VOCAL

Vovida Open Communication Application Library

Installation Guide

Software Version 1.4.0

Guide Versions The following table matches the software versions with the guide versions:

Software Version	***Guide Version***	***Date***	***Comments***
1.0.0			Internal Trials Only
1.1.0			Internal Trials Only
1.2.0	1.2	March 26, 2001	Open Release to Public
1.2.0	1.2 A	April 11, 2001	Copy edit errors corrected.
1.3.0	1.3	December 21, 2001	Support new open release to public
1.4.0	1.4	June 23, 2002	Update for release of version 1.4.0.

Version This manual is written to support VOCAL Version 1.4.0.

Support The primary location for support, information and assistance for the VOCAL system is http://www.vovida.org/. This site contains other documentation, training materials, development tools, development resources and informational mailing lists.

Preface

Introduction

This chapter is a general introduction to the System Installation manual, and provides information about the intentions and organization of the manual. It also provides information about additional resources available from http://www.vovida.org.

Objectives of this manual

This manual provides information for installing and initial provisioning for a VOCAL system. Information about adding users and assigning features is provided in the System Administration Guide.

Intended audience

This manual is intended for technicians who will be installing and provisioning the VOCAL system. These technicians should be familiar with either the Linux operating system or with the operating system on which VOCAL is being installed; and should also be familiar with Session Initiation Protocol (SIP) and the general concepts and principles of Voice over IP (VoIP) telephony networks.

Organization

This guide is organized as follows:

Chapter	***Title***	***Description***
Chapter 1	System Overview	A high level overview of the system architecture.
Chapter 2	Software Installation	The installation routine uses a command line interface (CLI). This section describes the commands, the system responses, and how to complete the routine.

Chapter	*Title*	*Description*
Chapter 3	Provisioning	The servers are configured through a graphical user interface (GUI). This sections describes the screen functions, the fields, and how to add and maintain system servers.
Appendix A	Engineering Guidelines	This appendix describes system capabilities, limitations and hardware requirements. It also provides information about building a high capacity, redundant system.
Appendix B	Resource on the Web	This appendix lists Web resources that our developers and engineers follow regularly.

Documentation Conventions

The following is a list of conventions used in this guide:

Convention	*Description*
bold text	Names of elements found on the GUI screen, including buttons, and selectable entities such as, servers and server groups.
< >	Text that appears between angle brackets describes variables such as, <group name>.
`courier font`	System responses and prompts either from the CLI or GUI.
`bold courier font`	Indicates information that you must enter.
■**Note**	Highlights points of additional interest for the user.
Caution	Be careful, this symbol highlights a potential for equipment damage or loss of data.

Additional resources

Publications

A System Administration Guide, which covers adding users, SNMP message flows, call flows and working with features is also available from Vovida.org (http://www.vovida.org)

On-Line Resources

Vovida.org is a community web site dedicated to providing a forum for open source software used in datacom and telecom environment. This site was created to provide an environment where open source communications information and software can be easily located, accessed, retrieved and shared.

Table of Contents

1

VOCAL System Overview

Chapter Contents This chapter describes the Vovida Open Communication Application Library (VOCAL) system from a high-level point of view, highlighting the Session Initiation Protocol (SIP) and the functionality of the VOCAL system.

Topic *(continued)*	**See Page**

Overview

Introduction

This section describes the VOCAL system from a high-level point-of-view.

What is VOCAL?

The VOCAL system is a distributed network of servers that provides Voice Over Internet Protocol (VoIP) telephony services. VOCAL supports devices that communicate Session Initiation Protocol (SIP, RFC 2543), Media Gateway Control Protocol (MGCP) or H.323 messages. VOCAL also supports analog telephones via residential gateways.

VOCAL supports on-network and off-network calling. Off-network calling enables subscribers to connect to parties through either the Internet or the Public Switched Telephone Network (PSTN).

High-Level System View

Figure 1-1 shows a high-level, simplified view of the system.

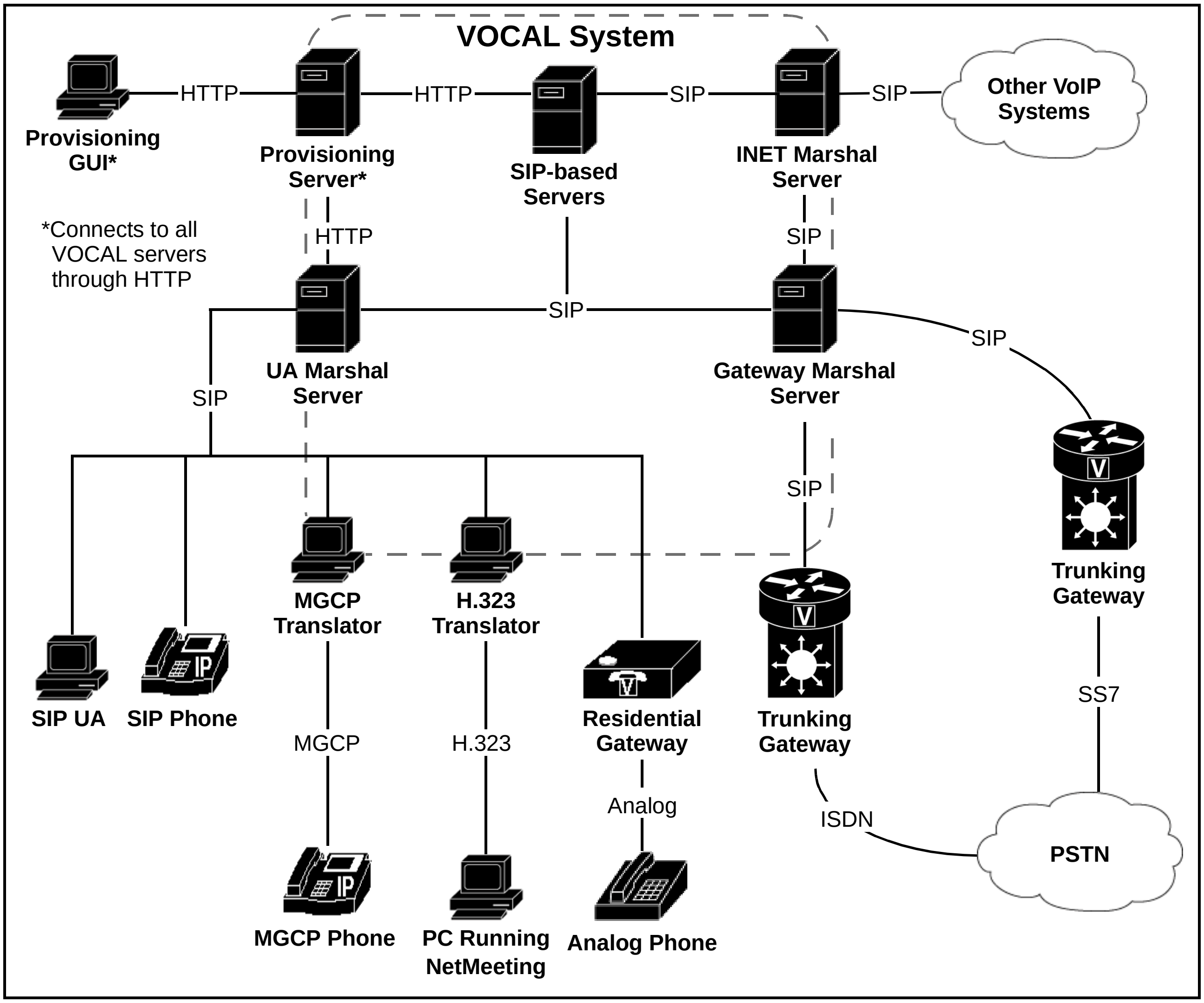


Figure 1-1. Simplified View of the VOCAL System

System Components

From a high-level point-of-view, the VOCAL system appears as an assembly of basic components. These components are described below in Table 1-1.

