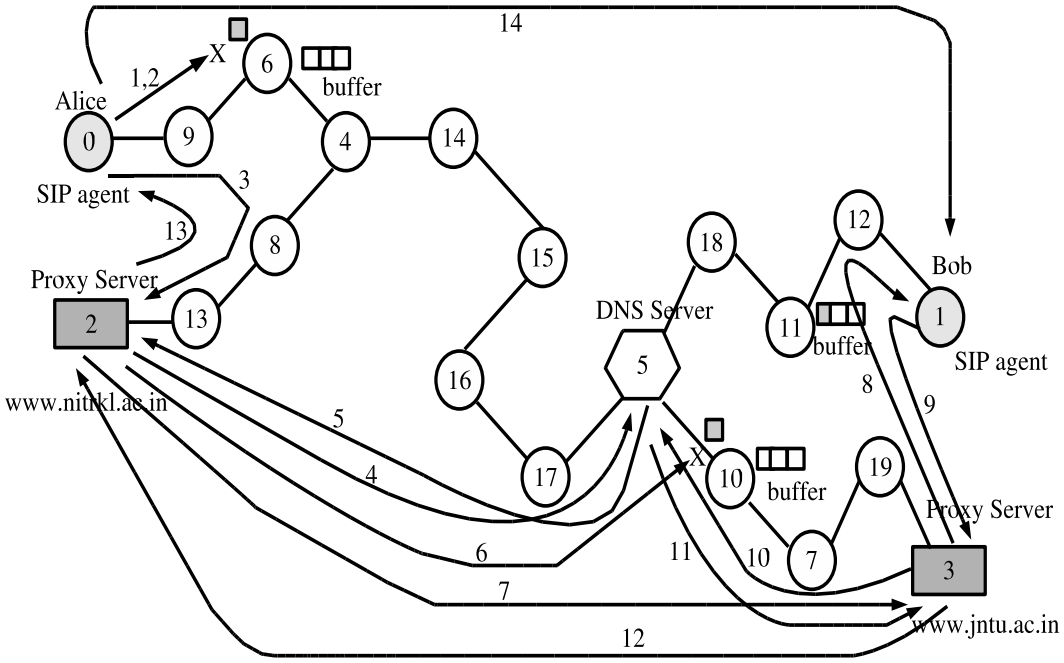


Fig. 4.12: Session establishment through proxies with 20 nodes

In Fig 4.12 the traffic was generated between nodes 11 and 12. When p2 forwards the INVITE request, it will set a timer. If the INVITE packet is lost at that node then, p2 will not receive the response, so after the timeout period the request was retransmitted by p2. The value of the timer was increased by using BEB algorithm. When alice gets the response, it will acknowledge bob and the session is established.

Table 4.5: Session establisment time in case 5

	Invite	**200 OK**	**Ack**	**Setup time**
nodes 11				
Send time	2.0,2.5,3.5			
Recv time	3.7			
Proxy send Invite	4.21	4.4	4.78	
Dest. recv. time	4.37	4.77	4.86	2.86 sec
nodes 20				
Send time	2.0,2.5,3.5			
Recv time	3.9			
Proxy send Invite	4.4	4.78	5.39	
Dest. recv. time	4.75	5.38	5.6	3.6 sec

The time taken for the session establishment is shown in Table 4.5, and the setup time was less in this case than case 3 and case 4. The number of message exchanges were 14, where as in case 4 it was 11. Though the number of message exchanges were more, the delay was reduced by sending the invite request from the proxy servers, because the timer value at the proxy servers is less, so he INVITE request is retransmitted quickly. Also, the distance (number of nodes) to be traversed was less this case.

4.3 BYE request

After the session establishment, the session can be terminated at any time by sending a BYE request. Any UA can send the BYE request to the other UA, which in turn sends a 200 OK response, and the session will be terminated. The BYE request should not be acknowledged.

4.4 Results

In simulation, the time taken for the establishment was read from the trace file in NS2. The graph is shown in Fig 4.13 for 11 nodes and 20. When there is no packet loss and no delay in receiving the packets, the number of message exchanges is reduced in case 2. If there is no congestion in the network then case 2 gives the best results. In case of congestion the time taken for session establishment is less in case 5 by reducing the distance to be traversed for the INVITE request.

