ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ
Τμήμα Διδακτικής της Τεχνολογίας και Ψηφιακών Συστημάτων

Πολυμεσικές Υπηρεσίες πάνω από IP δίκτυα

Γιανναράκης Νικόλαος

ΜΕΤΑΠΤΥΧΙΑΚΗ ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ιούλιος 2008

Περίληψη

Σε μια εποχή που το ευρυζωνικό διαδίκτυο διαδίδεται όλο και πιο πολύ, ο κόσμος της τεχνολογίας και των τηλεπικοινωνίων στρέφεται στην μετάδοση φωνής πάνω από IP δίκτυα και στην ενοποίηση των υπηρεσιών. Το κυρίαρχο πρωτόκολλο που χρησιμοποιείται για την μετάδοση φωνής πάνω από IP δίκτυα είναι το Session Initiation Protocol (SIP).

Στην παρούσα διατριβή γίνεται μια μελέτη του SIP όσον αφορά την διαχείριση πολυμεσικών συνόδων σε εφαρμογές Voice Over IP (VoIP). Εξετάζονται κάποιες SIP υλοποιήσεις ανοιχτού κώδικα και χρησιμοποιείται το Asterisk PBX για την υλοποίηση και παραμετροποίηση κάποιων υπηρεσιών.

Η διατριβή οργανώνεται ως εξής:

Το κεφάλαιο 2 δίνει μια επισκόπηση της διαχείρισης πολυμεσικών συνόδων. Αυτή η επισκόπηση εισάγει τις βασικές διαδικασίες διαχείρισης συνόδων, λειτουργίες και εφαρμογές, με μια έμφαση στο SIP. Γίνεται επίσης μια σύγκριση του SIP με το H.323 πρωτόκολλο που είναι πρωτόκολλα που χρησιμοποιούνται κυρίως για VoIP, ενώ γίνεται και μια αναφορά στην ποιότητα των υπηρεσιών που προσφέρονται από το SIP.

Στο κεφάλαιο 3, οι υπάρχουσες open source SIP υλοποιήσεις παρουσιάζονται και συγκρίνονται. Ανάμεσα σε αυτές, το Asterisk PBX θα χρησιμοποιηθεί ως πλατφόρμα δοκιμής σε αυτήν την διατριβή. Οι λόγοι που οδήγησαν σε αυτή την επιλογή παρουσιάζονται. Οι κυριότεροι λόγοι είναι οι πολλές λειτουργίες που προσφέρονται απο το Asterisk, η αλληλεπίδραση του Asterisk με διάφορες υπηρεσίες που προσφέρονται από το λειτουργικό σύστημα UNIX και η δυνατότητα να προγραμματίσει κάποιος σε διάφορες γλώσσες προγραμματισμού ώστε να αλληλεπιδράσει με τις διεπαφές που παρέχει το Asterisk για τον λόγο αυτόν.

Στο κεφάλαιο 4, το Asterisk μελετάται λεπτομερέστερα: τα χαρακτηριστικά του γνωρίσματα, τα σενάρια χρήσης του και η αρχιτεκτονική του, παρουσιάζονται. Έμφαση δίνεται στην χρησιμοποίηση του Asterisk PBX σαν VoIP σύστημα.

Στο κεφάλαιο 5, το Asterisk, οι υπηρεσίες voicemail to email και click to dial πειραματίζονται στο εργαστήριο. Επίσης γίνονται κάποια πειράματα χρησιμοποιώντας την προώθηση κλήσεων σε διάφορες καταστάσεις του χρήστη που έχει ενεργοποιημένη την υπηρεσία. Τα αποτελέσματα παρουσιάζονται και αναλύονται στο τέλος αυτού του κεφαλαίου. Τα αποτελέσματα αφορούν το κατά πόσο οι υπηρεσίες που υλοποιήθηκαν και που προσφέρει το Asterisk είναι σταθερές και συμβατές με τα standards του SIP πρωτοκόλλου.

Τέλος, στο κεφάλαιο 6, η εργασία συνοψίζεται και συνάγονται κάποια συμπεράσματα. Επίσης γίνεται μια αναφορά σε κάποιες ιδέες για μελλοντική δουλειά που αφορά την περαιτέρω εξέλιξη των υπηρεσιών που υλοποιήθηκαν καθώς και την υλοποίηση κάποιων νέων υπηρεσιών.

List of Abbreviations

3GPP Third Generation Partnership Project
ACE Application Creation Environment
ANSI American National Standards Institute
API Application Program Interface
B2B UA Back-To-Back User Agent
CPL Control Policy Language
CPU Central Processing Unit
CSPS Cisco SIP proxy servers
DHCP Dynamic Host Configuration Protocol
DNS Domain Name Service
ENUM E.164 number and DNS
FXO Foreign eXchange Office
FXS Foreign eXchange Service
GNU Gnu's Not Unix
GSM Global system for mobile communication
GUI Graphic User Interface
H.323 A standard approved by the ITU that defines how audiovisual conferencing data is transmitted across networks.
HTTP HyperText Transport Protocol
IAX Inter-Asterisk eXchange
IETF Internet Engineering Task Force
IM Instant Messaging
IMP Instant Messaging and Presence
IMS IP Multimedia Subsystem
IPv4 Internet Protocol, version 4
IPv6 Internet Protocol, version 6
ISP Internet Service Provider
ITU International Telecommunication Union
IVR Interactive Voice Response
MGCP Media Gateway Control Potocol
MIME Multipurpose Internet Mail Extensions
NBEN Northeast Business Environmental Network

OSS Open Source Software
PBX Private Branch eXchange
PDA Personal Digital Assistant
PHP Hypertext Preprocessor
PRI Primary-Rate Interface
PSTN Public Switched Telephone Network
QoS Quality of Service
RADIUS Remote Authentication Dial-In User Service
RFC Request For Comments
RPC Remote Procedure Call
RSS Really Simple Syndication
RTP Realtime Transport Protocol
SDP Session Description Protocol
SER SIP Express Router
SIP Session Initiation Protocol
SIPC Simply Interactive Personal Computer
SMS Short Message Service
SMTP Simple Mail Transfer Protocol
SQL Structured Query Language
SSL Secure Sockets Layer
TCP Transmission Control Protocol
TLS Transport Layer Security
TM Transaction Management
UA User Agent UAC User Agent Client
UAS User Agent Server
UDP User Datagram Protocol
URI Uniform Resource Identifier
VoIP Voice over IP
VPN Virtual Private Network
XML eXtensible Markup Language

List of Figures

List of Tables

1 Introduction

1.1 Background

As a new application-layer signaling protocol targeting on providing simplified session establishment, Session Initiation Protocol (SIP) [51] has been gaining more and more attention lately. SIP is independent of the underlying networks and that is probably the reason that makes SIP popular among the Internet communities. These communities are also called virtual communities that typically use Internet to share information and values upon common interests, for example, implementations of Voice over IP (VoIP). Many new applications have been implemented based on SIP; for example, VoIP uses SIP for connection establishment and 3GPP [68] has selected SIP as the basic signaling protocol in IP Multimedia Subsystem (IMS) [2,3,4].

1.2 Task and Scope

The primary task of this thesis is to evaluate, test and experiment some SIP services and features. A voicemail to e-mail service and a click-to-call service are implemented using some scripts and the proper configuration. The environment for this project is my private home LAN network connected to the Internet with an adsl broadband connection 1Mbit with one common ISP.

In order to have an in-depth understanding of the applicability of SIP to the mentioned services, in this thesis SIP is reviewed thoroughly ranging from its basic features, operations, and functions to its extensions. Other standards, such as H.323 [37], are also discussed in an effort to reveal the advantages that SIP provides to the Internet community.

Finally, some existing SIP-based implementations in the public domain are introduced to provide information on SIP-related implementations and their status in general.

The scope of this thesis is not limited to exploring SIP in general. An evaluation of a single SIP-based product is also carried out in this thesis, and this evaluation focuses on the voicemail to email and click-to-call services.

1.3 Thesis Outline

The thesis is organized as follows: Chapter 2 gives an overview of session management. This overview introduces the basic session management operations, functions and applications, with an emphasis on SIP. In Chapter 3, the existing open source SIP implementations are presented and compared. Among them, Asterisk will be used as the test platform in this thesis. In Chapter 4, Asterisk is studied in more detail: its features, scenarios and architecture, etc are presented. In Chapter 5, Asterisk, the voicemail to email service and the click-to-call service is experimented in the Laboratory. The results are presented and analyzed in the end of this chapter. Finally, in Chapter 6, the work is summarized and conclusions are drawn. Some ideas for future work are also discussed.

2 Session Management Protocols

In this chapter Session Initiation Protocol is covered in depth and is compared with H.323.

2.1 Session Initiation Protocol

Session Initiation Protocol (SIP) is an “application-layer control (signaling) protocol for creating, modifying, and terminating sessions with one or more participants” [51]. It is one of the Internet’s most popular signaling protocols used in applications that require session setup and parameter negotiation before the actual communication, as well as session management and termination. Applications such as Voice over IP (VoIP) telephony, multimedia conferences, chat sessions, and other interactive communications can take advantage of SIP.
RFC 3261 serves as the base standard for SIP. It specifies methods that are also called requests, responses, and basic operations for creating, modifying, and terminating sessions.

It also specifies behavior of user agents (UAs) and servers in each method and defines headers for these methods.

2.1.1 SIP Standardization

SIP standards were developed by the Internet Engineering Task Force (IETF) [28]. The first version of the SIP standard, RFC 2543 [26], was made within the Multiparty Multimedia Session Control (MMUSIC) working group [31] in March 1999. In June 2002, a revised version of the SIP standard, RFC 3261 [51], was developed by another IETF working group, the SIP working group [32], which

was chartered to continue the development of SIP. RFC 2543 was obsoleted by RFC 3261, which is referred as the base SIP standard in this thesis.

The SIP working group continues the development work. In addition, 3GPP [68] has selected SIP as the basic signaling protocol in its active IP Multimedia Subsystem (IMS) work program [2,3,4]. SIP event package for presence [49] was developed by network working group in August 2004.

2.1.2 SIP Operations

In this subsection, SIP entities and methods will be introduced and an example of using SIP to set up a session will be show to illustrate the SIP operations.

SIP Entities

There are three main entities in a SIP network: User Agent (UA), Server, and Database.Following paragraphs explain the tasks of these entities.

A User Agent (UA) is an end device (terminal) in a SIP network. UA performs two basic functions: listen for incoming SIP messages and send SIP messages upon user actions or incoming messages. UA can be a SIP phone, or PC running a SIP client software, etc.

UA contains a User Agent Client (UAC) and a User Agent Server (UAS). UAC initiates requests, while UAS generates responses to the received requests.

The base SIP standard RFC 3261 defines three types of SIP servers: Proxy Server, Redirect Server and Registrar Server. The division is logical, and different type of servers can physically be in a single server.

A SIP Proxy Server is an intermediary entity that acts as a server or a client, making requests on behalf of clients. Its primary job is to ensure that a request is sent to an entity that is routing wise close to the targeted user. A proxy interprets and, if necessary, rewrites specific parts of a request message before forwarding it.

Redirect Server directs the client to contact a user in alternate set of locations. The main difference between a proxy and a redirect server is that proxy server participates in the whole transaction while the redirect server only returns locations of a user Registrar Server accepts REGISTER requests and places the information it receives in those requests into the location service (database). Often, a registrar server for a domain is co-located with the proxy server for that domain.

Database is used by a SIP redirect or proxy server to obtain information about a user's possible location(s). Thus, the database is often called location server. It contains bindings of keys to contact addresses. The bindings can be created and removed in many ways; the base SIP standard defines a REGISTER method that updates the bindings. The next section describes this method among other SIP methods.

SIP Methods

SIP is a simple request/response protocol. A SIP request is also known as a method. In the base SIP standard RFC 3261, six methods are defined, as is shown in Table 1.

Table 1 SIP Methods [51]

Method	Description
INVITE	To set up a session.
ACK	To acknowledge the final response to INVITE.
BYE	To terminate a session.
CANCEL	To cancel a pending session.
REGISTER	To register a user's Uniform Resource Identifier (URI).
OPTION	To query servers about their capabilities.

Methods determine what kind of applications or services can be deployed in a SIP-enabled network. Many other methods, such as SUBSCRIBE, NOTIFY, and MESSAGE, have been proposed as extensions to the base SIP. They enable more applications like presence and instant messaging. Those extensions are specified in separate RFCs or Internet drafts that may become RFCs. The SUBSCRIBE and NOTIFY methods, for example, are specified in RFC 3265 [52], and the MESSAGE method is specified in RFC 3428 [10]. The description of these methods is shown in Table 2.

Table 2 SIP extension methods [10,52]

Method	Description
SUBSCRIBE	To request asynchronous notification of an event or set of events at a later time.
NOTIFY	To notify a SIP node that an event, which has been requested by an earlier SUBSCRIBE method has occurred. It may also provide further details about the event.
MESSAGE	To allow the transfer of Instant Messages. The content is carried in the form of MIME body part. MESSAGE does not initiate a SIP dialog.

Basic Operations of SIP

Session setup is a basic function of SIP. Figure 1 shows a typical session setup presented in RFC 3261 [51]: Two users, Alice and Bob, exchange SIP messages to setup a media session.

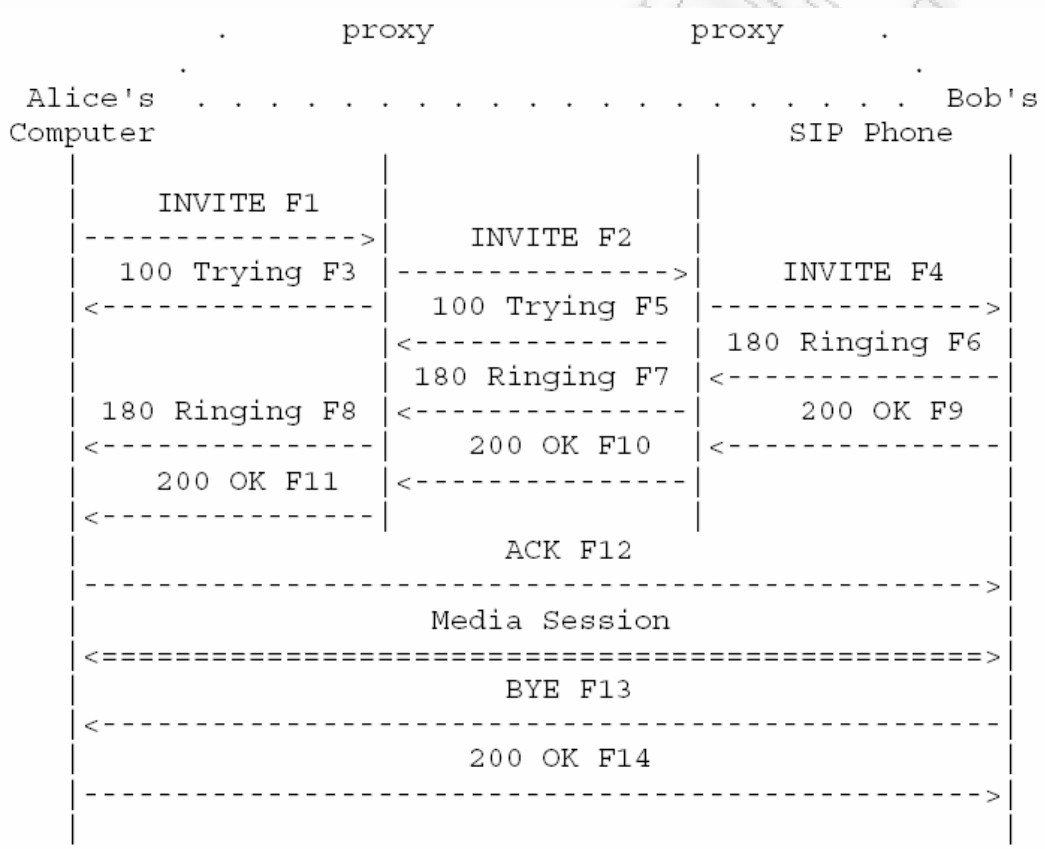


Figure 1 SIP session set up example [51]

In this example, Alice uses a SIP application on her computer to call Bob on his SIP phone over the Internet. There are also two SIP proxy servers that act on behalf of Alice and Bob to facilitate the session setup. Before creating any

sessions, Alice's computer needs some information about the SIP proxies of its own domain. A SIP proxy address can be, for example, be configured manually or obtained dynamically by using Dynamic Host Configuration Protocol (DHCP). Alice's computer does not need to know anything about the other proxy beforehand. The operation of the session setup can be described as follows:

First, Alice's computer sends an INVITE request addressed to Bob's SIP Uniform Resource Identifier (URI). A URI, generic description in [8], identifies a user (e.g., Bob in this example) or a service (e.g., a mailbox on a messaging system) to which a request is being addressed.
Bob's SIP URI, sip:bob@example.com, is given in Figure 2. The INVITE request contains a number of header fields that provide additional information about a message, e.g., the call identifier, Bob's address, Alice's address, and information about the type of the session. The INVITE request might look like it is shown in Figure 2. Detailed explanation of the request can be found in RFC 3261 [51].