Table 1-1. VOCAL System Components

Component	Description
VOCAL System	This is the telephony application. Figure 1-1 shows an abstract representation of the VOCAL system server modules. A description of each server appears in the next section, see “Servers” on page 1-6. **Protocols** The VOCAL system uses several protocols to communicate between its components. The call signaling processes use SIP messaging to communicate internally within the VOCAL system and externally with gateways and IP phones. For more information about SIP, see “SIP Overview” on page 1-8. For more information about the other protocols used in the system, see “Compatible Protocols” on page 1-9.
GUI	The graphical user interface (GUI) enables technicians to provision the system, and administrators to set up users and monitor the system’s performance. The GUI is web-enabled and requires a Java plug-in to run in a web browser. For more information, see “Accessing the Java Provisioning” on page 2-26. In Version 1.4.0, you can bypass the requirement for the Java plug-in by using a simplified HTTP-based provisioning GUI.
IP Phone	VOCAL supports a variety of phone appliances including SIP phones and SIP User Agent (UA) software applications. SIP phones may be connected to the VOCAL system over any IP network.
Translators	MGCP- and H.323-based appliances require translators to convert their messages into SIP before they can communicate with the VOCAL system. The translators are included in VOCAL.

Table 1-1. VOCAL System Components (Continued)

Component	Description
Gateways	Gateways not only provide entry points between networks, they also provide translation between SIP-based networks and other network types. The VOCAL system works with two types of gateways, the Residential Gateway and the Trunking Gateway. **Residential Gateway** Residential gateways translate analog signals into IP packets, to permit subscribers with analog phone sets/devices to make and receive SIP-based calls. **Trunking Gateway** Trunking gateways permit SIP-based networks to exchange calls with end-points on the PSTN, by providing translation between SIP messages and one of these signal types: • Analog • Channel Associated Signaling (CAS) • Primary Rate Interface (PRI)

Servers

Description Table 1-2 describes the server modules included in the VOCAL system.

Table 1-2. VOCAL Server Modules

Server Modules	***Description***
Marshal Server	The Marshal Server (MS) is an implementation of the SIP proxy server and acts as the initial point of contact for all SIP signals that enter the VOCAL system. The MS provides authentication, forwarding and billing functions. For more information about SIP proxy servers, see ”SIP User Agents and Servers” on page 1-10. For more information about authentication, see “Authentication” on page 1-33. For more information about forwarding, see ”Call Control” on page 1-16. For more information about billing, see “Call Detail Records and Billing” on page 1-36.
Redirect Server	The Redirect Server (RS) is a combined implementation of the SIP redirect, registration and location servers. The RS stores contact and feature data for all registered subscribers and a dialing plan to enable routing for off-network calls. For more information about SIP servers, see ”SIP User Agents and Servers” on page 1-10. For more information about registration, see “Authentication” on page 1-33.
Call Detail Record Server	The Call Detail Record (CDR) server receives call data from the Marshal Servers and formats it into data that can be transmitted to third party billing systems for invoicing. For more information about billing, see “Call Detail Records and Billing” on page 1-36.
Network Manager	The Network Manager provides the administrator with the ability to monitor the system through Simple Network Management Protocol (SNMP) messages. For more information about the Network Manager, see the System Administration Guide.
Voice Mail Server	The Voice Mail server provides unified messaging whereby voice mail messages can be distributed as .wav files attached to e-mail messages.
Feature Server	The Feature Servers are another implementation of the SIP proxy server. These servers are scripted in Call Processing Language (CPL) and provide basic system features such as Call Forward and Call Blocking. For more information about features, see “Features” on page 1-42.
JTAPI Server	The VOCAL system includes an implementation of the Core JTAPI package that supports basic third-party call control capability, and a basic User Agent application, the VOCALpad, that utilizes the implementation. For more information, see “JTAPI Servers” on page 3-60.
Provisioning Server	The Provisioning Server (PS) stores data records about each system user and server module, and distributes this information throughout the system via a subscribe-notify model. The PS provides a web-enabled graphical user interface (GUI) to permit technicians and system administrators to manage the system. The For more information about provisioning servers, see Chapter 3, Provisioning in this guide. For more information about provisioning users, see the System Administration Guide.

Table 1-2. VOCAL Server Modules (Continued)

Server Modules	Description
Policy Server	The Policy Server has been designed to use Common Open Policy Service (COPS, RFC 2748) to provide Quality of Service (QoS) bandwidth reservation for calls or call segments that are transmitted over the Internet. The Policy Server is also capable of using Open Settlement Protocol (OSP, a product of the Telecommunication and Internet Protocol Harmonization over Networks (TIPHON) project at the European Telecommunications Standards Institute (ETSI)) to interact with clearinghouses for authorization, authentication and accounting (AAA). For more information about QoS and OSP, see “Quality of Service” on page 1-25.
Heartbeat Server	The Heartbeat Server monitors the flow of pulsing signals emitted by the other servers, and provides information about to the flow of heartbeats to the Simple Network Management Protocol (SNMP, RFC 1157) GUI. This information helps the System Administrator know if the server modules are up or down. For more information about heartbeats, see “Heartbeat Servers” on page 3-62. For more information about the SNMP GUI, see the System Administration Guide.

Scaling the System

The VOCAL system can be provisioned onto a single hardware unit or onto multiple hosts. While a single hardware unit may be useful for laboratory testing, systems that are intended to support customers are normally scaled up to larger systems that may include any number of hosts. See Appendix A, Engineering Guidelines for more information about scaling the system.

SIP Overview

Introduction

This section describes SIP with respect to its features and benefits, compatible protocols, user agents and servers along with basic call flows.

What is SIP?

The Session Initiation Protocol (SIP: RFC 2543) is an ASCII-based, peer-to-peer protocol designed to provide rendezvous services over the Internet. SIP is an Internet Engineering Task Force (IETF) specification that was derived from Hyper-text Transfer Protocol (HTTP: RFC 2616) and Simple Mail Transfer Protocol (SMTP: RFC 821).

SIP, along with Media Gateway Control Protocol (MGCP: RFC 2705), and H.323 (an International Telecommunications Union (ITU) specification), is one of three commonly used open protocols for VoIP implementations. A slide presentation that compares these three protocols, called VoIP Protocol Overview, is available on http://www.vovida.org.

Features and Benefits

Table 1-3 describes some of the features and benefits of SIP-based systems:

Table 1-3. SIP Features and Benefits

Feature	*Benefit*
Simplicity	The SIP stack is smaller than other VoIP protocols. SIP can be considered as a simple toolkit that enables smart endpoints, gateways, processes and clients to be built and implemented.
Scalability	The peer-to-peer architecture permits inexpensive scaling. When compared to other Voice over IP (VoIP) protocols, the hardware and software requirements for adding new users to SIP-based systems is greatly reduced.
Distributed Functionality	A decentralized intelligence permits more functionality within each component. Changes made to specific components have a minor impact on the rest of the system. It is possible to connect one SIP phone to another with an ethernet cable and make calls between the sets without the aid of any other server modules. The other system components become useful when the network requires more than two phones.
Internet-enabled	SIP-based systems can take advantage of the growth of the Internet. Translating gateways permit SIP-based systems to contact parties on the Public Switched Telephone Network (PSTN) without being encumbered by its legacy standards.