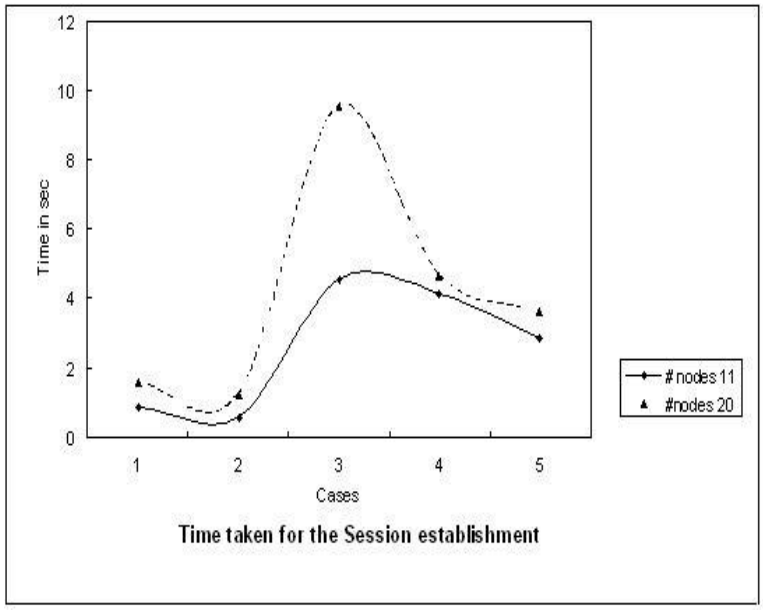


Fig. 4.13: Time taken for the Session establshment

CHAPTER 5

CONCLUSION AND FUTURE WORK

5.1 Conclusion

In the thesis we simulated the session establishment in SIP. In SIP the sender UA can not send the invitation directly to the destination, because the IP addresses are not known. For obtaining the IP addresses proxy servers and location servers are used. From the simulation we can conclude that the session establishment time can be reduced as follows :

- When the destination user agent gets the INVITE request it gets the source ip address, so the destination user agent can send the response directly to the source. In this way the number of message exchanges will be reduced.
- The user agent will use the BEB algorithm for setting timer values. When the packet is lost, the INVITE packet is retransmitted and the timer value is exponentially increased. When there is packet loss in the network, the proxy servers can send the INVITE request to the destination user by maintaining the timers. By sending the packets from proxy server will reduce the delay, because the timer value at the proxy will be less than the timer value at UA, and the number of nodes to be traversed is less.

5.2 Future Work

As a future work this architecture can be extended for sending audio/video data by using different compression algorithms and try to reduce the jitter delay. When the audio packets are sent then in the response a field can be added indicating the jitter delay and the number of packets lost. So that if the packet loss is very high then the UA can interrupt the session by sending the CANCEL request or can terminate the session by sending a BYE message. Otherwise if possible the packets may be sent by following a different path.

Bibliography

[1] Simone Leggio, ”Session Initiation Protocol and Signaling: state-of-the-art and Qos related open issues,” January 2006.

[2] Samir Chatterjee, Bengisu Tulu, Tarun Abhichandani, and Haiqing Li, ”SIP-Based Enterprise Converged Netwoks for Voice/Video-over-IP: Implementation and Evaluation of Components,” IEEE journal, VOL. 23, NO. 10, October 2005.

[3] J.Rosenberg, H.schulzrinne, G.Camarillo, A.Jhonston, J.Peterson, R.Sparks, N.Handley, and E.Schooler, ”SIP: Session Initiation Protocol,” Internet Engineering Task Force, RFC 3261, 2002.

[4] G.De Marco, G.Iacovoni, and L.Barolli, ”A Technique to Anaalyse Session Initiation Protocol Traffic,” IEEE Proceedings of the 2005 11th International Conference on Parallel and Distributed Systems.

[5] M.Handley, V.Jacobson, ”SDP : Session Description Protocol,” Internet Engineering Task Force, RFC 2327, 1998.

[6] D.Wills, B.Cambell, ”Session Initiation Protocol extension to Assure Congestion Safety”, Internet Draft (work in progress), February 2003, draft-ietf-sip-congestsafe-01.txt.

[7] M.Brunner ”Requirements for Session Policy for the Session Initiation Protocol (SIP)”. Inernet Draft (work in progress). June 2003. draft-ietf-nsis-req-o8.txt.

[8] J.Rosenberg, H.schulzrinne, ”An Offer/Answer Model with the Session Description Protocol (SDP)”, RFC 3264, June 2002.

[9] J. Rosenberg, H. Schulzrinne, ”Session Initiation Protocol (SIP): Locating SIP Servers,” RFC 3263, June 2002.

[10] NS2 documentation http://www.isi.edu/nsnam/ns/ns-documentation.

[11] Internet: Kevin Fall,Kannan Varadhan, (August 17, 2006).The ns Manual. Available e-mail : kannan@catarina.usc.edu.

[12] G.Camarillo, W.Marshall, J.Rosenberg, ”Integration of Resource Management and Session Inititation Protocol (SIP)”. RFC 3312, October 2002.

[13] H. Schulzrinne, B. Volz, ”Dynamic Host Configuration Protocol (DHCPv6) Options for Session Initiation Protocol (SIP) Servers,” RFC 3319, July 2003.

[14] S. Donovan, J. Rosenberg, ”Session Timers in the Session Initiation Protocol (SIP),” RFC 4028, April 2005.

[15] C. Jennings, F. Audet, J. Elwell, ”Session Initiation Protocol (SIP) URIs for Applications such as Voicemail and Interactive Voice Response (IVR),” RFC4458, April 2006.

[16] S.Dunstun, ”Converged voice and data services in the network,” *Telecomm.* J.Australia,vol.51,no. 2,pp. 17-31, 2001.