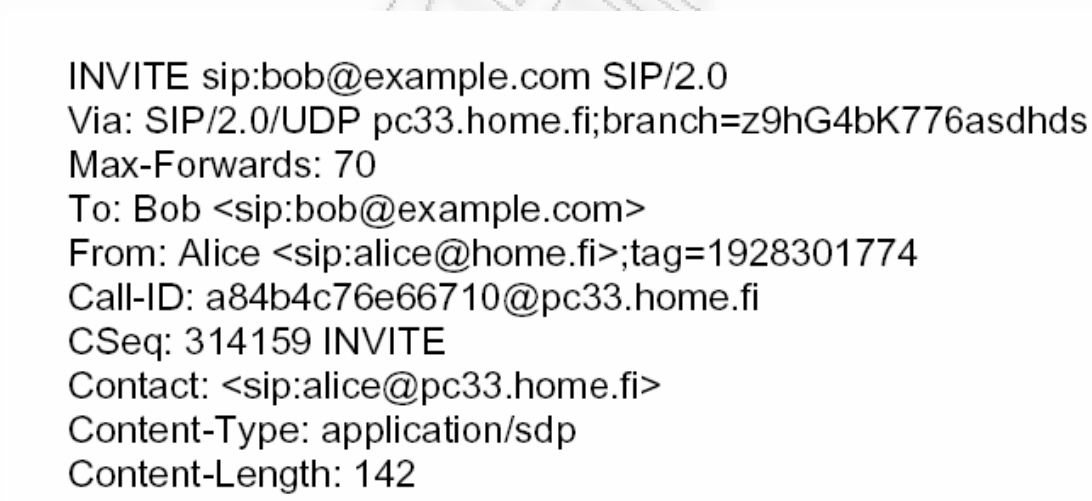


Figure 2 INVITE request [51]

Since Alice's computer does not know the location of Bob or his SIP server, it sends the INVITE message to the SIP proxy server serving Alice's domain. When receiving the INVITE request, the proxy server returns a 100 Trying

response to Alice’s computer to indicate that it has received the INVITE and is processing the request. Meanwhile, the server obtains the IP address of the example.com proxy server by using a particular Domain Name Service (DNS) query for SIP servers. The query works similarly to finding the electronic mail server of a domain. The proxy server adds an additional via header field value that contains its own address to the INVITE message and then forwards the request to the proxy server of example.com. Similarly, the proxy server of example.com returns a 100 response to the proxy server. The example.com server looks up Bob’s current IP address from the database, or a location service. It then adds its own address to another via header field of the INVITE message and forwards it to Bob’s SIP phone.

When Bob’s SIP phone receives the INVITE request, it rings and returns a 180 Ringing response, which is routed back to Alice’s computer. Each proxy uses the via header field to determine where to send the 180 response and removes its own address from the message. When Alice’s computer receives the 180 response, it rings or displays a message on the screen.

In the example of Figure 1, Bob answers the call and picks up the handset. His SIP phone then sends a 200 OK response to indicate that the call has been answered. The 200 OK message contains the Session Description Protocol (SDP) [24]. The SDP message exchange is not shown in Figure 1. However, its purpose is to describe the type of the session that Alice offers and Bob is willing to establish with her. Finally, Alice’s computer sends back an acknowledgment message to Bob’s SIP phone. The ACK message can be sent directly from Alice's computer to Bob's SIP phone, because the endpoints have learned each other's addresses from previous messages and the proxies are no longer needed.

Now, Alice and Bob have media session agreed in the SDP exchange and media packets can be directly routed to their destination.

The last two messages in the example close the session. Here, Bob hangs up first and a BYE message is sent directly to Alice's computer. Alice confirms closing by sending 200 OK response.
The above description is about the operation of session setup. Other operations in SIP, such as registration, options, and cancellation, can be found in the base SIP standard RFC 3261 [51].

2.1.3 SIP Functions

The functions of SIP are determined by the methods defined by the protocol. They can be divided into two broad categories: session-related functions, and non-session-related functions [58]. The six methods defined in the base SIP, as shown in Table 1, are mostly related to session-related functions, while many new methods defined in the SIP extensions are mostly related to non-session-related functions. For example, event subscription and notification is presented here due to their important role in IMP service. These functions and their related methods are described below.

Session-related functions are address resolution, session setup, media negotiation, session modification, session termination and cancellation, mid-call signaling and call control.

Address resolution helps to obtain the username and its IP address from the known SIP URI, e.g., from sip: Bob@example.com to sip: Bob@100.101.102.001. The address resolution may need help of the DNS servers and the location service. The SIP method used is INVITE sent by the caller to any SIP server knowing about the callee's location.

Session setup is the primary function of SIP. The operation has been introduced in the previous example: a successful SIP session setup involves an INVITE

message, optional provisional responses, a 200 OK final response, and an ACK message.

Media negotiation is part of the process of establishing a session, as can be seen in the above description of the session setup operation. Actually, SIP itself does not provide the media negotiation: Session Description Protocol (SDP) is used inside the SIP session setup to negotiate the actual media session parameters, such as the media type, codec, IP addresses, and port numbers for each media stream.

Session modification. An on-going session can be modified by sending another INVITE request. This is referred to as re-INVITE and can be sent by either participant. A successful re-INVITE may change any of the media characteristics, such as the session type, codec used, and even the source IP address and the port number. However, a failed re-INVITE will not cause the on-going session to cease.

Session termination and cancellation are the two ways to end a session. Either party in a session can send a BYE to terminate the session. Similarly, either party can end a CANCEL message to end a session before the session setup is completed.

Mid-call signaling function is realized by using the INFO method, which is specified in a SIP extension RFC 2976 [16]. INFO allows signaling message to be exchanged between two parties during a session. Unlike re-INVITE, INFO does not change the parameters of the session. For example, INFO can be used to carry mid-call Public Switched Telephone Network (PSTN) signaling messages between PSTN gateways, or to carry wireless signal strength information in support of wireless mobility application.

Call control allows a third party to direct or control a call between two other parties. It can be realized in two ways. One is to employ a controller that intercepts the SIP INVITE request, answers it, and then proxies it to a third party. Another way is use the REFER method, which is specified in a SIP extension RFC 3515 [61].

Non-session-related functions include mobility, message transport, event subscription and notification, and authentication and security. These functions are described below.

SIP has the built-in ability to support personal **mobility**. It means that a SIP user, identified by a unique personal identity, SIP URI, e.g., sip: Bob@example.com, can originate and receive calls on any terminal in any current location, be it at home or at office. The mobility support is realized by using the REGISTER method, in which a SIP user can register the URIs where he/she wishes to receive calls with the registrar. For instance, when Bob goes back home after work, he sends a REGISTER request to the registrar, indicating that his current address is sip: Bob@home.com. Then the proxy or redirect server will connect any incoming calls to Bob's home address, even if they are addressed to Bob's other address sip:Bob@example.com. Nevertheless, this cannot happen in the middle of a session, because existing sessions are bypassing the registrar, and will become invalid when the user moves. To keep communicating, a new session must be established.

Message transport. A SIP user can send instant messages to another user with a MESSAGE request [10]. MESSAGE request, unlike INFO method, does not require an established session. In addition, MESSAGE request supports MIME. Message transport is an example of SIP supporting presence and instant messaging.

Event subscription and notification are other methods for SIP to support instant communications. It is the ability to request and receive notification when a certain event occurs. An event can be a friend's presence status, new voice message in one's voice mailbox, etc. The request is made by the subscriber sending a SUBSCRIBE method [52] to the notifier and the reply is delivered to the subscriber by sending a NOTIFY message [52]. A SUBSCRIBE request and the corresponding 200 OK response then create a SIP session – a subscription. This session will be terminated either when expires (the session duration is determined in the 'Expires' header) or explicitly by sending an unsubscribe request (a SUBSCRIBE request with an 'Expires' header value of 0) [52].

Authentication and security. SIP provides authentication between UA and a server or another UA. They are based on a shared secret and a challenge/response protocol borrowed from the Hypertext Transfer Protocol (HTTP) authentication. The message exchanges in the UA-to-UA authentication and the UA-to-server authentication are shown in Figure 3 and Figure 4 respectively.

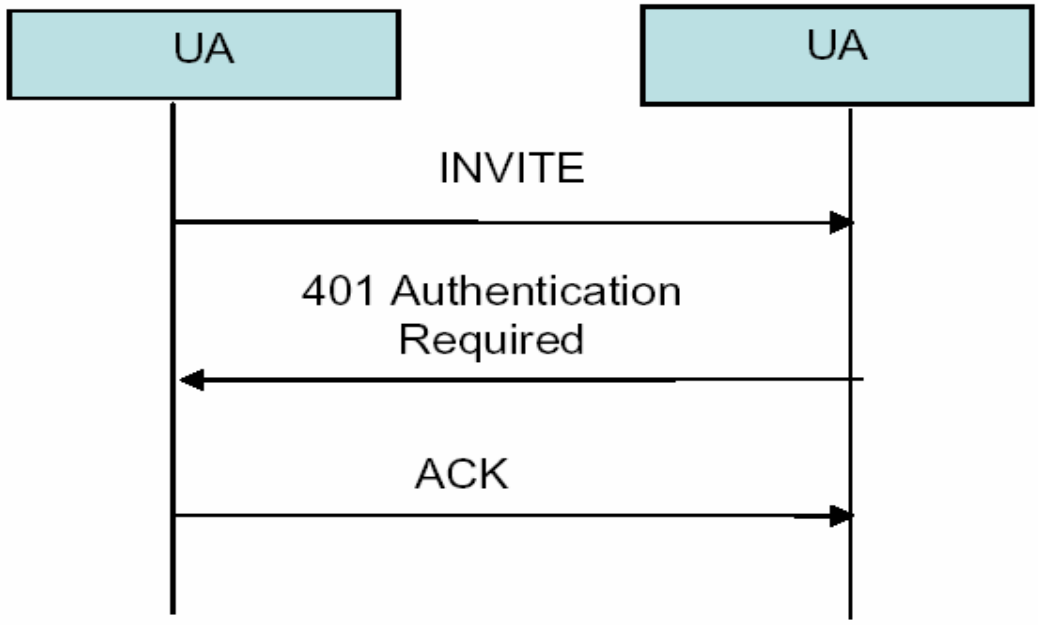


Figure 3 UA-to-UA authentication

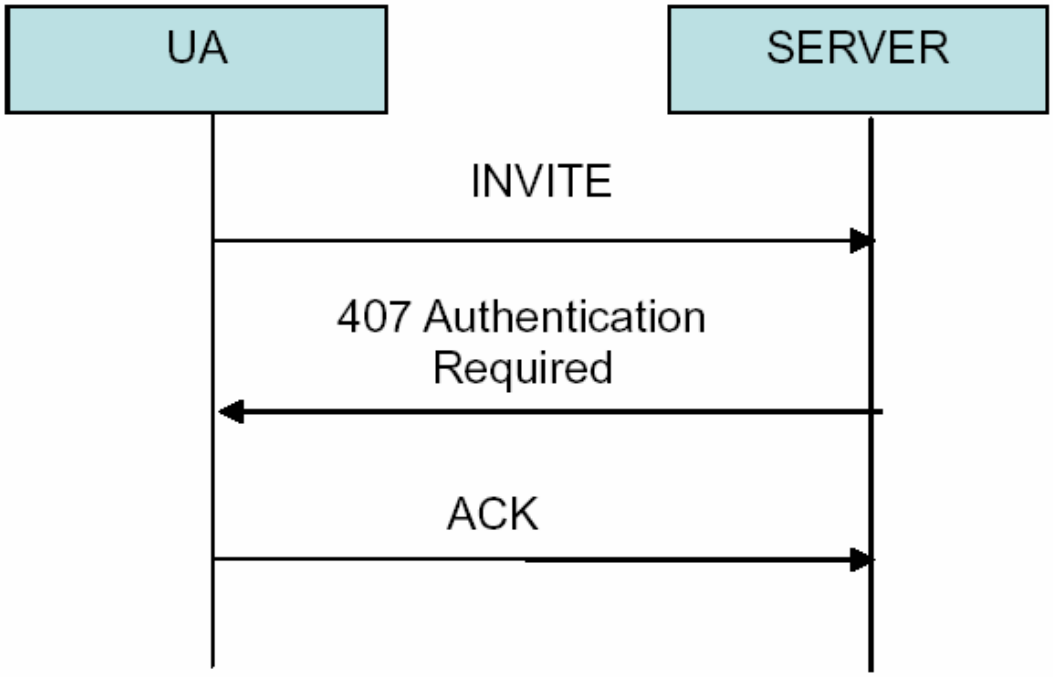


Figure 4 UA-to-Server authentication

SIP is designed to be **extensible**. UAs can implement new extensions using new headers and message bodies without requiring intermediate servers such as proxies to support the extensions. The proxies just forward the unknown messages to their destinations.

2.1.4 SIP Extensions and On-going SIP Development

The base standard of SIP does not specify every detail of the operations of SIP methods. Moreover, only a few methods are defined in the base SIP standard, and they are not enough to support some growing applications like presence and instant messaging. In order to support more applications and avoid the encountered problems, more methods need to be specified. For instance, the base SIP does not define a keep-alive mechanism for the sessions it establishes.

When a BYE message fails to be sent at the end of the session due to, e.g., network problem, a call to a stateful proxy will not know whether the session has ended or not. In this situation, the proxy will retain state for the call and has no deterministic method of determining when the call state information no longer applies. An extension [17] is proposed to solve this problem, as explained in the next paragraph. Up to date, some extensions have been standardized, e.g., RFC 3262, RFC 3263, RFC 3264, RFC 3265, and RFC 2796. These extensions were developed mainly by the SIP working group and this working group will continue to act as the main force to specify and develop SIP extensions.

Currently, many new enhancements, mechanisms, and methods for the base SIP are being proposed. They are first documented as Internet-Drafts within the SIP working group and if found worthy, they further proceed into the RFC standards track. For instance, “Session Timers in the Session Initiation Protocol (SIP)” [17], is a proposed standard extension to the SIP that has received an RFC status. It defines a keep-alive mechanism for the sessions established by the SIP. Using this mechanism, a call stateful proxy can determine whether a session is still active or not. Many other proposals can be found in the home page of the SIP working group [32]. Despite of these extensions, the basic model and architecture defined by SIP is unlikely to be changed.

2.2 Review of H.323

Besides SIP, there are other signaling protocols that allow for the session establishment between interested parties. For the purpose of comparison with SIP, this section briefly introduces H.323 developed by ITU-T. Finally, a comparison of SIP and H.323 is drawn.

The **H.323** standard is specified by the International Telecommunication Union (ITU) Telecommunication Standardization Sector (ITU-T) Study Group 16 [37]. It specifies the components, protocols, and procedures that provide multimedia

communication services, e.g., real-time audio, video, and data communications, over packet networks, including IP-based networks. H.323 is part of a family of the ITU-T recommendations called H.32x that provides multimedia communication services over a variety of networks. It competes directly with SIP but takes telecommunication-oriented approach as opposed to SIP's Internet-oriented approach. The H.323 version 1 named "visual telephone systems and equipment for LANs that provide a non-guaranteed quality of service (QoS)" was accepted in 1996. In 1998, the version 2 of H.323 was specified due to the emergence of VoIP.