Compatible Protocols

Introduction

This section describes protocols that are compatible with SIP.

Protocols and Descriptions

SIP can work alone or together with the following protocols:

Table 1-4. Compatible Protocols

Protocol Acronym	Protocol Name	Description
COPS	Common Open Policy Service	Used to signal network, routers and switches with requests for Quality of Service. COPS is a companion protocol to RSVP.
DHCP	Dynamic Host Configuration Protocol	Helps systems automatically configure network settings.
DNS	Domain Name System	Resolves host names to IP addresses.
HTTP	Hypertext Transfer Protocol	HTTP is the standard protocol used for serving web pages over the Internet.
MGCP	Media Gateway Control Protocol	MGCP is a master/slave protocol whereby the gateways are under the direct control of the user agents. SIP-based systems can communicate to MGCP endpoints through translators.
OSP	Open Settlement Protocol	OSP is used to exchange authorization, authentication and accounting (AAA) information with clearinghouse servers.
RADIUS	Remote Authentication Dial-In User Service	A freely available distributed security system that can be used to transmit call detail records to a billing system.
RSVP	Resource Reservation Protocol	Enables SIP-based systems to reserve bandwidth for call sessions. RSVP is a companion protocol to COPS.
RTP	Real-time Transport Protocol	Provides voice channels between end points.
SDP	Session Description Protocol	Describes the content of multi-media sessions. SDP messages are attached to SIP messages as Multi-Purpose Internet Mail Extensions (MIME).
TCP	Transmission Control Protocol	Can be used as the underlying transport protocol in SIP-based systems.
UDP	User Datagram Protocol	Provides best effort service to deliver packets with minimal overhead and minimal delay.

SIP User Agents and Servers

Introduction

This section describes SIP User Agents and Servers, and how they function within the VOCAL system.

User Agents

User Agents (UA's) are specified in RFC 2543 as applications such as, SIP phones and software that initiate and receive calls over a SIP network.

Servers

Servers are specified in RFC 2543 as application programs that accept requests, service requests and send back responses to those requests.

Table 1-5 describes the servers included in RFC 2543 and how they function within the VOCAL system.

Table 1-5. SIP Servers

Server Type	***RFC 2543 Definition***	***VOCAL Functionality***
Location Server	A Location Server can be used by a SIP redirect or proxy server to obtain information about a called party's possible location. The location server can also be an entity outside of the SIP network that uses an alternative protocol, such as Telephony Routing over IP (TRIP, RFC 3219) to communicate with the Redirect Server.	The Location server is a logical function within the VOCAL Redirect server.
Proxy Server	An intermediary program that acts as both a server and a client for the purpose of making requests on behalf of other clients. Unlike User Agents, Proxy Servers do not initiate new SIP requests. A Proxy Server interprets, and, if necessary, rewrites a request message before forwarding it. Requests are serviced internally or by passing them on, possibly after translation, to other servers.	The VOCAL system includes specialized SIP Proxy servers called Marshal and Feature servers.
Redirect Server	A redirect server is a server that accepts a SIP request, maps the address into zero or more new addresses and returns these addresses to the client. Unlike a proxy server, it does generate SIP requests on behalf of UA's and it does not accept calls.	The SIP Redirect server is a logical function within the VOCAL Redirect server.
Registrar Server	A registrar is a server that accepts REGISTER requests. A registrar is typically co-located with a proxy or redirect server and *may* offer location services.	The SIP Registrar server is a logical function within the VOCAL Redirect server.

Basic SIP Call Flow

Introduction

This section illustrates a simple call flow. More complex examples can be found in the System Administrator's Guide.

Call Scenario

Figure 1-2 shows a simple SIP phone call where user A is calling user B.

Figure 1-2. Basic Connection Between Two SIP Phones

Call Flow

Figure 1-3 and Table 1-6 describe the SIP messages exchanged for call establishment and tear down.

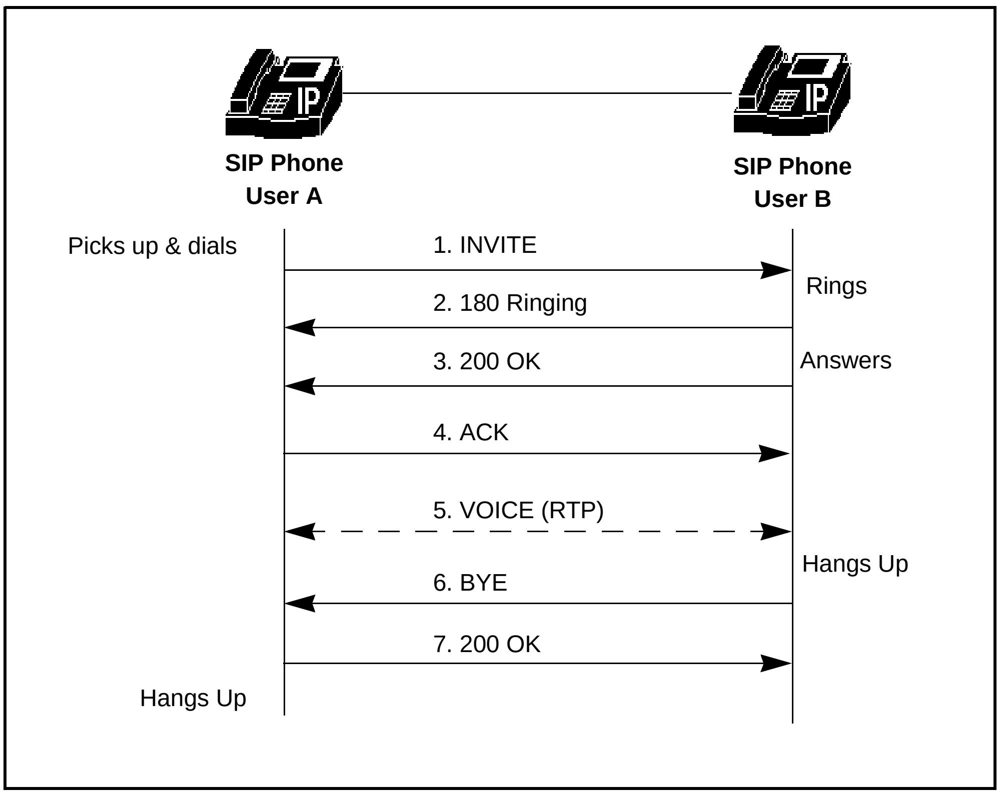


Figure 1-3. Basic Call Flow Diagram

Call Flow Details

The following table explains each of the “hops” shown in Figure 1-3.

Table 1-6. Call Flow Details

Step	Description
1	INVITE: User A initiates a call to User B.
2	180 Ringing: User B sends a ringing signal back to User A.
3	200 OK: User B picks up.
4	ACK: User A acknowledges that it received the 200 message.
5	VOICE: A two-way voice channel is established over Real-time Transport Protocol (RTP) and a conversation takes place between User A and B.
6	BYE: User B hangs up.
7	200 OK: The call is torn down and User A hangs up.

SIP Messages

Overview

SIP messages can be divided into requests and responses.

Request messages

Table 1-7 shows a few of the most commonly used SIP request messages.

Table 1-7. Some SIP Request Messages

SIP Request Message	Description
INVITE	Indicates that the user or service is being invited to participate in a session.
ACK	Confirms that the client has received a final response to an INVITE request.
BYE	Indicates that the user wishes to terminate the call.
REGISTER	Indicates that a User Agent is attempting to add its address to the Redirect server’s user database.
CANCEL	Cancels a pending request but does not affect a completed request.

Response Messages

The SIP response messages are numbered, and the first digit in each response number indicates the type of response. Table 1-8 explains the different message types.