2.2.1 Comparison of SIP and H.323

H.323 and SIP have been compared in many researches. Some of the key comparisons are shown in table 3.

Table 3 Comparisons between SIP and H.323

	SIP	H.323	Implications
Encoding	Textual	Binary	Textual encoding typically results in higher bandwidth overhead, but is easy to extend, debug, and process by text-processing tools.
Call setup delay	1.5 RTT (round-trip time)	1.5 RTT or more	
Extensibility	Open to new protocol features.	Vendor specific	
Architecture	Modular, additional functions reside in separate orthogonal protocols.	Monolithic, all services provided by H.323 components.	Monolithic design makes component updates difficult and expensive.
Instant Messaging Support	Yes	No	
Addressing	Any URL, including E-mail address, H.323, http, E.164 URLs, ...	Host (without username), gatekeeper-resolved alias (arbitrary case-sensitive string, e.g., E-mail address), E.164 telephone numbers.	
Transport protocol	UDP and TCP, most implementations use UDP.	UDP and TCP, most implementations use TCP.	Usage of TCP results in higher call set-up time.

Compared with SIP, H.323 can ensure more reliable signaling message transport because it mainly uses TCP and UDP as the transport protocol. Another advantage of H.323 over SIP, at least from the operators or service providers' point of view, is that H.323 allows more centralized control. There are many signaling and media control elements in H.323, such as Multipoint Control Unit (MCU), gateways and gatekeepers. The operators or service providers mostly control these elements.

2.3 SIP Applications

In section 2.2, the SIP methods and the functions they enable have been discussed. Many applications and services can be implemented based on these functions. SIP supports multimedia communications over the IP networks. That is, the media can be audio, text, or video, and can be used to make a voice call, to hold a video conference, and to send a text notification message, etc. In the following section the most popular application of SIP will be introduced, that is the Voice over IP.

2.3.1 Voice over IP

Voice over IP (VoIP) is also known as telephony over IP network. It is a way to enable people talking over the Internet instead of the traditional telephony network. To do this, a VoIP system turns the analogue voice into digital data (IP data) and transmits them over IP networks. VoIP has been gaining more popularity, and two reasons may contribute to this popularity: one is the lower cost and another is the increased functionality.

A VoIP system consists of four basic components: user terminal, media gateway, call processing server, and IP network. A user terminal is the user's talking and listening device. It can be a PC installed software (so-called a softphone), a

normal phone connected to a PC through an analog-to-digital adaptor, or a specialized IP phone which has all the necessary hardware and software to handle the calls. The media gateway is mainly used to convert voice from analog to digital to create voice IP packets. There are several kinds of media gateways: residential gateways, trunk gateways, and access media gateways. Residential gateways provide a traditional analog interface to the VoIP network.

Examples of residential gateways include cable modem, xDSL modem and the previously mentioned analog-to-digital adaptor. Trunk gateways provide an interface between the traditional telephone network and the VoIP network. Access media gateways provide a traditional analog or digital private branch exchange (PBX) interface to a VoIP network. The call processing server mainly handles the signaling traffic of the VoIP network. The basic function of the signaling traffic is to establish and terminate a call. The signaling traffic also provides control over the call session. There are different protocols available to serve as the VoIP signaling protocol, H.323 and SIP being the two major choices. The fourth basic component of VoIP is the IP networks. The IP networks provide transmission and switching of the IP data - the digitalized human voice. In addition, to make a VoIP system work, many other supporting protocols are needed. For example, G.711 [38] is one of the standards used to digitalize the analog voice, code and decode the digitalized signal. The Real-time Transport Protocol (RTP) [56] is used to transport the audio stream between the talking parties after a conversation session is created (by using SIP or H.323).

2.3.2 QoS for VoIP

QoS stands for Quality of Service. Depending on where, how and why it is used, people see it in different angles and have different appreciations of it.

The most common definition we have of QoS is the differentiation between types of traffic and types of services so that the different types of service and traffic can be treated differently.

QoS is more in demand on corporate LANs, private networks and intranets (private networks interconnecting parts of organizations) than on the Internet and ISP networks. For example, you are likely to see QoS being deployed over a campus where students in dorm play half-life over the campus LAN, thereby congesting the network and hindering traffic for other more important types of data. QoS deployment in this case can favor traffic more the more important office data at the detriment of trivial network gaming, without however killing the latter. On the other hand, surfing the global Internet, there is most of the time no real QoS (unless the ISP has deployed QoS mechanisms). So, how quick one draws audio, text or video traffic generally depends on the bulk of the media. Text comes first, naturally. If an ISP provides QoS for favouring voice, voice reception would be great, and depending on the bandwidth, other media types might suffer.

QoS is an important tool for VoIP success. Through the years QoS mechanisms have become more and more sophisticated. Now, there can exist QoS mechanisms for small LANs up to giant networks.

In networking, quality can mean many things. In VoIP, quality simply means being able to listen and speak in a clear and continuous voice, without unwanted noise. Quality depends on the following factors:

- data loss
- consistent delay characteristics (called jitter)
- latency, leading to echo

ITU-T G.114 recommends a maximum of a 150 ms one-way latency.

3 Existing SIP-based Open Source Implementations

There are numerous SIP products available on the Internet either as free open source or as commercial products. Products listed below are partly or fully open source. Most of them aim at promoting SIP-based VoIP. In addition, the interoperability issue is also an important aspect of their motivations.

3.1 Asterisk Open Source PBX

Asterisk is a free PBX, developed and sponsored by Digium [15], Asterisk provides seamless integration between analog Time Division Multiplexing (TDM) and VoIP based networks in a single PBX. It also enables full interactive voice response (IVR) functionality through any scripting language available on Linux.

3.2 X-Lite

X-Lite [48] is a free SIP soft phone with many PBX-like features. X-Lite based on open standards, such as the SIP stack, which enhance its network interoperation and integration. X-Lite is made by CounterPath Solutions Inc. [12], a Canadian soft phone manufacturer. CounterPath claims that now X-Lite has hundreds of thousands of installations around the world.

In addition to X-Lite, CounterPath has also provided the fully featured X-PRO commercial soft phone, and eyeBeam, a SIP video over IP soft phone with IM and Presence [43].

3.3 SIP Express Router (SER Project)

SIP Express Router (SER) is a SIP server licensed under the GNU General Public License. It can be configured to act as SIP registrar, proxy or redirect server. SER features presence support, RADIUS/syslog accounting and

authorization, XML-RPC-based remote control and others. Web-based user provisioning, serweb, is available.

SER's performance allows it to deal with operational burdens, such as broken network components, attacks, power-up reboots and a rapidly growing user population. SER can be configured for many scenarios including small-office use, enterprise PBX replacements and carrier services.

SER was initially developed by Fraunhofer FOKUS in 2001. Part of the Fokus team then moved, with the SER copyrights, to a newly created company, iptelorg in 2004. Two of the five SER core developers and one main contributor started a new Open Source project named OpenSER. Additional open source code was contributed by independent third parties.

SER and OpenSER followed different development paths.

SER 2.0 Release Candidate 1(Ottendorf) was made available May 12, 2007 and is currently undergoing heavy development.

The SER project remains open source.

in 2005 the company IPtel.org was bought by TEKELEC, and is responsible for the TEKELEC session router and CSCF.

3.4 Brekeke SIP Server and OnDo PBX

The Brekeke SIP Server is a product of Brekeke Software, Inc [9]. It is a call control server compliant with SIP. The server provides functions of a Registrar, Proxy Server, and Call Routing. With Brekeke SIP Server, one can use SIP hard phones, SIP soft phones, and SIP-PSTN Gateways for VoIP communications.
Brekeke SIP Server is ready for any office environment. It supports sophisticated Network Address Translator (NAT) [19] traversal features. Therefore, it serves various kinds of network environments. By combining VPN

(Virtual Private Network), Brekeke SIP Server enables customers to build a secure VoIP system.
Personal and educational use of Brekeke SIP Server is free.

3.7 SIP Application Server

SIP Application Server by Ubiquity [59] is a deployment platform and Application Creation Environment (ACE) aimed at service providers for testing and developing SIP applications. It provides testing features, such as proxy, call setup and tear down. However, free download of source codes is not available.
Ubiquity, however, provides a free User Agent (UA) and a free SIP Test Messenger tool. The UA acts as a soft phone allowing the user to control IP telephony calls on screen. The UA can also be used as a test tool. For example, it helps developers of SIP applications to benchmark their products by using the known parameters of the UA. The SIP Test Messenger is Java software that can be used to test the sending text message and listening to the response between it and the user's SIP implementation. According to Ubiquity, this tool is especially useful for stress testing products with scenarios that are otherwise difficult to reproduce.

3.9 SIPfoundry

SIPfoundry [60] is a not-for-profit open source community, whose mission is to promote and advance Session Initiation Protocol (SIP) - related open source projects. Through SIPfoundry, the users, developers, and distributors of SIP-based products can collectively support each other and accelerate the growth and adoption of SIP.The projects include:
• sipX, a family of projects consists of a SIP soft phone sipXphone and a full PBX solution sipXpbx.

• reSIProcate, a project consists of a stack and a small collection of applications ideally suited to implement SIP applications, such as, soft phones, proxies and IM/Presence servers or clients, etc.
• OSPclient, a complete software development toolkit for developers.

3.10 Chosen Test Bed for SIP Services

The presented software were evaluated taking into account availability, implemented features, open interface with a lot of programming languages, interoperability, suitability for its intended use in Laboratory, and maintainability. Based on the evaluation results, Asterisk was selected as the SIP proof-of-concept test-bed system installed in the test laboratory in order to test the Voicemail to e-mail and click to call services. The reasons of choosing Asterisk among other SIP products include:
• Asterisk is an open source with a rich set of features.
• Asterisk is used for commercial purposes from some ISPs offering VoIP services
• Asterisk is the only open source product that has open interfaces with a lot of programming languages and scripting languages.
• Asterisk is the only open source product that can interact with a lot of servers – services running on a UNIX machine e.g. sendmail server and Apache web server.
• Asterisk has different distributions with documentation.
• Asterisk has an active mailing list; both Asterisk developers and Asterisk users are actively participating.
• Asterisk support various (both free and commercial) SIP user agents, such as Cisco 7960, X-Lite, etc.

Asterisk and its use are presented in more detail in the following chapter.

4 Asterisk PBX

Asterisk is a Private Branch Exchange (PBX). A PBX can be thought of as a private phone switchboard, connecting to one or more telephones on one side, and usually connecting to one or more telephone lines on the other. This is usually more cost effective than leasing a telephone line for each telephone needed in a business.

4.1 Asterisk Features

Asterisk-based telephony solutions offer a rich and flexible feature set. Asterisk offers both classical PBX functionality and advanced features, and interoperates with traditional standards-based telephony systems and Voice over IP systems. Asterisk offers the advanced features that are often associated with large, high end (and high cost) proprietary PBXs. It is based on the latest SIP standard.

Furthermore, it acts as a voicemail and application server supporting voicemail services and IVR services. Asterisk runs on Solaris,CentOs, Linux, and BSD platforms and supports IPv4.

Asterisk can be characterized as a successful SIP implementation used in both commercial and academic environment. This conclusion is based on its wide range of performance testing results.

• *cost effectiveness* - Asterisk can handle calls on relatively low-cost platforms, enabling setting up inexpensive and easy-to-manage networks.

• *high configurability* - Asterisk provides a lot of configuration files, and some distributions offer web user interface in order to make configuration easier. It also provides an open interface so that it can interact with a lot of scripting languages.

• *portability* - Asterisk is implemented in ANSI C, making it easily portable to various platforms. It has been extensively tested on Linux, Solaris and CentOS.

• *interoperability* - Implementation is based on the SIP standard, making Asterisk highly interoperable.

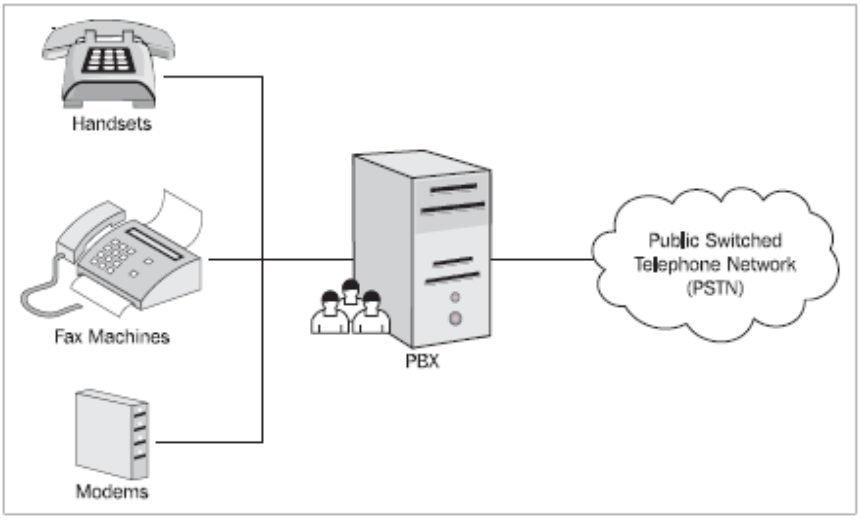
• *small size* - Asterisk is very small.

4.2 Asterisk Scenarios

Asterisk is a highly adaptable and applicable telephony platform. Therefore nearly unlimited possible applications and different scenarios exist. Some of the most commonly usage -scenarios are introduced.

4.2.1 Station-To-Station Calls

Asterisk offers station-to-station calls. This means that users can dial from one phone to another phone. While this seems obvious, elementary phone systems are available (often referred to as Key Systems) that support multiple phones and multiple lines, and allow each phone to use any line. In operation, the handsets do not have individual extensions that can be dialed, and so there is no way to initiate a call from one handset to another. These systems can usually be identified by having all outgoing lines on every telephone, usually with a blinking light. Unlike Key Systems, Asterisk allows for station-to-station calls, allowing directed internal communications.



4.2.2 Telco Features

Asterisk supports all of the "standard" features we would expect from any telephone company (or telco). Asterisk supports sending and receiving Caller ID, and even allows us to route calls based on the Caller ID. Using Caller ID with the PSTN requires us to subscribe to that feature with our PSTN connection provider.

Asterisk also supports other features as expected, such as call waiting, call return (*69), distinctive ring, transferring calls, call forwarding, and so on. These basic features and more are provided by Asterisk.

4.2.3 Asterisk as a Voice over IP (VoIP) System

Asterisk gives us the ability to use Internet Protocol (IP) for phone calls, in tandem with more traditional telephone technologies.

Choosing to use Asterisk does not mean that we can only use Voice over IP for calls. In fact, many installations of Asterisk do not even use it at all. But each of

those systems has the ability to add Voice over IP easily, at any time, with no additional cost.

Most companies have two networks: one for telephones, and one for computers. What if we could merge these two networks? What would the savings be? The biggest savings are realized by reducing the administrative burden for Information Technology staff. We can now have a few experts on computing and networking, and since telephony will run on top of a computer and over our IP network, the same core knowledge will empower our staff to handle the phone system.

We will also realize benefits from decreased equipment purchasing in the long run. Computer equipment gets progressively cheaper while proprietary phone systems seem to remain nearly constant in price. Therefore, we may expect the costs for network switches, routers, and other data network equipment to continue to decrease in price.

In most current phone systems, extensions can only be as far away as the maximum cabling length permitted by the telephone system manufacturer. While this seems perfectly reasonable, sometimes we would like it not to be so. When using VoIP we can have multiple users using the same Asterisk service from a variety of locations. We can have users in the local office using PSTN phones or IP phones, we can have remote VoIP users, we can even have entire Asterisk systems operated and run completely separately but with integrated routing.