Table 1-8. Some SIP Response Messages

SIP Response Message Types	***Description***
1xx	Information Responses For example: 180 Ringing
2xx	Successful Responses For example: 200 OK
3xx	Redirection Responses For example: 302 Moved Temporarily
4xx	Request Failures Responses For example: 403 Forbidden
5xx	Server Failure Responses For example: 504 Gateway Time-out
6xx	Global Failure Responses For example: 600 Busy Everywhere

Further examples of these messages are shown in the following sections where call flows through distributed networks are discussed.

For More Information

For more information about SIP messages, see RFC 2543.

Vocal System Functionality

Introduction

This section provides a high-level overview of message flows between VOCAL system components for selected functions.

What is Functionality?

Functionality refers to how the system components interact with each other to produce desired results. These results include phone calls being established and torn down, new users being added to the system, unauthorized users being kept out, and customers receiving invoices for the service.

In this guide, functionality is organized into discussions about the VOCAL system, the operation support system and the features.

Illustration

Figure 1-4 shows a high-level view of some of the VOCAL system elements and how they connect to outside entities such as User Agents, billing servers and others.

The connections between VOCAL and User Agents, gateways, clearing houses, billing servers and other VoIP systems are explained later in this chapter.

The other connections are optional and are to be documented in the user guides.

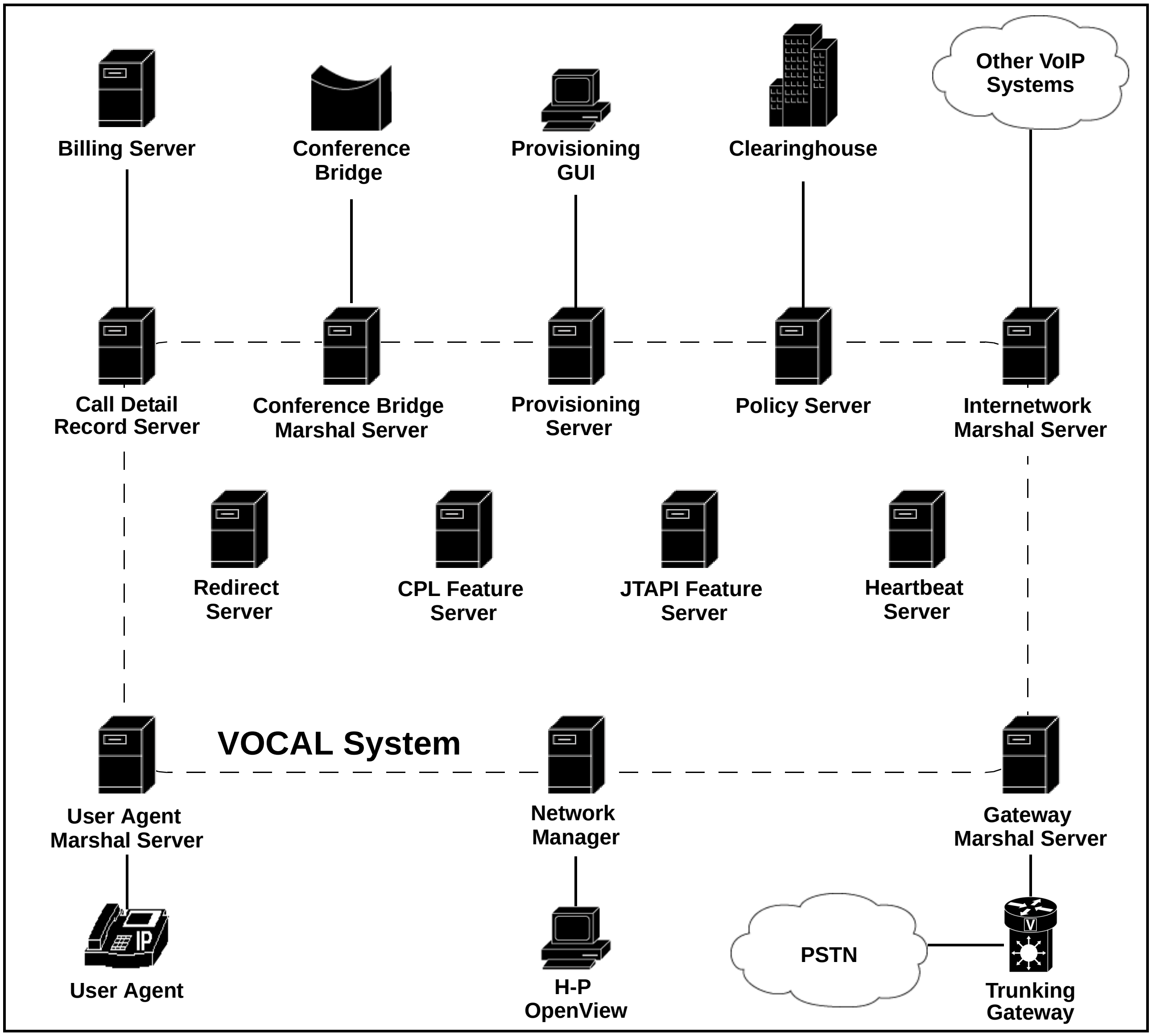


Figure 1-4. High-Level View of the Vocal System

SIP-Based Call Control

What is SIP-Based Control?

The SIP-Based Call Control portion of the VOCAL system includes those elements that enable call processing.

Signaling

What is Signaling?

The VOCAL system uses SIP messages to signal requests and responses between the core, call processing servers. For examples of SIP messages, see "SIP Messages" on page 1-12.

Call Control

Introduction

This section describes how calls are controlled over the VOCAL system through SIP messages, and how these messages are transmitted when the call is routed to the PSTN or to a feature server.

What is Call Control?

Call control is the ability to initiate, establish and tear down calls.

Diagram #1: Call Initiation

Figure 1-5 shows a SIP phone initiating a call by sending an INVITE message through the VOCAL system.

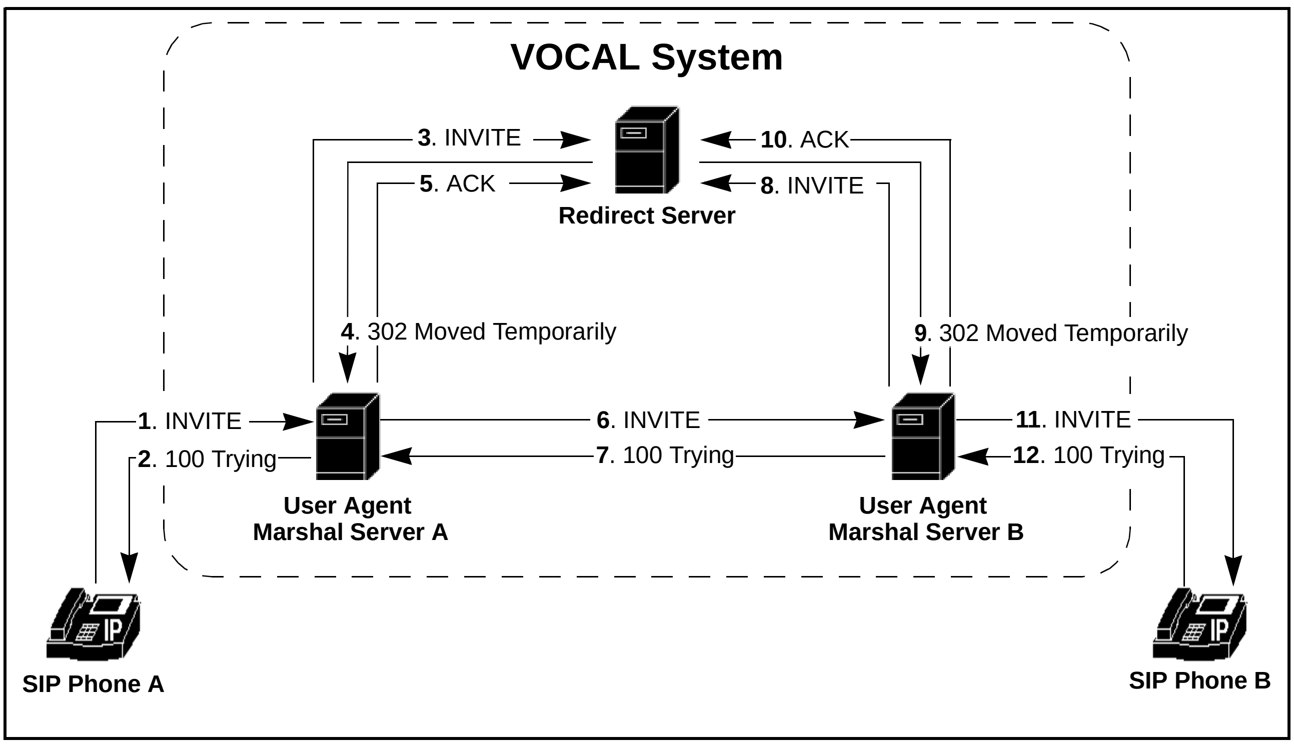


Figure 1-5. Call Initiation

Messages 1 - 9 Described

Table 1-9 describes the messages illustrated in Figure 1-5.

Table 1-9. Interactions Shown in Figure 1-5

Interaction	*Step*	*Description*
SIP Phone A to UAMS A	***1-2***	SIP Phone A and SIP Phone B are on the local IP network. SIP Phone A sends an INVITE message intended for SIP Phone B. The INVITE is received by User Agent Marshal server (UAMS) A, which responds with a 100 Trying message to stop retransmissions of the INVITE.
UAMS A to the RS	***3***	UAMS A authenticates the user and forwards the INVITE message to the Redirect server (RS). This is the normal routine: the UAMS forwards all INVITE's from authorized users to the RS.
	4	The RS responds with a 302 Moved Temporarily, message and sends it to UAMS A. The 302 message servers two purposes: • It informs the UAMS that the INVITE was not intended for the RS. • It provides routing information that enables the UAMS to forward the INVITE towards its intended destination. This destination could be a Feature or a Marshal server. Feature severs are explained later in this chapter.
	5	UAMS A sends an ACK message back to the RS acknowledging receipt of the 302 message. This completes the transaction.
Forwarding the INVITE Message to SIP Phone B	***6 - 12***	UAMS A forwards the INVITE message to UAMS B, the proxy server for the intended destination, which responds with a 100 Trying message. UAMS B forwards the INVITE message to the RS, which responds with a 302 Moved Temporarily message and sends it to UAMS B. UAMS B sends an ACK message back to the RS acknowledging receipt of the 302 message, and forwards the INVITE message to SIP Phone B, which responds with a 100 Trying message. The INVITE, having arrived at its final destination, makes the phone ring. The ringing can be a sound, a visual indicator, a vibration, a combination of these indicators or any indicator that has been implemented in the phone.

Diagram #2: Call Establishment

Figure 1-6 shows SIP phone B responding and setting up an RTP path with SIP Phone A.

■ **Note**
In a distributed network, the RTP path may travel over the VOCAL system's backbone without being processed by any of the servers. The RTP path may also bypass the VOCAL system altogether.

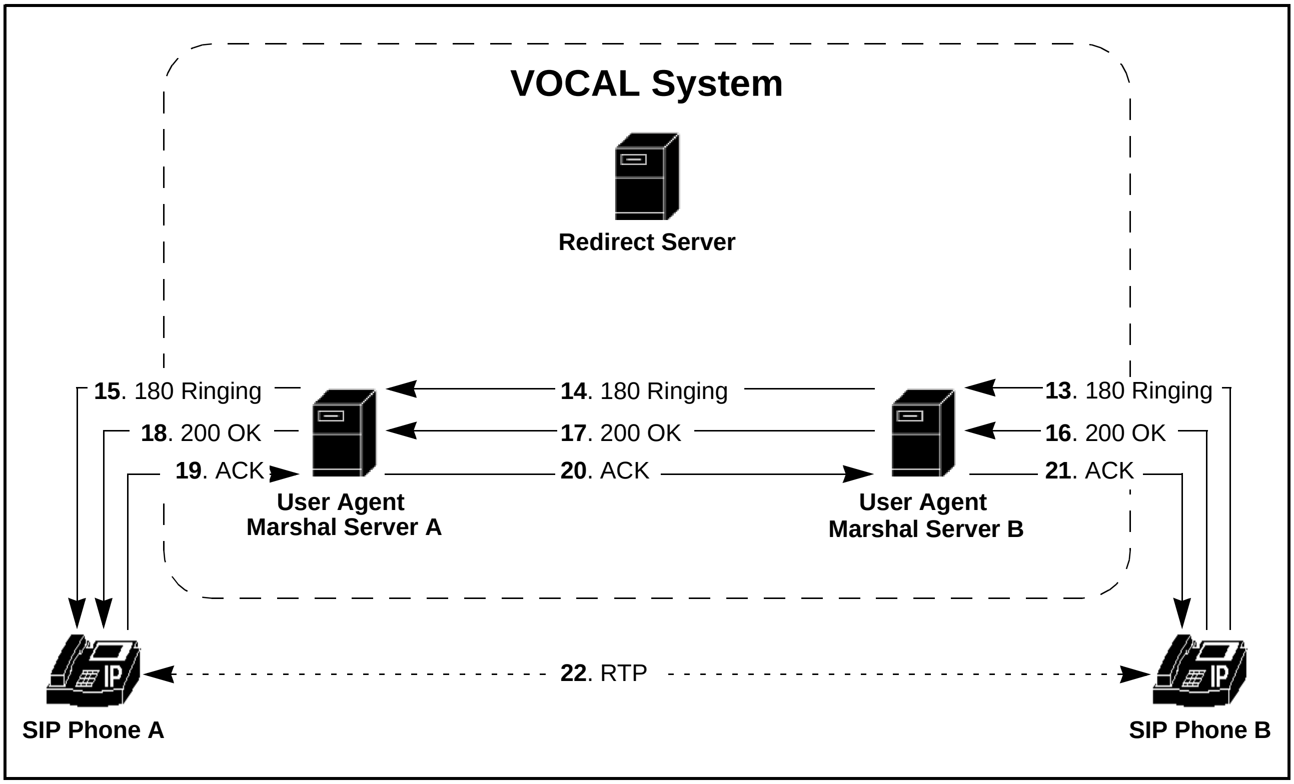


Figure 1-6. Call Establishment

Messages 10 - 19 Described

Table 1-10 describes the messages illustrated in Figure 1-6.

Table 1-10. Interactions Shown in Figure 1-6

Interaction	*Step*	*Description*
Ringing	***13 - 15***	SIP Phone B rings and sends a 180 Ringing response to UAMS B, which is forwarded through the network back to SIP Phone A. ■ **Note** The 180 does not pass through the RS because the RS does not request to be included in further messages from this call.