One way to slash overhead is to reduce the amount of office space required. Many businesses use telecommuting for this purpose. This often creates a problem: which number do we use to reach a telecommuter? Imagine the flexibility if telecommuting employees could simply use the same extension when at home as when in the office or even when using their mobile!

Voice over IP allows us to have an extension anywhere we have a reasonably fast Internet connection. This means employees can have an extension on the phone system at home if they have a broadband connection. Therefore, they will have access to all of the services provided in the office, such as voicemail, long-distance calling, and dialing other employees by extension.

Just as we can bring employees into the PBX from their homes, we can do the same for remote offices. In this way, employees at multiple locations can have consistent features, accessed in exactly the same way, helping to ease the burden of training employees.

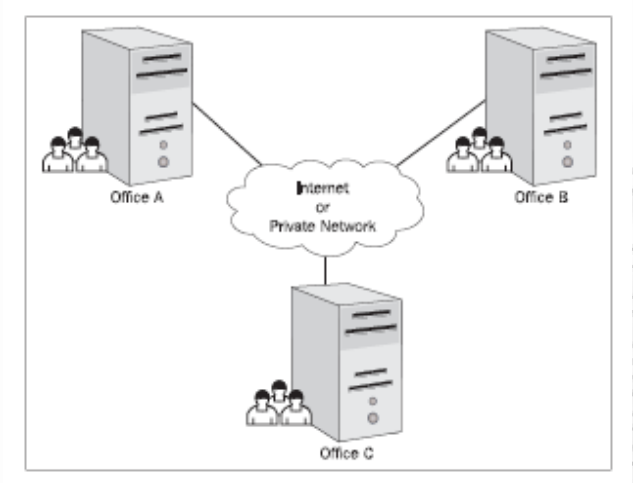


Figure 5 Asterisk scenarios

4.3 Asterisk Architecture

Asterisk is carefully designed for maximum flexibility. Specific APIs are defined around a central PBX core system. This advanced core handles the internal interconnection of the PBX, cleanly abstracted from the specific protocols, codecs, and hardware interfaces from the telephony applications. This allows Asterisk to use any suitable hardware and technology available now or in the future to perform its essential functions, connecting hardware and applications.
Items handled by core internally :

PBX Switching

The essence of Asterisk, of course, is a Private Branch Exchange Switching system, connecting calls together between various users and automated tasks. The Switching Core transparently connects callers arriving on various hardware and software interfaces.

Application Launcher

Launches applications which perform services for uses, such as voicemail, file playback, and directory listing.

Codec Translator

Uses codec modules for the encoding and decoding of various audio compression formats used in the telephony industry. A number of codecs are available to suit diverse needs and arrive at the best balance between audio quality and bandwidth usage.

Scheduler and I/O Manager

Handles low-level task scheduling and system management for optimal performance under all load conditions.

Loadable module APIs

Four APIs are defined for loadable modules, facilitating hardware and protocol abstraction. Using this loadable module system, the Asterisk core does not have to worry about details of how a caller is connecting, what codecs are in use, etc.

Channel API

The channel API handles the type of connection a caller is arriving on, be it a VoIP connection, ISDN, PRI, Robbed bit signaling, or some other technology.

Dynamic modules are loaded to handle the lower layer details of these connections.

Application API

The application API allows for various task modules to be run to perform various functions. Conferencing, Paging, Directory Listing. Voicemail, In-line data transmission, and any other task which a PBX system might perform now or in the future are handled by these separate modules.

Codec Translator API

Loads codec modules to support various audio encoding and decoding formats such as GSM, Mu-Law, A-law, and even MP3.

File Format API

Handles the reading and writing of various file formats for the storage of data in the filesystem.

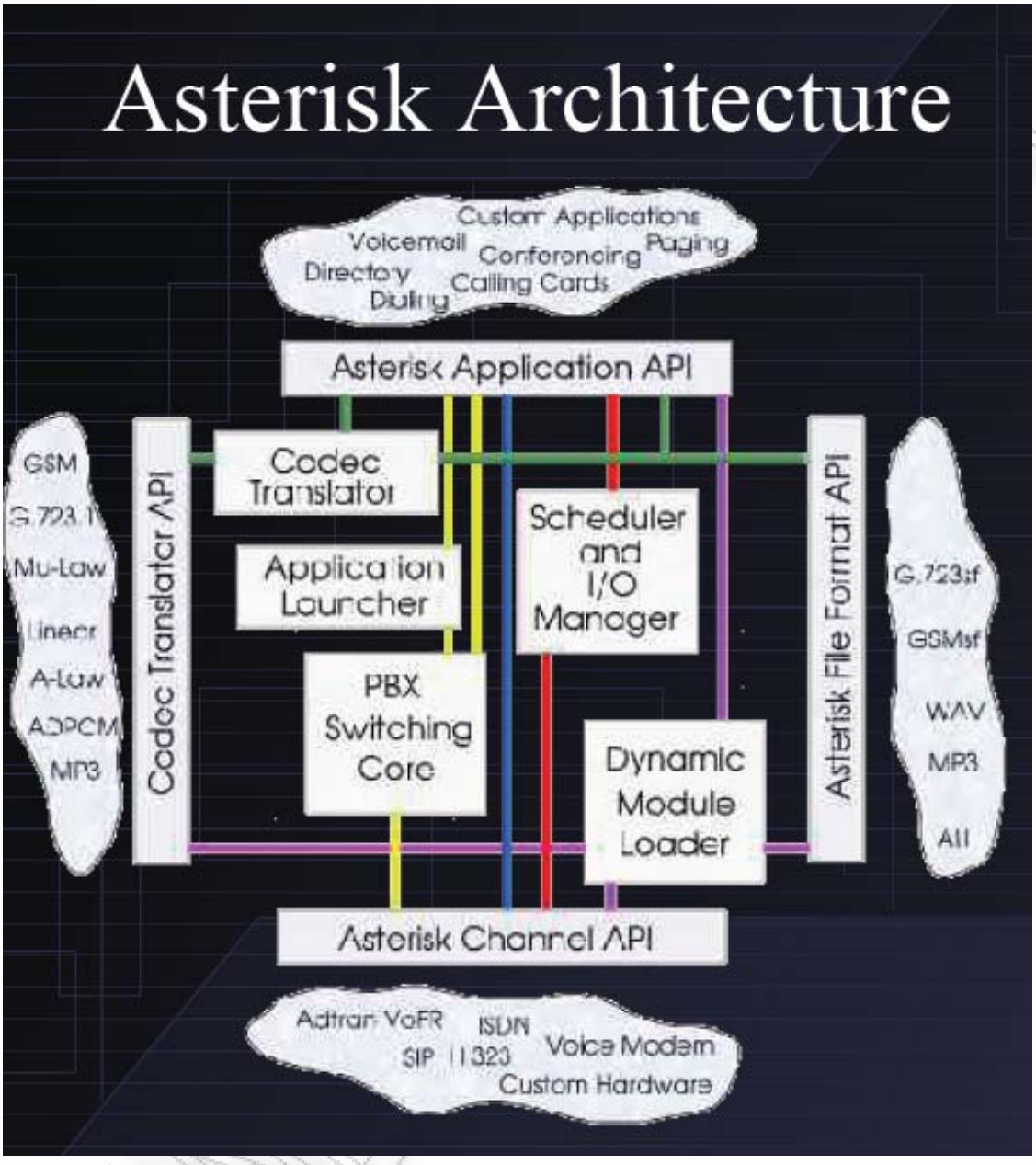


Figure 6 General Architecture of Asterisk

5 SIP applications using Asterisk

The main purpose of this thesis is to pilot a click-to-call and a voicemail to e-mail service. The environment for the pilot is my Laboratory. The reasons why Asterisk is chosen for the experimentation have been explained in Chapter 3.
This chapter will introduce the Asterisk system's and Laboratory's requirements as well as the test environment. The tools and methods used to carry out the experiment will also be briefly introduced. The tested services will be defined based on some scenarios. The captured traffic corresponding to the scenarios will be analyzed and the achieved results will be reported in the end of the chapter.

Asterisk-based telephony solutions offer a rich and flexible feature set. Asterisk offers both classical PBX functionality and advanced features, and interoperates with traditional standards-based telephony systems and Voice over IP systems. Asterisk offers the advanced features that are often associated with large, high end (and high cost) proprietary PBXs.

Some of the call features offered by Asterisk are : Music On Hold, Call Detail Records, Call Forward on Busy, Call Forward on No Answer, Call Forward Variable, Call Monitoring, Call Parking, Call Queuing, Call Recording, Call Retrieval, Call Routing (DID & ANI), Call Snooping, Call Transfer and Call Waiting.

The codecs supported by Asterisk are : ADPCM, G.711 (A-Law & μ-Law), G.722, G.723.1, G.726, G.729, GSM, iLBC, Linear, LPC-10 and Speex.

Any developer can extend Asterisk by working with the C API or by using AGIs, which are analogous to CGI scripts. The Asterisk Gateway Interface, or AGI, provides a standard interface by which external programs may control the Asterisk dialplan. Usually, AGI scripts are used to do advanced logic, communicate with relational databases (such as PostgreSQL or MySQL), and

access other external resources. Turning over control of the dialplan to an external AGI script enables Asterisk to easily perform tasks that would otherwise be difficult or impossible. Perl, PHP, and Python are the most commonly languages used for AGI programming.

5.1 Asterisk System Requirements

5.1.1 Software Requirements

There are several operating system platforms in use at TML and it is essential that the selected software can be used in all of them. Asterisk is a good choice, because it supports various platforms. Table 4 shows Asterisk's general software requirements and the existing options.

Table 4 Asterisk Software Requirements and Options

Requirements	Options
OS (operating system)	Linux/i386(RedHat, Fedora, Mandrake etc), Linux/armv4l, FreeBSD/i386, OpenBSD/i386, Solaris/sparc64, NetBSD/sparc64
Compilation	gcc or icc, bison or yacc, flex, GNU make, sed and tr
Installation	GNU tar and gzip, GNUor BSD install
Database	MySQL, libmysqlclient and libz
For web support	Apache web server, PHP, MySQL-PHP
For optional modules	Sendmail server

5.1.2 Hardware Requirements

In general, Asterisk runs on a PC with quite modest resources. As a general guideline, considering only voice calls, a few factors have to be taken into account:

- A PC with a 100Mbps network connection and reasonable resources should be adequate, particularly for trials.
- In some exceptional cases, e.g., universities or a medium scale company, a memory size with at least 512 megabytes is suggested and a gigabyte memory should be better.
- The additional functionality of click-to-call and voicemail to e-mail services, would add a load on the machines due to the servers running on the machine but there is not any need to be taken into account.

5.2 Test Environment and Features to Be Tested

Asterisk was installed in the test laboratory as a proof-of-concept test-bed system in order to test the Voicemail to e-mail and click to call services. The verification of the system was planned to expand to a larger scope, i.e., internet, if the tested services were proved applicable in the real world. This section presents the test environment and the tested features.

5.2.1 Test Environment

The test environment in the Laboratory, as shown in Figure 18, incorporates three computers, one server machine (Linux) and two SIP client machines (Windows XP), and one PDA Nokia N95 connected to the LAN via WiFi. A linksys wireless router is used. The server machine is also accessible from the

Internet. The router is connected to the internet through a broadband internet connection (1Mbit). The router is configured with QoS (best effort) to the SIP messages. UDP or TCP messages that contain SIP messages are using the port 5060. Also the firewall of the router is configured with port 5060 open to listen to connections and the also range of the ports 10000 – 20000 open to traffic from outside the LAN. The requests to these ports are routed to the Asterisk server with the IP address 192.168.1.101 by using port forwarding properly configured in the router's settings, Any SIP client can connect through the internet to the server machine and can make SIP calls with the proper configuration.

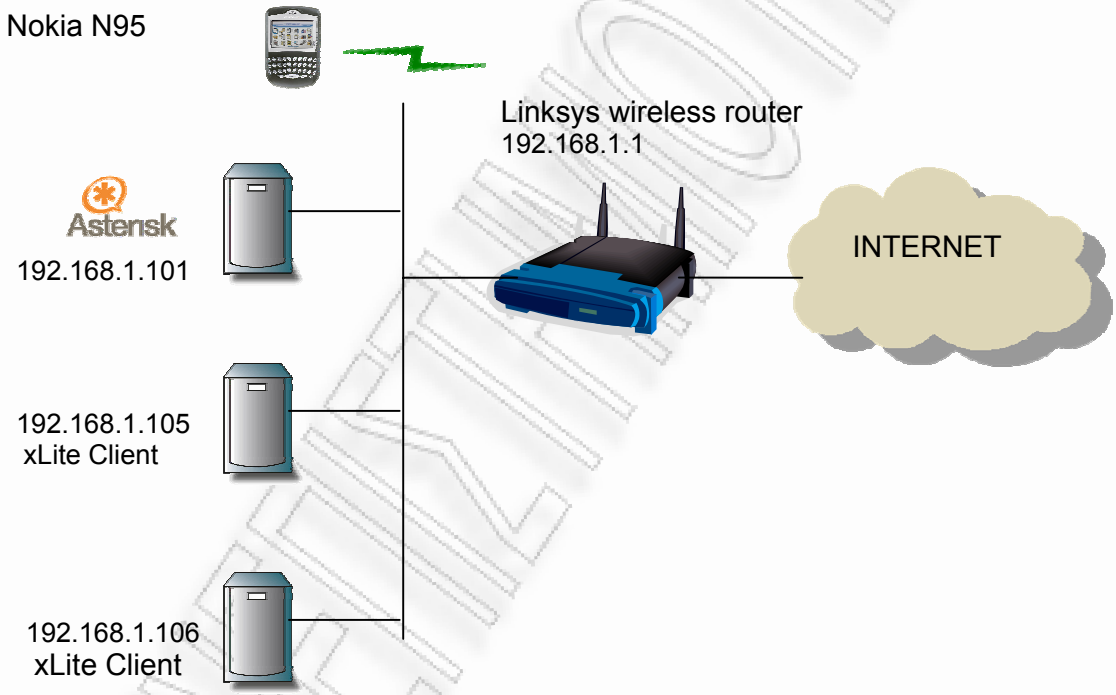


Figure 7 the Asterisk test environment in Laboratory

The configurations of these machines are as follows:

Server machine is running CentOS as an operating system and Mysql, Apache, PHP, sendmail, MyfreePBX open source web interface for the management of

Asterisk PBX. Also WebMin open source interface for the management of Unix systems is used.

Client machines are running Microsoft Windows XP and Mozilla or Internet Explorer as Internet Browser.
The machines have variable processor speeds ranging from 2.4MHz to 3.2MHz. Their memories vary from 512MB to 1GB RAM and all are equipped with a 120GB hard disk.

5.2.2 Testing Tools and Methods

Ethereal is used to capture and analyze the SIP traffic, i.e., messages between the server and the three Windows Messenger clients. It has been proved to be a powerful network traffic diagnosis tool.
FreePBX is an open source web-based user interface that displays operational statistics and running processes and can manage all the operations and features of Asterisk.
WebMin is also an open source web-based interface which is used to manage a UNIX system. It displays information about all the services – servers running on the machine and an easy way to configure them.
Therefore, it provides a convenient way of monitoring the health of the server.
Based on the features under testing, test cases have been deployed to define the test scenarios as well as the corresponding traffic. These are explained in more detail in the following section.
Message flows are used to clarify and analyze the captured traffic.

5.2.3 Features and Services to Be Tested

We selected the following scenarios to be tested and analyzed.

1. Click-to-Dial service
2. Voicemail to e-mail service
3. calling features :
    - Call Forward Unconditional
    - Call Forward Busy

These scenarios are demonstrated in the level of message flows in section 5.4, in an effort to provide in depth understanding of SIP working logic at the package level.

5.3 Voicemail to e-mail Service

5.3.1 Scenario Message Flow and Analysis

In the test, two clients were created and added to the database: sip:2001@192.168.1.101, sip:2008@myfreepbx.dyndns. The SIP user 2001 has the IP address 192.168.1.105. The user 2008 is connected via WiFi to the LAN and has enabled the voicemail feature in case he is unavailable. He also has an e-mail address assigned to his voicemail service in order his voicemail messages to be e-mailed to this address. The asterisk server has the IP address 192.168.1.101.

User 2001 has registered to Asterisk PBX , while 2008 is currently unavailable because of not having registered. Figure 21 presents the corresponding message flow of scenario 1.

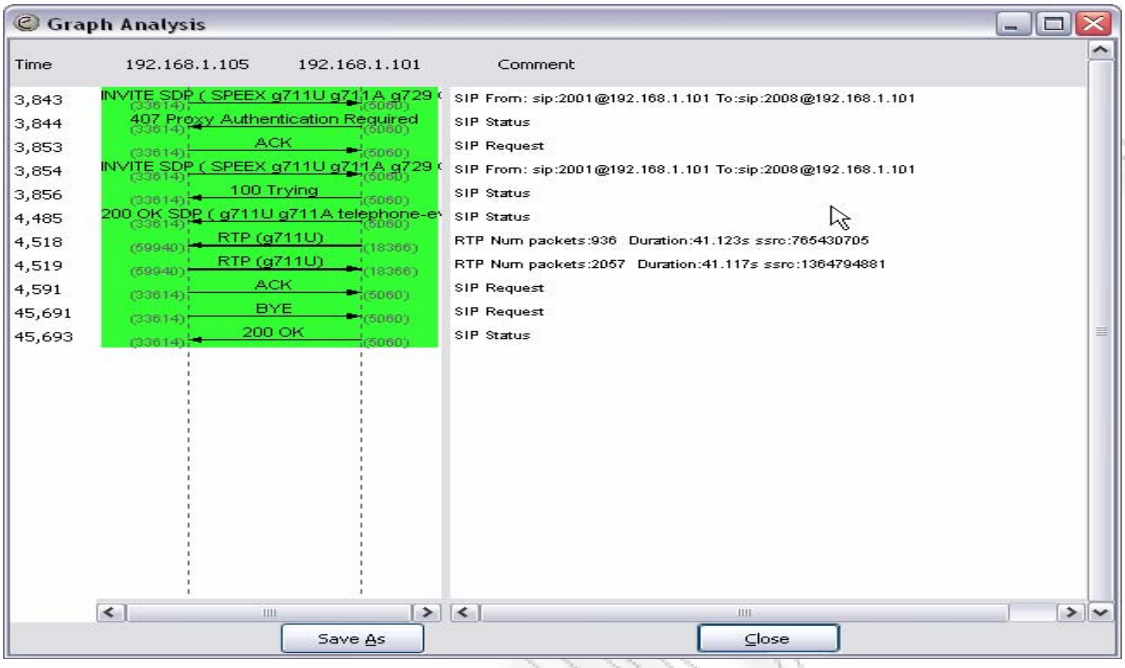


Figure 8 Voicemail service Call Flow

Here it is easy to see Asterisk acting as a voicemail server.

- User 2001 (192.168.1.105) initiates a call to User 2008. He sends a SIP : INVITE message.
- User 2008 is unavailable so the phone does not ring. User 2008 has voicemail service enabled in this case.
- Asterisk answers the call by sending a SIP : 200 OK message with the rtp data of the voicemail server.
- So the call is established and the User 2001 leaves a message to the voicemailbox of User 2008.
- User 2008 has assigned the suncity13@gmail.com address to his voicemail service. So the voicemail message is send via e-mail to his e-mail account.

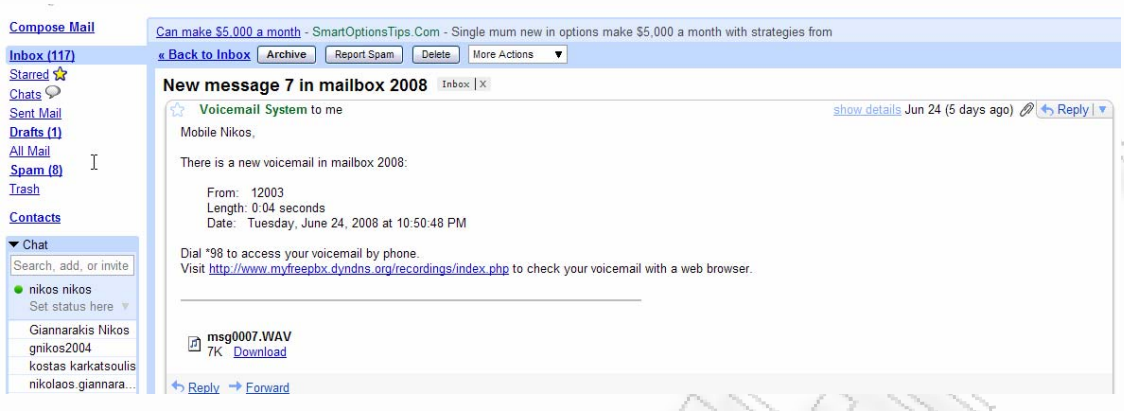


Figure 9 Voicemail to e-mail service.

E-mail with voice message attached as wav file attached delivered to e-mail address.

The voicemail.conf file contains the following line :

2008=>123456,Mobile Nikos,suncity13@gmail.com,suncity13@gmail.com,attach=yes|saycid=yes|envelope=yes|delete=no

By this way to the user 2008 the e-mail address suncity13@gmail.com is assigned.

5.4 Click – to – Dial Service

This service is implemented using php scripts. The php scripts interact with Asterisk using the PHP AGI (Asterisk Gateway Interface).

PHPAGI is a PHP class for the Asterisk Gateway Interface. The package is available for use and distribution under the terms of the GNU Public License

By using this class it is possible to originate a call. The script that performs this action is call_both.php.

5.4.1 Scenario and Analysis

The user 2001 browses his personal address book. The IP address of his personal address book is : http://myfreepbx.dyndns.org/click-to-dial.

User 2001 can add his contacts to his address book. The URL for adding users to the address book is : http://www.myfreepbx.dyndns.org/click-to-dial/admin.php. This URL and the form for adding, editing and deleting contacts can be shown in the figure 10.

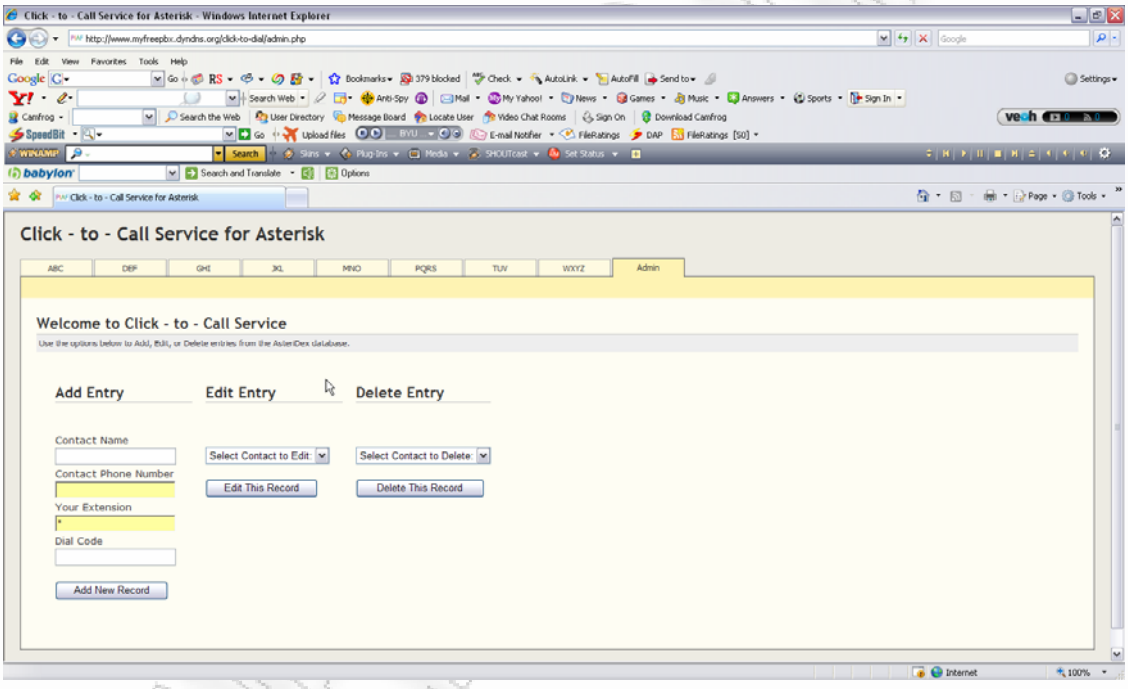


Figure 10 Adding contacts to address book for click-to-dial service.

The mysql commands that were used to create the Database and the tables for the address book are :

```
CREATE DATABASE `address_book`;
USE address_book;
```

```
CREATE TABLE `contact` (
  `id` mediumint(9) NOT NULL auto_increment,
  `name` varchar(40) NOT NULL default '',
  `in` varchar(40) NOT NULL default '*',
  `out` varchar(40) NOT NULL default '',
  `dialcode` varchar(5) NOT NULL default '',
  PRIMARY KEY  (`id`),
  KEY `name` (`name`,`in`,`out`)
) TYPE=MyISAM
```

The User 2001 (Nikos) wants to make a call to the User 2008 (Mobile Nikos). User 2001 finds the User in the address book and clicks on his name.

After that a pop up window is opened with the message : *Extension SIP 2001 is ringing now. When 2001 answers, call to 2008 will be placed.*

Then the User's 2001 phone is ringing (figure 11).

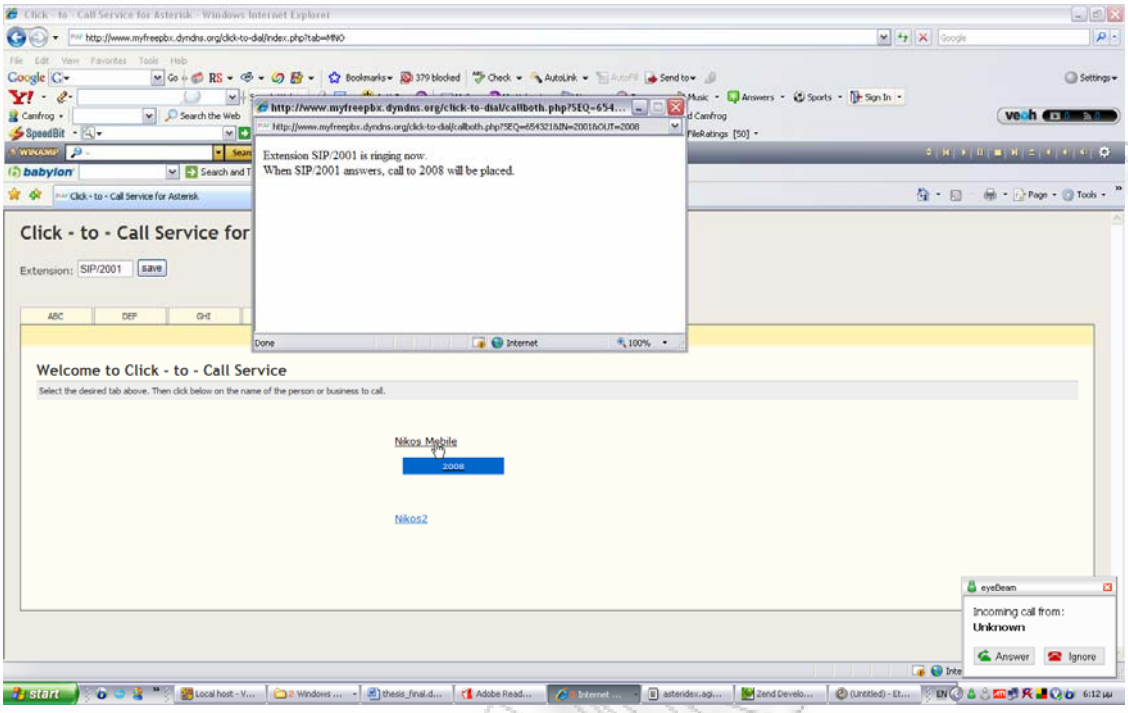


Figure 11 Call a contact from address book.

By clicking on the desired name in the address book, an http request is sent to the web server with some specific variables posted. The web server is running on the same machine with the Asterisk server. The http request can be shown in figure 12. The method of the request is GET and the variables that are posted are the extensions 2001 of the user that uses the service and the extension 2008 of the user that being called from address book.

The Asterisk server receives these variables and originates a call to User 2001. When user 2001 answers, Asterisk server originates a call to User 2008. Then Asterisk connects User 2001 and User 2008 and the Users can talk.

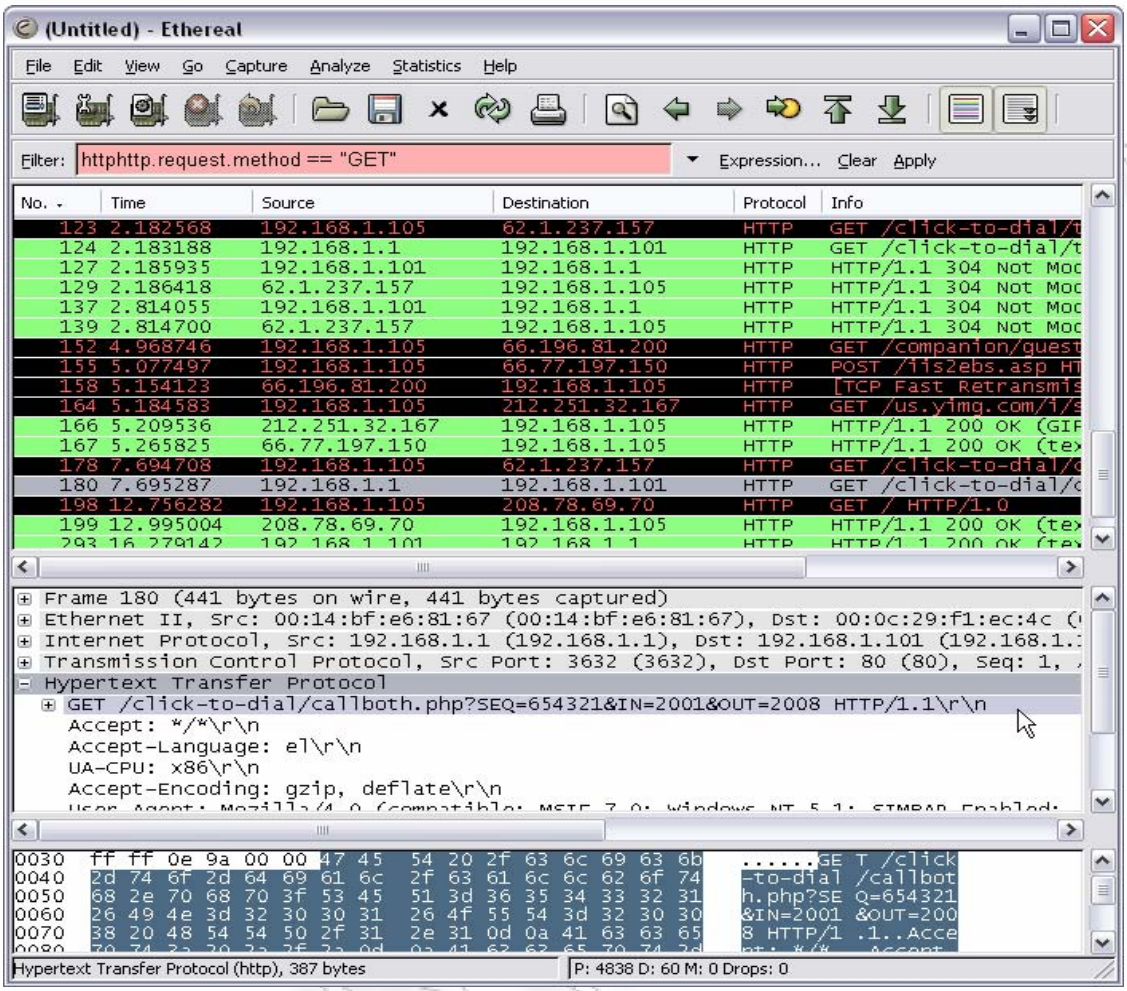


Figure 12 Method GET of http request.