Table 1-10. (Continued) Interactions Shown in Figure 1-6

Interaction	*Step*	*Description*
Pick-up	***16 - 18***	SIP Phone B sends a 200 OK response to the UAMS. This means that the phone has been activated and is ready to establish voice channel contact with SIP Phone A.
Pick-up acknowledged	***19 - 21***	SIP Phone A sends an ACK message confirming that it is ready to connect to a voice channel.
A conversation takes place	***22***	A voice channel is established using Real-time Transfer Protocol (RTP), and the users can talk to each other. ■ **Note** It is also likely for the 180 Ringing message to contain session description information that permits a one-way audio path to be established from the called party to the calling party. This is known as early RTP. If early RTP is established, the return media path is setup after the called party has sent an ACK in response to the 200.

How is the Call Torn Down?

When the conversation is over, both phones hang up. The first phone to hang up sends a BYE message to the other. This BYE message tears down the RTP path for that phone. The other phone responds with 200, OK, and tears down its side of the RTP path.

Diagram #3: Call Tear Down

Figure 1-7 shows the call being torn down. In this example, SIP Phone B hangs up first.

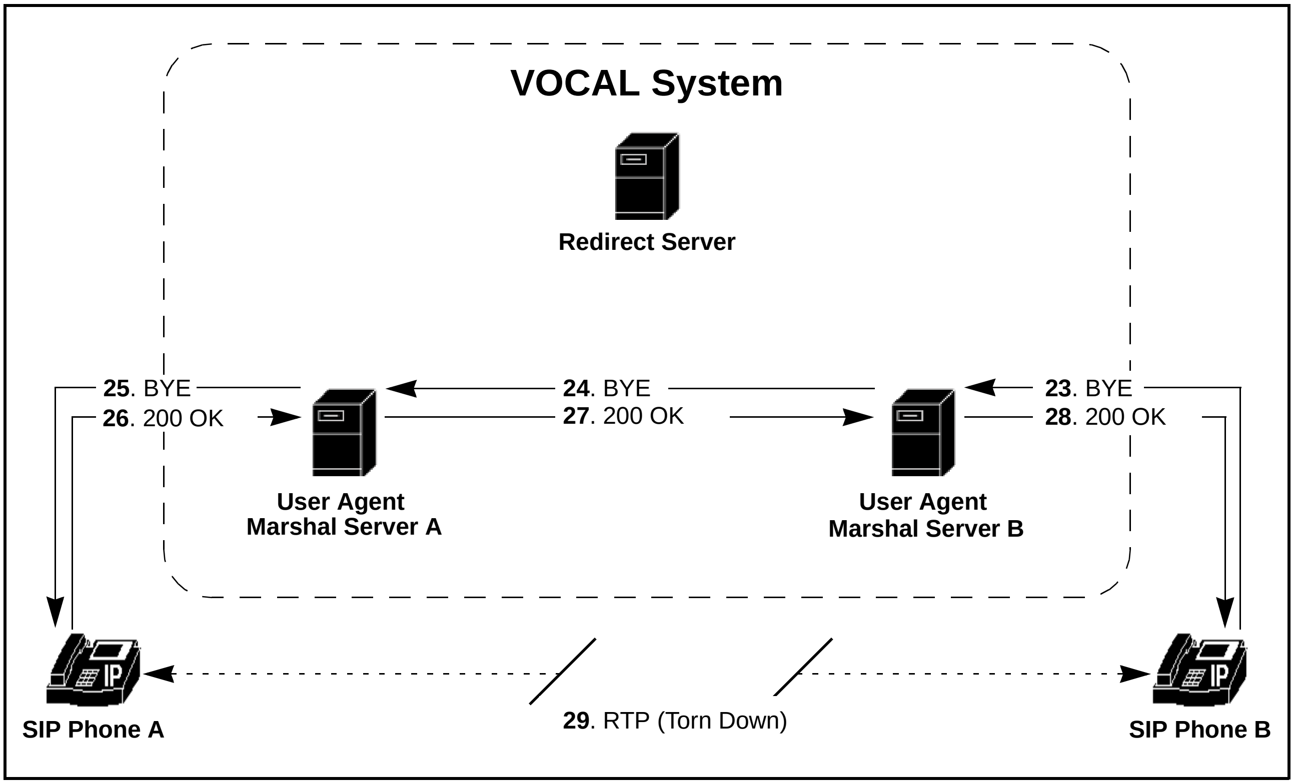


Figure 1-7. Call Tear Down

Messages 20 - 26 Described

Table 1-11 describes the messages illustrated in Figure 1-7.

Table 1-11. Interaction Shown in Figure 1-7

Interaction	*Step*	*Description*
SIP Phone B Hangs Up	*23 - 25*	SIP Phone B sends a BYE request through the system to SIP Phone A.
	26 - 28	SIP Phone A responds with a 200 OK.
The Voice Channel is torn down	*29*	The BYE and 200 messages trigger the voice channel to shut down.

Calling to Parties on the Public Switched Telephone Network

Introduction

This section describes how calls are routed to the public switched telephone network (PSTN).

Translating SIP into PSTN Signals

Connections to the PSTN are made through SIP-based PSTN gateways, which are attached to a Gateway Marshal Server, one Marshal server per gateway.

Diagram #1: SIP Phone to the PSTN

Figure 1-8 shows a message path from a SIP phone to the VOCAL system.

■ **Note**

In version 1.4.0 of VOCAL this is the only example where a Marshal server forwards an INVITE directly to another device without going through the Redirect Server for further routing information.

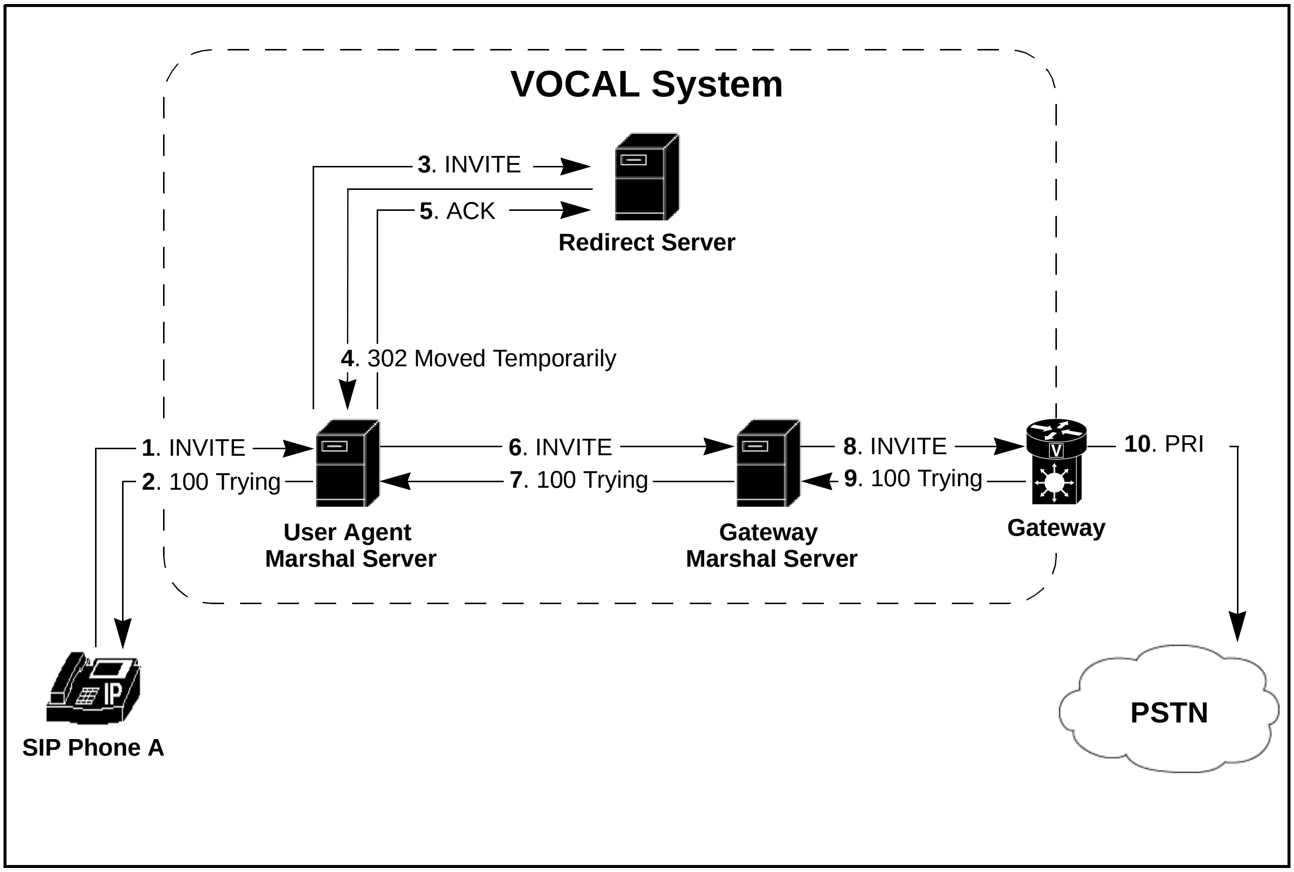


Figure 1-8. INVITE Message Sent to the VOCAL System

Messages 1 - 7 Described

Table 1-12 describes the messages illustrated in Figure 1-8.

Table 1-12. Interaction Shown in Figure 1-8

Interaction	*Step*	*Description*
SIP Phone A to the RS via the UAMS	*1 - 4*	SIP Phone A sends an INVITE message intended for a destination on the PSTN. The User Agent Marshal Server (UAMS) authenticates the message and forwards it to the Redirect Server (RS). The RS returns a 302 Moved Temporarily, message that provides routing information.
UAMS to the GWMS	*5 - 7*	The UAMS acknowledges receipt of the 302 message and forwards the INVITE to the Gateway Marshal Server (GWMS).

Table 1-12. Interaction Shown in Figure 1-8

Interaction	*Step*	*Description*
GWMS to the gateway and the PSTN	***8 - 10***	The GWMS forwards the INVITE to the SIP-based PSTN gateway, where it is translated into a format that is understood on the PSTN.

Call Routing Through a Feature Server

Introduction

This section describes how calls are routed through feature servers.

Features

The VOCAL system supports a variety of system and set features. For more information, see “Features” on page 1-42.

Call Routing to Feature Servers

As it has been shown above, the marshal servers forward INVITE messages to the Redirect Server for routing information. The INVITE message contains data describing its origin and intended destination. The Redirect Server looks up the origin and destination on a table that includes the dialing plan and system features, and then generates a Redirect message that includes routing information.

Example - Call Blocking

If the calling user agent has call blocking enabled, the Redirect server instructs the User Agent Marshal Server to forward the INVITE message to the Call Blocking Feature Server. The Call Blocking Feature Server looks up the call destination on its table of forbidden destinations. If the call matches a forbidden destination, the Call Blocking Feature Server disallows the call by sending a 403 Forbidden, message back to the User Agent Marshal Server, which forwards this message back to the calling user agent.

If the call destination does not match a forbidden destination, then the Call Blocking Feature Server forwards the call to the Redirect Server for further routing.

Diagram #1: Sending the INVITE

Figure 1-9 shows an INVITE message being sent to the VOCAL system.

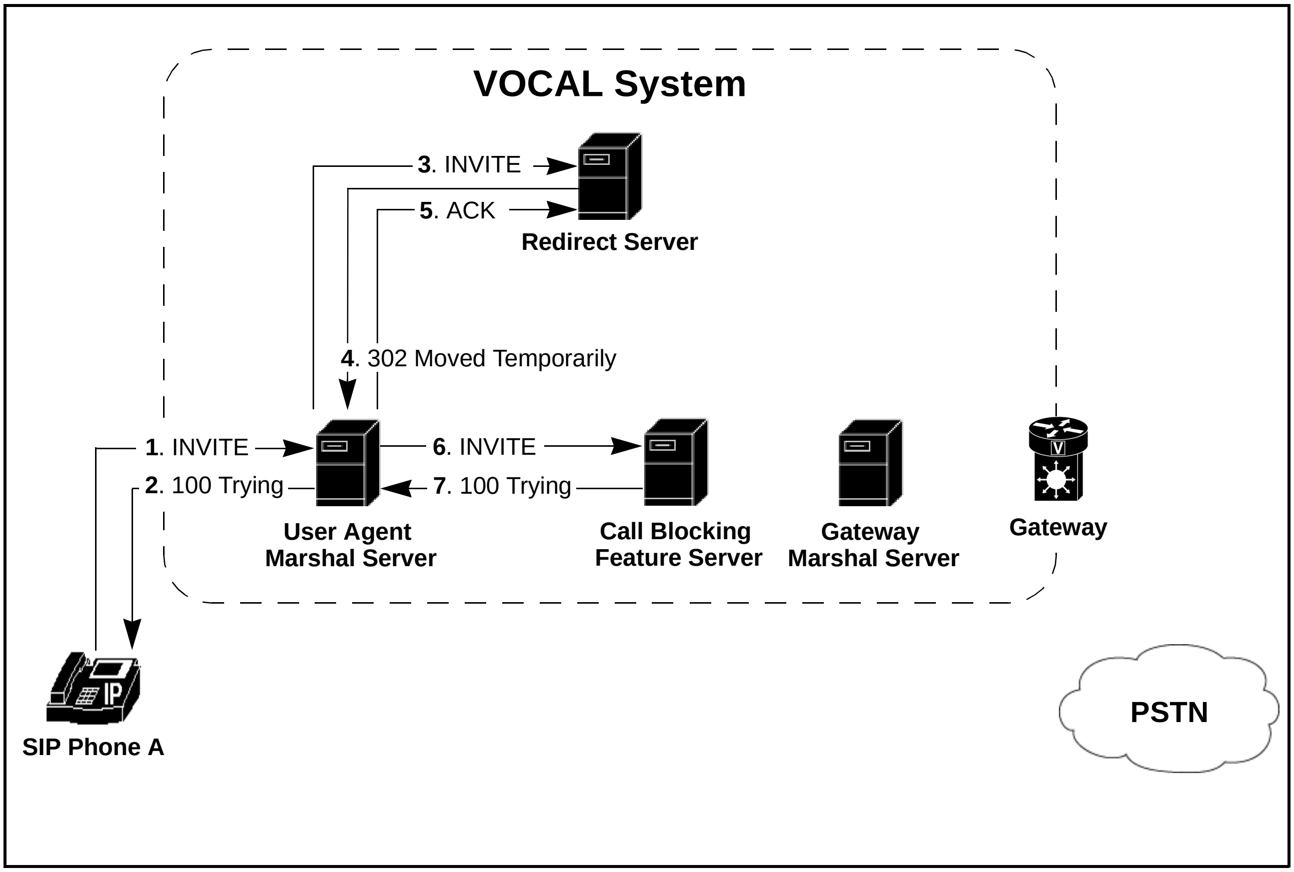


Figure 1-9. SIP Phone Sending an INVITE to the VOCAL System

Messages 1 - 5 Described

Table 1-13 describes the messages illustrated in Figure 1-9.