The php scipt that makes the described actions is call_both.php.

Call_both.php

```
<?php


// $INdefault defines the default trunk and extension for incoming calls when the IN variable is set to an asterisk.
// $INtrunk defines which trunk you wish to use to process incoming calls. The usual options are SIP and IAX2.
```

```
// $CallerID defines the caller id to be sent with the outbound call.

$INdefault = "sip/2001" ;
//$INdefault = "local/222@from-internal" ;
$INtrunk="SIP" ;
$LDprefix="1" ;
$CallerID="6781234567" ;



$IN=$_REQUEST['IN'];
$OUT=$_REQUEST['OUT'];
$SEQ=$_REQUEST['SEQ'];
if ($IN < "1" and $IN<>"*") :
  exit() ;
endif ;
if ($IN<>"*") :
 $IN = $INtrunk . "/" . $IN ;
else  :
 $IN = $INdefault ;
endif ;
$OUT= $LDprefix . $OUT ;

$IN = str_replace( chr(13), "", $IN );
$IN = str_replace( chr(10), "", $IN );
$IN = str_replace( ">", "", $IN );
$IN = str_replace( "<", "", $IN );
$OUT = str_replace( chr(13), "", $OUT );
$OUT = str_replace( chr(10), "", $OUT );
$OUT = str_replace( ">", "", $OUT );
$OUT = str_replace( "<", "", $OUT );


$pos = false ;
if (strlen($OUT)>100) :
 $pos=true ;
endif ;
if (strlen($IN)>100) :
 $pos=true ;
```

```
endif ;
if ($SEQ<>"654321") :
 $pos=true ;
endif ;

if ($pos===false) :
$errno=0 ;
$errstr=0 ;
$fp = fsockopen ("localhost", 5038, &$errno, &$errstr, 20);
 if (!$fp) {
  echo "$errstr ($errno)<br>\n";
} else {
 fputs ($fp, "Action: login\r\n");
 fputs ($fp, "Username: phpagi\r\n");
 fputs ($fp, "Secret: phpagi\r\n");
 fputs ($fp, "Events: off\r\n\r\n");
sleep(1) ;
 fputs ($fp, "Action: Originate\r\n");
 fputs ($fp, "Channel: $IN\r\n");
 fputs ($fp, "Context: custom-callboth\r\n");
 fputs ($fp, "Exten: $OUT\r\n");
 fputs ($fp, "Priority: 1\r\n\r\n");
 fputs ($fp, "Callerid: $CallerID\r\n\r\n");
 fputs ($fp, "Timeout: 30\r\n\r\n");
sleep(2) ;
 fclose ($fp);
 }

echo "<HTML><HEAD>\n" ;
echo "<script>\n";
echo "var duration = 4000;\n" ;
echo "x = null;\n";
echo "function closeIt(){\n";
echo "x = setTimeout(\"self.close()\",duration);\n";
echo "}\n";
echo "</script>\n";
echo "</HEAD><BODY onload=\"closeIt();self.focus()\">\n" ;
```

```
echo "Extension $IN is ringing now. <BR>When $IN answers, call to $OUT
will be placed.\n" ;
echo "</BODY></HTML>\n" ;

else :
 echo "<HTML><HEAD>\n" ;
 echo "<script>\n";
 echo "var duration = 1000;\n" ;
 echo "x = null;\n";
 echo "function closeIt(){\n";
 echo "x = setTimeout(\"self.close()\",duration);\n";
 echo "}\n";
 echo "</script>\n";
 echo "</HEAD><BODY onload=\"closeIt();self.focus()\">\n" ;
 echo "</BODY></HTML>\n" ;
endif;
?>
```

5.4.2 Message Call Flow

The SIP Call Flow of the service can shown in figure 13.

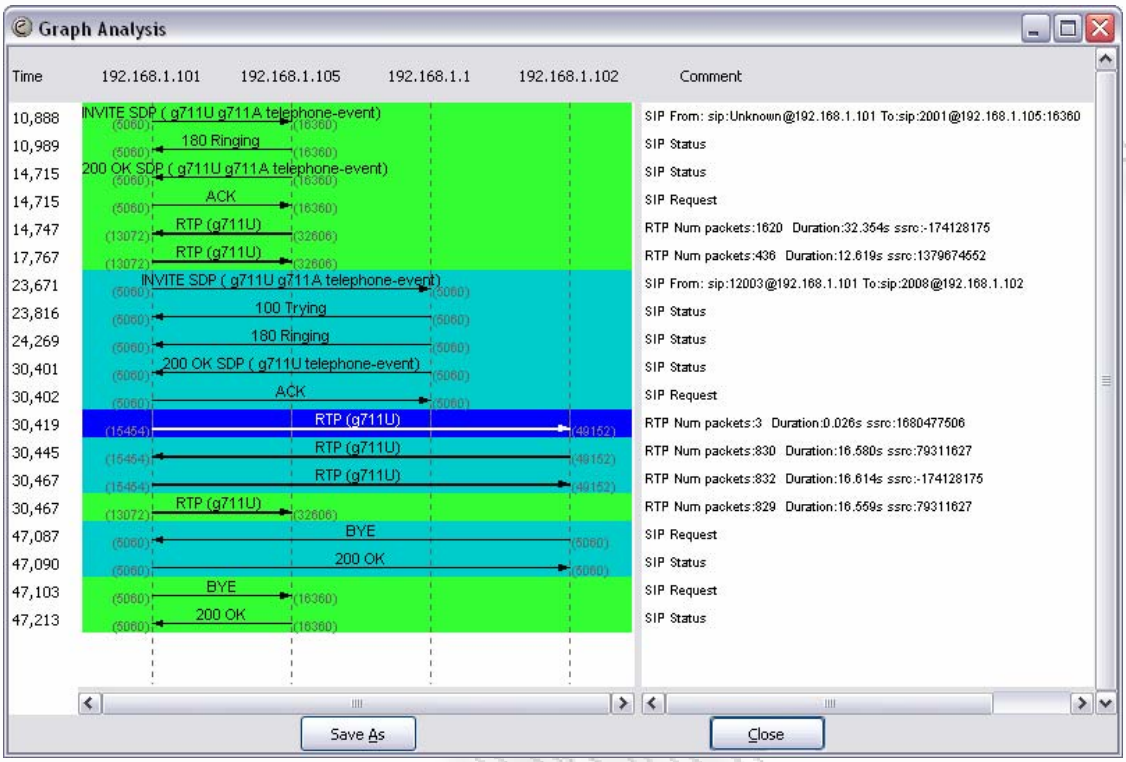


Figure 13 Sip Call Flow of Click – to – Dial Service.

- Asterisk server (192.168.1.101) sends a SIP : INVITE message to User 2001 (192.168.1.105) in order to initiate a call to that User.
- User 2001 answers with a SIP: 200 OK message sending his SDP data and the call is established.
- After that Asterisk server sends a SIP: INVITE message to User 2008 (192.168.1.102) and
- when User 2008 answers with a SIP : 200 OK message a rtp path is established between User 2001 and User 2008.
- The users can now talk.

5.4.3 Quality Of Service (QoS)

The Quality of Service of the Click – to – Dial service can be shown in figure 14. Figure 27 represents the rtp stream analysis for the call described above. The offered QoS for Lan Ethernet and Wireless clients is good enough. Studies has shown that Jitter between the starting and the final point of the communication should be less than 100 ms.

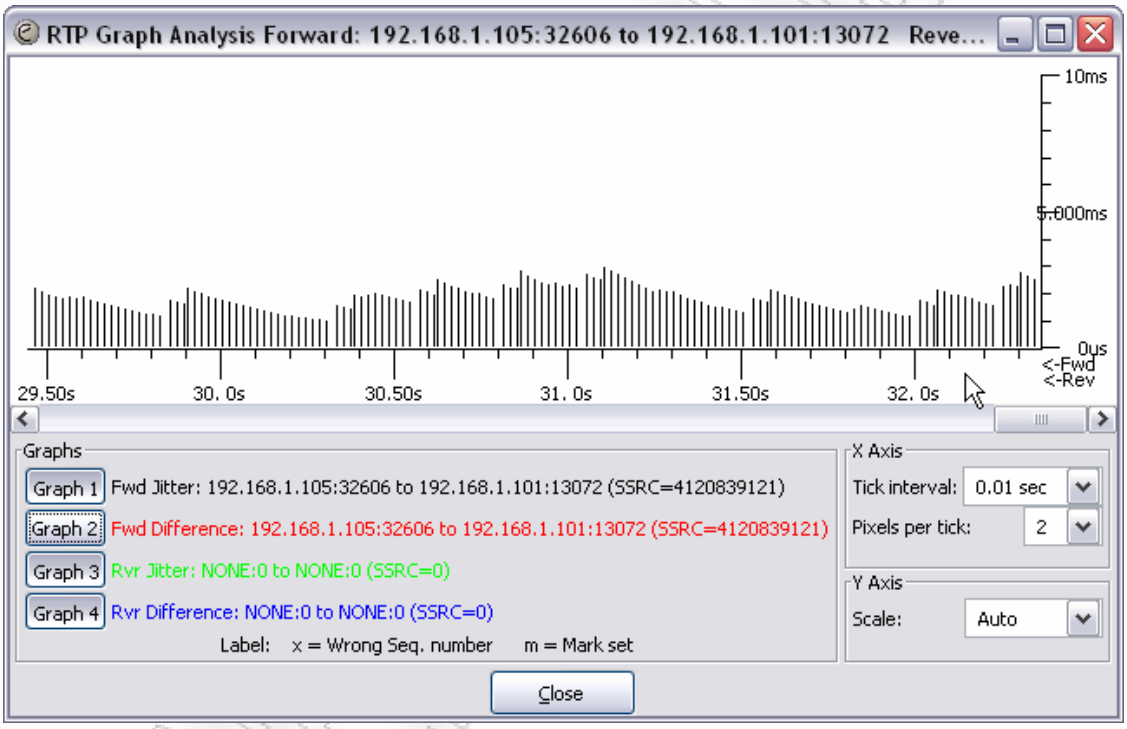


Figure 14 QoS analysis for click – to –call Service.

5.5 Calling Features

Call forwarding (or call diverting), in telephony, is a feature on some telephone networks that allows an incoming call to a called party, which would be otherwise unavailable, to be redirected to a mobile telephone or other telephone number where the desired called party is situated.

5.5.1 Unconditional Call Forwarding

Unconditional call forwarding redirects all incoming calls to the chosen number.

5.5.1.1 Scenario and Analysis

Call forward can be configured by each user. In our scenario User 2008 wants to enable call forward unconditional. User 2008 can configure this feature using FreePBX web interface. He signs in to his private configuration and voice mailbox page by using his voicemail Mailbox and Password (figure 15).

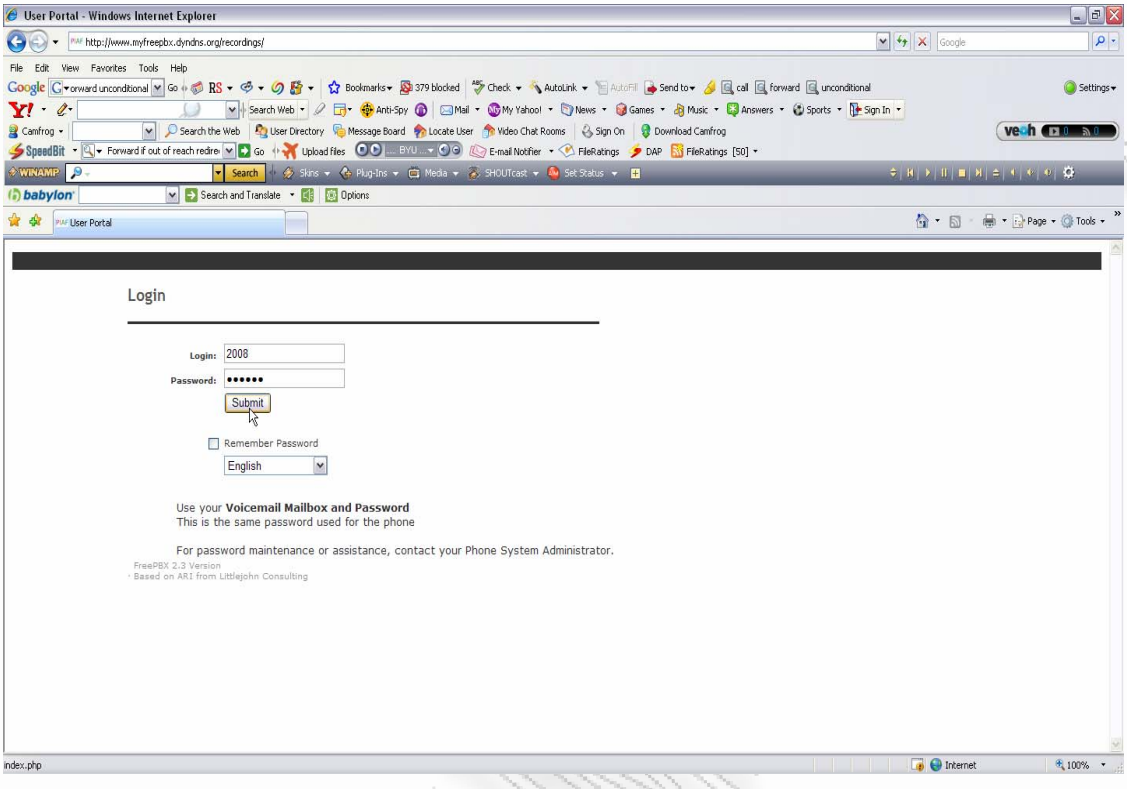


Figure 15 Signing in Voicemail Mailbox.

User 2008 can activate call forward feature by clicking Phone Features link. In the field unconditional the new extension, the number where the call will be forwarded, is entered by the User (figure 16).

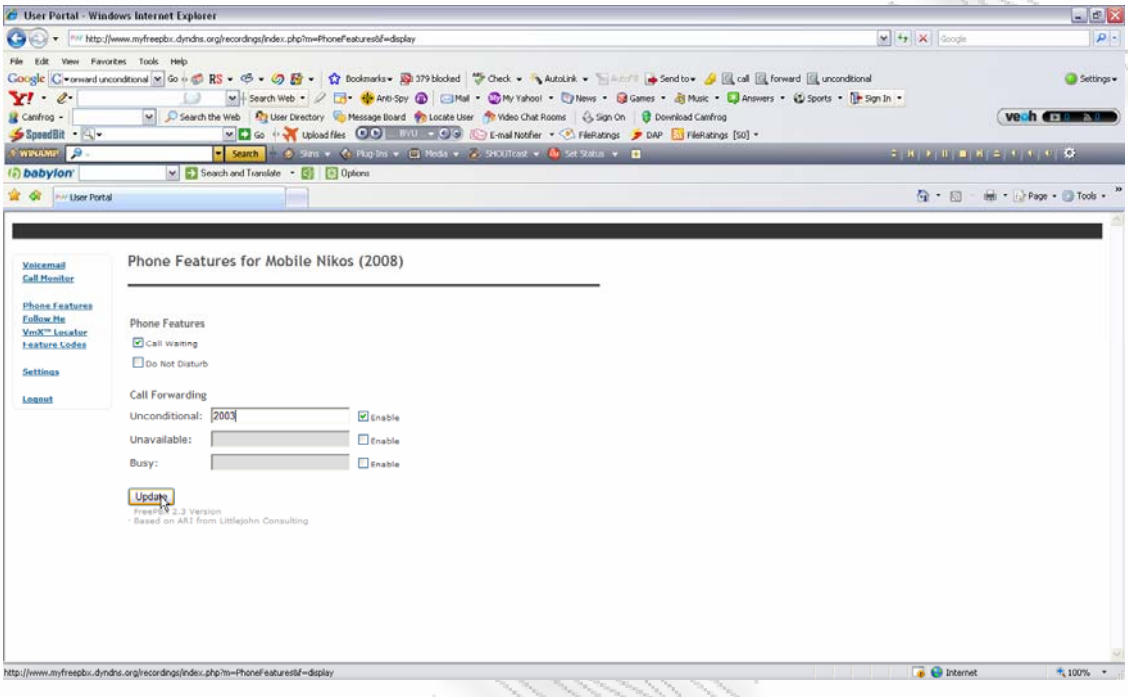


Figure 16 Enabling Unconditional Call Forwarding.

- User 2001 calls User 2008.
- Asterisk receives the SIP : INVITE message and because User 2008 has activated the Unconditional Call Forwarding Feature activated,
- Asterisk forwards this SIP :INVITE message immediately to User 2003.
- When User 2003 answers the phone, User 2001 and User 2003 can talk.

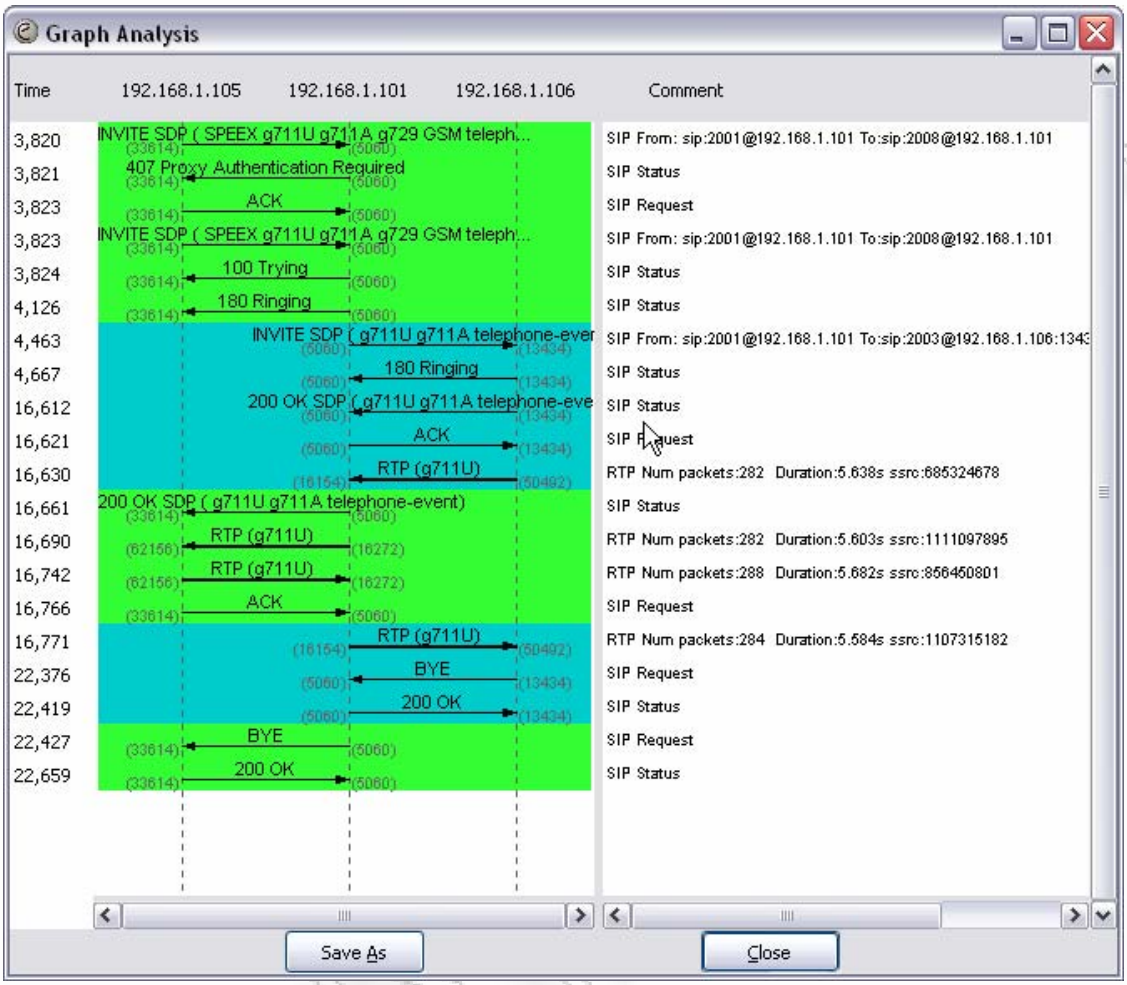


Figure 17 Unconditional Call Forwarding Call Flow.

5.5.2 Conditional Call Forwarding

Conditional call forwarding redirects incoming calls according to the specific call circumstance:

1. Forward if busy redirects incoming calls to the desired number when you are already on the phone
2. Forward if not answered redirects incoming calls to the desired number if you do not answer, normally within 15 seconds
3. Forward if out of reach redirects incoming calls to the desired number if the phone is switched off or is outside of the coverage area

The first case will be described and analyzed, the call forward when busy feature.

5.5.2.1 Scenario and Analysis of Call Forward Busy

With the same way as described for Unconditional Call Forwarding (figure 16) the User 2008 can activate Call Forward Busy feature.

- User 2001 calls User 2008. User 2008 is busy and send as an answer to the initial SIP : INVITE message a SIP : 486 Busy Here message.
- Then Asterisk server initiates a call to User 2003. The SIP : INVITE message that is sent by Asterisk server contains the SDP data of User 2001.
- When User 2003 answers the phone, a SIP : 200 OK message is sent with his SDP data.
- After that User 2008 and User 2003 can talk.

The SIP Call Flow of Call Forward Busy feature can be shown in figure 18.

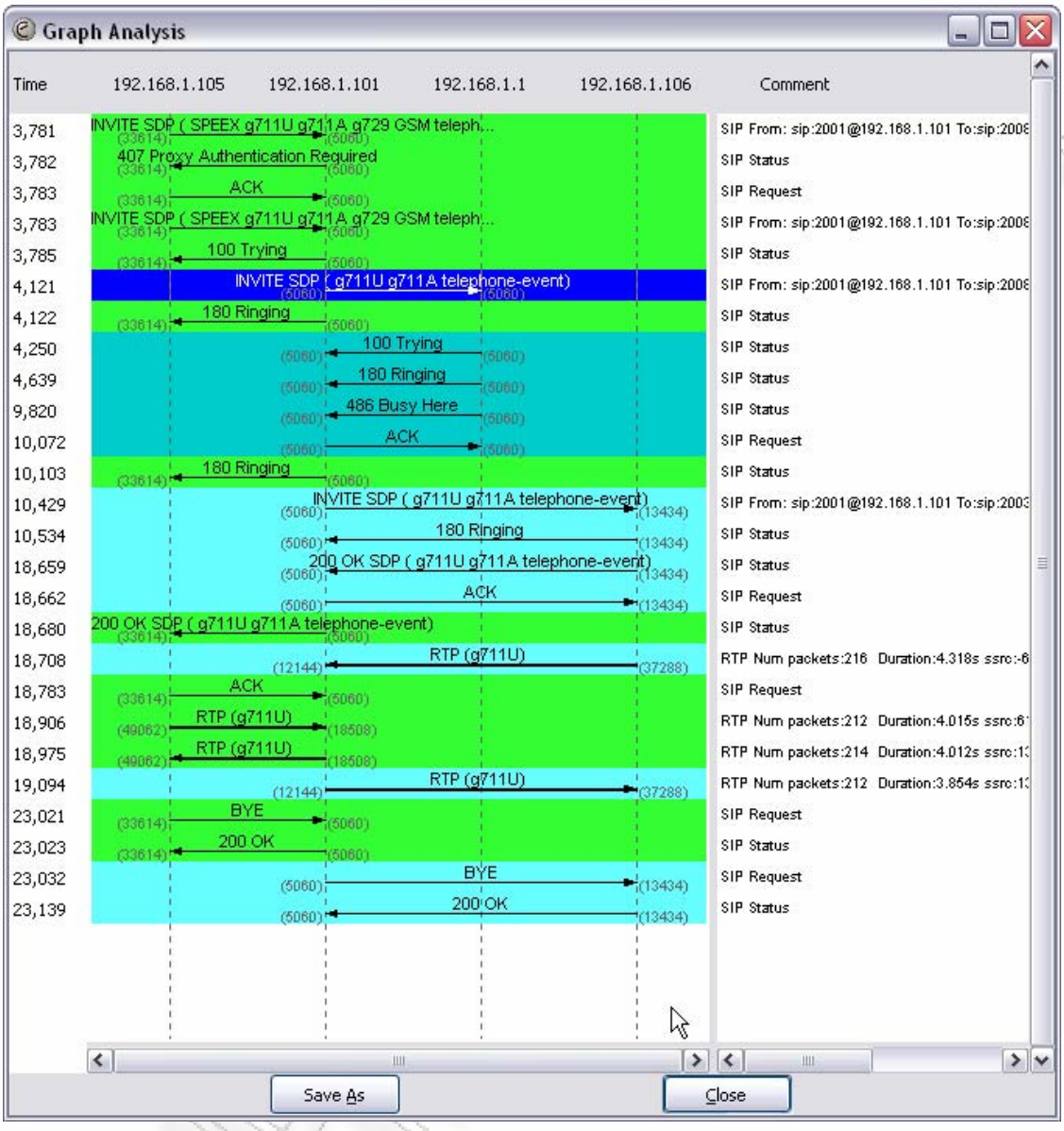


Figure 18 Call Forwarding Busy Call Flow.

5.6 Test Results and Analysis

Based on the message flow analysis, Asterisk showed its implementation complying with the standard defined in RFC 3261. X-lite clients and Nokia N95 embedded SIP client showed also complying with the standard defined in RFC 3261.

The test results were achieved by monitoring and analyzing the traffic using the ethereal tool. All tests were conducted 3 to 5 times for each scenario and the results were identical. Table 5 summarizes these test results.

Table 5 Achieved test results

Tested feature	**Asterisk**	**X-lite client**	**Nokia n95 embedded SIP client**
Registration	Stable	Stable	Stable
Click – to – call service	Stable	Stable	Stable
Voicemail to e-mail service	Stable	Stable	Stable
Unconditional Call Forwarding	Stable	Stable	Stable
Call Forward if Busy	Stable	Stable	Stable

6 Conclusions

In this thesis, the base SIP standard and its extensions were investigated, including their basic features, operations, and functions. The base SIP standard defines six methods for establishing media sessions. In order to support more kinds of services like Instant Messaging and Presence, SIP extension methods are defined in different RFCs or Internet drafts.

The representative of traditional telecommunication signalling protocol H.323 was compared with the representative of IP network signalling protocol SIP, and the comparison showed that SIP seems more suitable for the Internet world.
Implementations were reviewed in this thesis, including Asterisk, SIPSet, X-Lite and SER. By evaluating the features, availability and development status and ease, Asterisk, one of the candidates, was selected as the pilot system and installed in the lab. Accordingly, X-Lite was selected from a list of Asterisk-supported clients because it is free and widely used.

During the experimental test, voicemail to e-mail and click-to-dial services were deployed in the Laboratory. The needed configuration and implementation were made. Also some calling features concerning call forwarding were studied in terms of rfc compliance and Quality of Service.

Traffic analysis between Asterisk and the clients suggests that Asterisk is a stable and efficient SIP server. Because it is available as an open source, Asterisk therefore provides a good platform for the research community to explore SIP-enable features.

6.1 Future work

The click to call service can be extended and ported to a 3-d map –based directory like Google maps. The user would be able by this to find in the map a company or a shop for example and when move the cursor on the wanted location some information would be visible concerning the specific location like the telephone numbers. Then by clicking on the available information mentioned (this information would be a hyperlink) to direct call to the given telephone of the company.

The Asterisk PBX has a lot of “free space” for implementations in order to be fully compliant with the latest SIP standards and in order to support new features and services. The chan_sip.c file is the file that handles all SIP messages and can be extended in order to support Instant Messaging and Presence by adding the SIP method MESSAGE.

The voice mail to email service can also be “modified”. A very interesting service would be the following scenario. A user would be able to call to a default extension assigned to him. By calling this extension Asterisk would be able, using a text-to-speech engine, to read user’s e-mails from the e-mail address assigned to user’s extension.

Appendix A

A Configuration and Installation of Asterisk in Laboratory

The following steps were the configuration and installation procedure of Asterisk in the laboratory.

A1. Install CentOS, enabling the following packages:

*DNS Server

*Web Server

*Mail Server

*MySQL Database

*Development Tools

yum install nano

reboot

A2. Edit Network settings

nano /etc/sysconfig/network

HOSTNAME=myfreepbx.dyndns.org (internal hostname name here)

Ctrl-X to save, 'Y' to confirm

nano /etc/sysconfig/network-scripts/ifcfg-eth0

IPADDDR=192.168.1.20

NETMASK=255.255.255.0

NETWORK=192.168.1.0

Ctrl-X to save, 'Y' to confirm

echo "options {" >> /etc/named.conf

echo " directory \"/var/named\";" >> /etc/named.conf

echo " dump-file \"/var/named/data/cache_dump.db\";" >> /etc/named.conf

echo " statistics-file \"/var/named/data/named_stats.txt\";" >> /etc/named.conf

echo "};" >> /etc/named.conf

echo "include \"/etc/rndc.key\";" >> /etc/named.conf

service named start

chkconfig named on

nano /etc/resolv.conf

search myfreepbx.dyndns.org (internal domain name here)

nameserver 192.168.1.5

nameserver 127.0.0.1

nano /etc/hosts

127.0.0.1 myfreepbx.dyndns.org (your internal hostname name here)

127.0.0.1 asterisk1.local

127.0.0.1 localhost

Ctrl-X to save, 'Y' to confirm

iptables -P INPUT ACCEPT

iptables -P OUTPUT ACCEPT

iptables -P FORWARD ACCEPT

iptables -F

iptables -X

/etc/init.d/iptables save

service network restart

A3. Update:

yum -y update

A4. Disable Selinux:

echo "selinux=disabled" > /etc/selinux/config

reboot

A5. Install dependencies and extra packages:

yum install e2fsprogs-devel keyutils-libs-devel krb5-devel libogg libselinux-devel libsepol-devel libxml2-devel libtiff-devel gmp php-pear php-pear-DB php-gd php-mysql php-pdo kernel-devel ncurses-devel audiofile-devel libogg-devel openssl-devel mysql-devel zlib-devel perl-DateManip sendmail-cf

cd /usr/src

wget http://easynews.dl.sourceforge.net/sourceforge/lame/lame-3.97.tar.gz

tar zxvf lame-3.97.tar.gz

cd lame-3.97

./configure

make

make install

A6. Install Asterisk and FreePBX:

cd /usr/src

wget http://downloads.digium.com/pub/asterisk/asterisk-1.4-current.tar.gz

wget http://downloads.digium.com/pub/asterisk/asterisk-addons-1.4-current.tar...

wget http://downloads.digium.com/pub/zaptel/zaptel-1.4-current.tar.gz

wget http://downloads.digium.com/pub/libpri/libpri-1.4-current.tar.gz

wget http://easynews.dl.sourceforge.net/sourceforge/amportal/freepbx-2.3.1.ta...

tar zxvf asterisk-1.4-current.tar.gz

tar zxvf asterisk-addons-1.4-current.tar.gz

tar zxvf zaptel-1.4-current.tar.gz

tar zxvf libpri-1.4-current.tar.gz

tar zxvf freepbx-2.3.1.tar.gz

```
cd /usr/src/zaptel-1.4-current

./configure
make
make install
make config
/sbin/ztcfg

echo "/sbin/ztcfg" >> /etc/rc.d/rc.local

cd /usr/src/libpri-1.4-current

./configure
make
make install

cd /usr/src/asterisk-1.4-current

useradd -c "Asterisk PBX" -d /var/lib/asterisk asterisk
mkdir /var/run/asterisk
mkdir /var/log/asterisk
chown -R asterisk:asterisk /var/run/asterisk
chown -R asterisk:asterisk /var/log/asterisk
chown -R asterisk:asterisk /var/lib/php/session/

nano +231 /etc/httpd/conf/httpd.conf
```

Change User apache and Group apache to User asterisk and Group asterisk.

Ctrl-X to save, 'Y' to confirm

nano +329 /etc/httpd/conf/httpd.conf

Change AllowOverride None to AllowOverride All

Ctrl-X to save, 'Y' to confirm

./configure

make

make install

/etc/init.d/mysqld start

cd /usr/src/freepbx-2.3.1

mysqladmin create asterisk

mysqladmin create asteriskcdrdb

mysql asterisk < SQL/newinstall.sql

mysql asteriskcdrdb < SQL/cdr_mysql_table.sql

mysql

GRANT ALL PRIVILEGES ON asteriskcdrdb.* TO asteriskuser@localhost IDENTIFIED BY 'SOMEPASSWORD';

GRANT ALL PRIVILEGES ON asterisk.* TO asterisk@localhost IDENTIFIED BY 'nikos1982';

flush privileges;

\q

mysqladmin -u root password 'nikos1982'

cd /usr/src/asterisk-addons

./configure

make

make install

cd /usr/src/freepbx-2.3.1

./start_asterisk start

./install_amp --username=root --password=nikos1982

echo "/usr/local/sbin/amportal start" >> /etc/rc.local

chkconfig httpd on

chkconfig mysqld on

Open browser to http://www.myfreepbx.dyndns.org/admin

Click red bar in FreePBX

A7. Fix ARI password:

nano -w /var/www/html/recordings/includes/main.conf.php

$ari_admin_password = "SOMEPASSWORD";

Ctrl-X to save, 'Y' to confirm

A8. Configure Sendmail:

nano /etc/mail/sendmail.mc

define(`SMART_HOST', `relay.myfreepbx.dyndns.org)dnl

MASQUERADE_AS(`myfreepbx.dyndns.org')dnl

FEATURE(`masquerade_envelope')dnl

Ctrl-X to save, 'Y' to confirm

make -C /etc/mail

A9. Edit sip_nat.conf for proper NAT:

nano /etc/asterisk/sip_nat.conf

localnet=192.168.1.0/255.255.255.0

externhost=myfreepbx.dyndns.org (Set your external hostname name here)

externrefresh=10

nat=yes

qualify=yes

canreinvite=no

Ctrl-X to save, 'Y' to confirm

A10. Add extra codecs to config:

nano /etc/asterisk/sip_custom.conf

allow=gsm

allow=h261

allow=h263

allow=h263p

videosupport=yes

Ctrl-X to save, 'Y' to confirm

nano /etc/asterisk/iax_custom.conf

allow=gsm

allow=h261

allow=h263

allow=h263p

videosupport=yes

Ctrl-X to save, 'Y' to confirm

asterisk -rx reload

A11. Edit voicemail config:

nano /etc/amportal.conf

The web interface on your PBX will be accessible on the internet:

AMPWEBADDRESS=www.myfreepbx.dyndns.org (External hostname name)

Ctrl-X to save, 'Y' to confirm

or if your users will NOT have access to the web interface:

nano /etc/asterisk/vm_email.inc

remove "Visit http://AMPWEBADDRESS/cgi-bin/vmail.cgi?action=login&mailbox=${VM_MAILBOX} to check your voicemail with a web browser.\n"

Ctrl-X to save, 'Y' to confirm

nano /etc/asterisk/vm_general.inc

serveremail=myfreepbx@dyndns.org ; Who the e-mail notification should appear to come from

fromstring=Asterisk PBX ; Real name of email sender

Ctrl-X to save, 'Y' to confirm

12. Fix MOH directory:

ln -s /var/lib/asterisk/moh /var/lib/asterisk/mohmp3

asterisk -rx reload

A12. Adding SIP users using the FreePBX web user interface :

User Extension … 2001
Display Name … Nikos
Outbound CID …

Emergency CID …

Device Options

secret … `1234`

dtmfmode … `rfc2833`

Voicemail & Directory … `Enabled`

voicemail password … `1234`

email address … `giannarakis.nikolaos@gmail.com [voicemail messages emailed to this e-mail address]`

pager email address … `giannarakis.nikolaos@gmail.com [be paged when voicemail messages arrive]`

email attachment … `yes [voicemail message included in the email message]`

play CID … `yes [CallerID played when you retrieve a message]`

play envelope … `yes [date/time of the message played before`

delete Vmail … `no [voicemail message are not deleted after it's emailed to you]`

vm options …

vm context … `default`

A13. Configuring sendmail in order to e-mail voicemail messages using SMTP Relay Host:

Because almost all the hosting providers block downstream SMTP servers to reduce spam, we used a gMail account as a SMTP Relay Host.

```
cd /etc/mail
hostname -f > genericsdomain
touch genericstable
makemap -r hash genericstable.db < genericstable
mv sendmail.mc sendmail.mc.original
wget http://pbxinaflash.net/source/sendmail/sendmail.mc.gmail
mv sendmail.mc.gmail sendmail.mc
mkdir -p auth
chmod 700 auth
cd auth
```

```
echo AuthInfo:smtp.gmail.com "U:smmsp"
"I:giannarakis.nikolaos@gmail.com" "P:password" "M:PLAIN" > client-info
echo AuthInfo:smtp.gmail.com:587 "U:smmsp"
"I:giannarakis.nikolaos@gmail.com" "P:password" "M:PLAIN" >> client-
info
# Save changes (Ctrl-X, Y, then Enter)
chmod 600 client-info
makemap -r hash client-info.db < client-info
cd ..
make
service sendmail restart
```

REFERENCES

1. 3GPP – Third Generation Partnership Project: Project home page. [online] [cited November 10, 2005]. Available from: http://www.3gpp.org/

2. 3GPP – Third Generation Partnership Project: Technical Specification Group Services and System Aspects; Service requirements for the Internet Protocol (IP) multimedia core network subsystem; Stage 1 (Release 7). October 2005.

3. 3GPP – Third Generation Partnership Project: Technical Specification Group Services and System Aspects; IP Multimedia Subsystem (IMS); Stage 2 (Release 7). September 2005.

4. 3GPP – Third Generation Partnership Project: Technical Specification Group Core Network and Terminals; Signalling flows for the IP multimedia call control based on Session Initiation Protocol (SIP) and Session Description Protocol (SDP); Stage 3 (Release 5). June 2005.

5. H. Alvestrand: Mission Statement for IETF, RFC3935, October 2004.

6. America Online, Inc.: AIM.com, 2005. [online] [cited Oct 5, 2005]. Available from: http://www.aim.com.

7. M. Arango, A. Dugan, I. Elliot, C. Huitema, S. Pickett: Media Gateway Control Protocol (MGCP) Version 1.0, RFC2705, October 1999.

8. T. Berners-Lee: Uniform Resource Identifiers (URI): Generic Syntax, RFC 2396, Aug 1998.

9. Brekeke Software, Inc.: SIP Proxy Server and SIP Registrar Server [online] [cited February 18, 2005]. Available from: http://www.brekeke.com/en/products/sipserver_en.html.

10. B. Campbell, J. Rosenberg, H. Schulzrinne, C. Huitema, D. Gurle: "Session Initiation Protocol (SIP) Extension for Instant Messaging", RFC 3428, 18p, Dec 2002

11. Cisco Systems, Inc.: Vovida Open Communication Application Library User Manual [online]. Software Version 1.5.0, Guide Revision 2 [cited March 21, 2005]. Portable Document Format. Available from: http://www.vovida.org/document/vocaldoc/guide/userguide.pdf.

12. CounterPath Solutions Inc. Company Web Page [online] [cited October 4, 2005]. Available from: <http://www.xten.com>.

13. M. Day, S. Aggarwal, G. Mohr, J. Vincent: "Instant Messaging / Presence Protocol Requirements", RFC 2779, 26p, Feb 2000

14. M. Day, J. Rosenberg, H. Sugano: "A Model for Presence and Instant Messaging", RFC 2778, 17p, Feb 2000

15. Digium Inc.: Asterisk: The Open Source PBX [online] [cited March 1, 2005]. Available from: http://www.asterisk.org/about.

16. S. Donovan: "The SIP INFO Method", RFC 2976, 9p, Oct 2000

17. S. Donovan, J. Rosenberg: "Session Timers in the Session Initiation Protocol (SIP)", RFC4028, 27p, April 2005

18. DOROTEL Communications, Inc.: "What's Hot" [online] [cited March 21, 2005]. Available from: http://www.doretel.com/html/whatshot.asp.

19. K. Egevang, P. Francis: "The IP Network Address Translator (NAT)", RFC1631, May 1994.

20. P. Faltstrom: " E.164 number and DNS ", RFC 2916, Sep 2000.

21. M. Gudgin, M. Hadley, N. Mendelsohn, J. Moreau, H. Nielsen: Soap Version 1.2 Part 1: Messaging Framework, W3C Recommendation, June 2003. [online] [cited September 29, 2005]. Available from: http://www.w3.org/TR/2003/REC-soap12-part1-20030624/.

22. M. Gudgin, M. Hadley, N. Mendelsohn, J. Moreau, H. Nielsen: Soap Version 1.2 Part 2: Adjuncts, W3C Recommendation, June 2003. [online] [cited September 29, 2005]. Available from: http://www.w3.org/TR/2003/REC-soap12-part2-20030624/.

23. T. Hall: SIPing with Sametime, Sep 2002. [online] [cited October 7, 2005]. Available from: http://www-128.ibm.com/developeworks/lotus/library/ls-SIP/.

24. M. Handley, V. Jacobson: "SDP: Session Description Protocol", RFC 2327, 42p, Apr 1998

25. M. Handley, C. Perkins, E. Whelan: "Session Announcement Protocol", RFC 2974, 18p, Oct 2000

26. M. Handley, H. Schulzrinne, E. Schooler, J. Rosenberg,: "SIP: Session Initiation Protocol", RFC 2543, 153p, Mar 1999

27. M. Handley, D. Thaler, R. Kermode: "Multicast-Scope Zone Announcement Protocol (MZAP)", RFC 2776, 27p, Feb 2000

28. H. Harrison: "Zephyr on Athena (AC-34)", Version 10, October 2002. [online] [cited November 12, 2005]. Available from: http://web.mit.edu/olh/Zephyr/Zephyr.html

29. IEEE Std 1003.1, 2004 Edition Single UNIX Specification Version 3.

30. IETF Secretariat: The Internet Engineering Task Force [cited February 23, 2005]. Available from http://www.ietf.org/.

31. IETF Secretariat: Multiparty Multimedia Session Control (MMUSIC) [online] [cited March 21, 2005]. Available from: http://www.ietf.org/html.charters/mmusic-charter.html.

32. IETF Secretariat: Session Initiation Protocol (SIP) [online] [cited March 21, 2005]. Available from: http://www.ietf.org/html.charters/sip-charter.html.

33. IETF Secretariat: SIP for Instant Messaging and Presence Leveraging Extensions (simple) [online] [cited March 21, 2005]. Available from: http://www.ietf.org/html.charters/simple-charter.html.

34. International Business Machines Corporation; IBM Lotus SameTime [online] [cited February 23, 2005]. Available from: http://www-128.ibm.com/developerworks/lotus/downloads/.

35. Iptel.org: iptel.org SIP Express Router [online] [cited September 29, 2005]. Available from: http://www.iptel.org/ser/.

36. Iptel.org: Sip versus H.323 [online] [cited March 18, 2005]. Available from: http://www.iptel.org/info/trends/sip.html.

37. The ITU Telecommunication Standardization Sector (ITU-T) [cited March 18, 2005]. Available from: http://www.itu.int/ITU-T/index.html.

38. ITU-T Recommendation G.711: Pulse code modulation (PCM) of voice frequencies. November 1988.

39. ITU-T Recommendation E.164: Assigned Country Codes, May 2005.

40. Jabber Inc.: Jabber Software Foundation. [online] [cited October 5, 2005]. Available from: http://www.jabber.org/jsf/,

41. J. Kuthan, J. Janak, and Y. Rebahi: "iptel.org SIP Express Router v0.11.0 – Admin's Guide" [online] [cited March 21, 2005]. Available from: http://www.iptel.org/ser/doc/seruser/seruser.html.

42. C. de Laat, G. Gross, L. Gommans, J. Vollbrecht, D. Spence: "Generic AAA Architecture", RFC2903, p26, Aug 2000.

43. Erik Lagerway: Xten completes SIP softphone offering with IM, presence and contact list management using simple, xcap & webdav. Wins product of the year award [online] [cited February 18, 2005]. Available from: http://sipthat.com/archives/000197.html.

44. Microsoft Corporation: MSN Messenger [online]. Software Version 7.0 [cited March 18, 2005]. Available from http://messenger.msn.com/.

45. Mirabilis company website. [online] [cited September 29, 2005]. Available from: http://www.icq.com.

46. N. Mitra: Soap Version 1.2 Part 0: Primer, W3C Recommendation, June 2003. [online] [cited September 29, 2005]. Available from: http://www.w3.org/TR/2003/REC-soap12-part0-20030624/.

47. C. Moore: XMPP vs. SIMPLE: The race for messaging standards. [online] [cited October 7, 2005]. Available from: http://www.infoworld.com/article/03/05/23/21FExmpp_1.html.

48. NERO AG: X-Lite : The Best Free Softphone for Mac OS and Windows powered by nikotel SIP network [online] [cited February 23, 2005]. Available from: http://www.nikotel.de/en/software.html.

49. J. Oikarinen, D. Reed: Internet Relay Chat Protocol, RFC 1459, May 1993.

50. J. Rosenberg: "A Presence Event Package for the Session Initiation Protocol (SIP)", RFC 3856, 27p, Aug 2004

51. J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, E. Schooler: "SIP: Session Initiation Protocol", RFC 3261, 269p, Jun 2002

52. A. B. Roach: "Session Initiation Protocol (SIP)- Specific Event Notification", RFC 3265, 38p, Jun2002

53. P. Saint-Andre: "Extensible Messaging and Presence Protocol (XMPP): Core", RFC 3920, 90p, Oct 2004

54. P. Saint-Andre: "Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence", RFC 3921, 107p, Oct 2004

55. H. Schulzrinne, A. Rao, R. Lanphier: "Real Time Streaming Protocol (RTSP)", RFC 2326, 92p, Apr 1998

56. H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson: "RTP: A Transport Protocol for Real-Time Applications", RFC 3550, July 2003.

57. A.V. Shutko: OSCAR (ICQ v7/v8/v9) protocol documentation, February 2005. [online] [cited September 29, 2005]. Available from: http://iserverd.khstu.ru/oscar/.

58. Henry Sinnreich, Alan B. Johnston: "Internet Communications Using SIP", John Willy & Sons, Inc, 2001

59. SIP Center: Test Area Introduction [online] [cited February 23, 2005]. Available from: http://www.sipcenter.com/sip.nsf/html/Test+Area+Introduction.

60. SIPfoundry, Inc.: About SIPfoundry [online] [cited February 23, 2005]. Available from: http://www.sipfoundry.org/about.html.,

61. R. Sparks: "The Session Initiation Protocol (SIP) Refer Method", RFC 3515, 23p, Apr 2003

62. R. Srinivasan: RPC: Remote Procedure Call Protocol Specification Version 2, RFC1831, August 1995.

63. Forrest Stroud: AOL Instant Messenger Instant Messaging Via AOL [online] [cited March 21, 2005]. Available from: http://www.winplanet.com/article/2236-3079.htm.

64. TriTech: Polycom Telephones [online] [cited March 21, 2005]. Available from: http://www.tritechcoa.com/product/b-9PZ7-138.html.

65. W3C: Extensible Markup Language (XML), Aug 2005. [online] [cited Octobe 5, 2005]. Available from: http://www.w3.org/XML/.

66. Wavestream Ltd.: Zultys IP Phones [online] [cited March 21, 2005]. Available from: http://www.wave-stream.co.uk/new_tech/frames/tech_zultys.htm.

67. D. Winer: RSS 2.0 Specification, Fall 2002. [online] [cited September 29, 2005]. Available from: http://blogs.law.harward.edu/tech/rss/.

68. Yahoo! Inc.: Yahoo! Messenger with Voice, 2005. [online] [cited October 5, 2005]. Available from: http://messenger.yahoo.com/..