Table 1-13. Interaction shown in Figure 1-9

Interaction	*Step*	*Description*
SIP Phone A to the Redirect Server via the UAMS	*1 - 4*	SIP Phone A sends an INVITE message intended for a destination on the PSTN. The User Agent Marshal Server (UAMS) authenticates the message and forwards it to the Redirect Server (RS). The RS returns a 302 Moved Temporarily, message that provides routing information.
UAMS to the Call Blocking FS	*5 - 7*	The UAMS acknowledges receipt of the 302 message and forwards the INVITE to the Call Blocking Feature Server.

Diagram #2: Blocking the Call

Figure 1-10 shows the Call Blocking Feature Server preventing the call from going through to the PSTN.

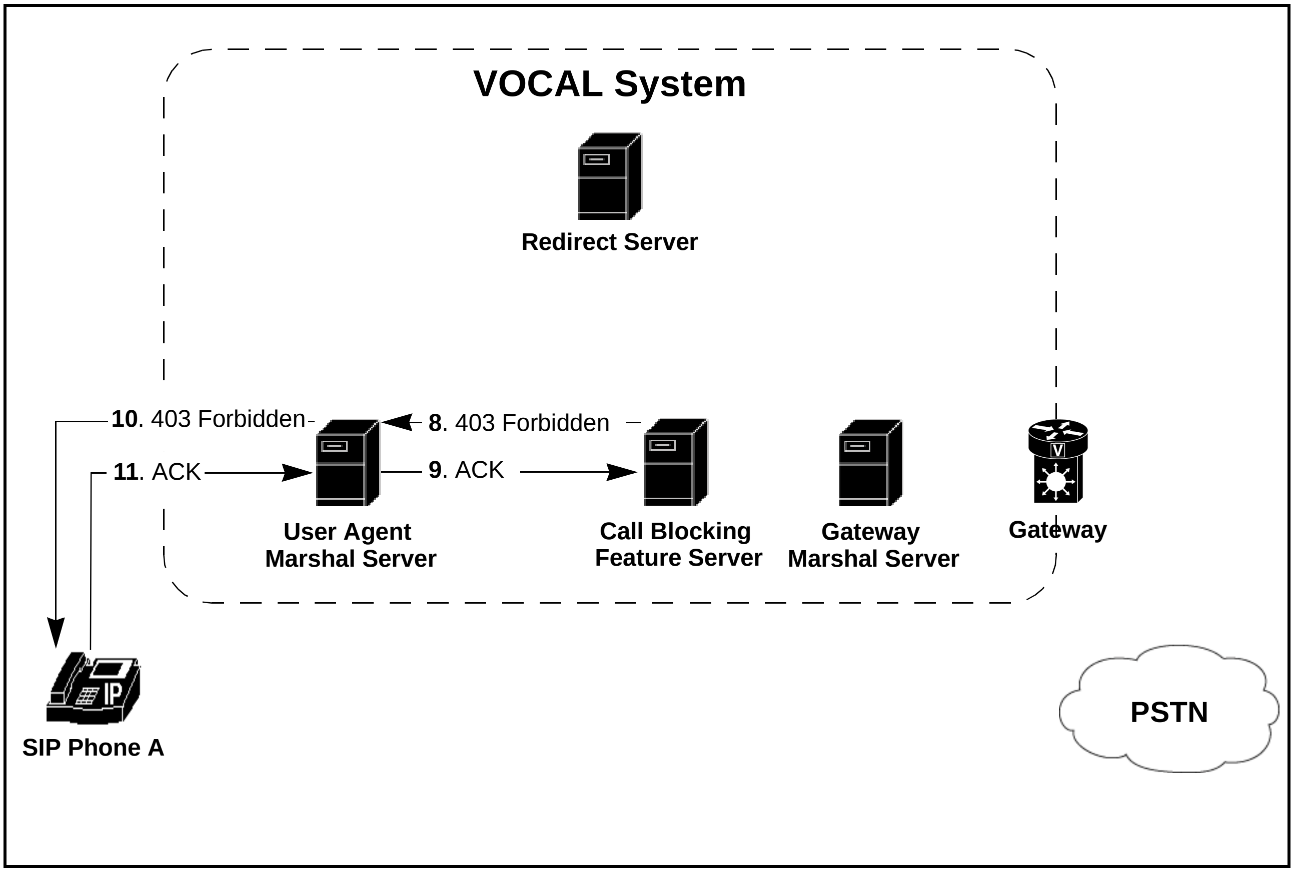


Figure 1-10. The Call Blocking Feature Server Blocks the Call

Messages 6- 8 Described

Table 1-14 describes the messages illustrated in Figure 1-10.

Table 1-14. Interaction shown in Figure 1-10

Interaction	*Step*	*Description*
The call is blocked	*8*	The Call Blocking Feature Server checks the calling destination against its list of forbidden destinations. The destination is forbidden, therefore, it sends a 403 Forbidden message back to the User Agent Marshal Server (UAMS).
	9 - 11	The UAMS acknowledges receipt of the 403 Forbidden message and forwards it to the SIP Phone, which acknowledges. The call has been blocked.

Quality of Service

Introduction

This section describes how VOCAL has implemented Quality of Service (QoS) for calls transmitted over the Internet.

What is QoS?

Quality of Service (QoS) is, in theory, an effort to manage transmission and error rates, and to minimize latency, packet loss and jitter during internetwork calls. The purpose of this effort is improve the quality of internetwork calls.

VOCAL does admission control based on resource availability. If resources cannot be allocated, VOCAL resorts to a “best effort only” delivery. Calls are still processed, but they may not be of the best quality.

What is Policy?

“Policy” is a broadly used, and widely interpreted term, that describes the business rules of the organization applied to the operation of its telecommunications systems. The term stems from the same source as “corporate policy” meaning the rules that guide the behavior of those who work for or with a corporation.

With respect to the practical application of QoS, policy is a combination of enforcement and decision making that permits calls to be initiated, established and torn down between networks over the Internet, or between managed IP networks. Enforcement and decision making are explained in detail in this section.

Function of the Policy Server

The Policy server is the key component used to achieve QoS. Service providers typically will only ensure QoS if authorizations and payments are guaranteed by a third party. The Policy server administers admission control for QoS requests and provides the Internetwork Marshal (policy client) with the information necessary to enforce the admitted QoS requests. The Policy server outsources the Authorization, Authentication and Accounting (AAA) requests to a third-party clearing house, which then acts as a trusted broker among a large number of network providers.

The Policy server supports two protocols, Common Open Policy Service (COPS) and Open Settlement Protocol (OSP). It acts as a COPS server when it communicates with the network routers, and acts as an OSP client when it exchanges authorization requests and usage reports with the clearinghouse server.

QoS Protocols

The QoS process works with the following protocols.

Table 1-15. Protocols Used with QoS

Protocol	*Description*
COPS	Common Open Policy Service Protocol (COPS) is a proposed Internet Engineering Task Force (IETF) standard for implementing QoS policies as an end-to-end service. It allows a Policy server to control devices on the network, such as routers and switches, whereby a consistent policy based on business priorities can be achieved. COPS is a companion protocol to Resource Reservation Protocol (RSVP).

Table 1-15. Protocols Used with QoS

Protocol	Description
RSVP	Resource Reservation Protocol (RSVP) allows paths on the Internet to be reserved so that voice conversations can be transmitted with minimal delays.

PEP's

Policy Enforcement Points (PEP's) can be routers, gateways and other devices that transfer voice channel signals between subscribers and their calling destinations. When the call is initiated, the PEP's query the Policy server for authorization to reserve bandwidth. Regardless of whether the bandwidth is available or not, VOCAL allows the call to go through.

When the call ends, the Policy server sends instructions to the PEP's to release the bandwidth.

PDP

The Policy Decision Point (PDP) is the Policy server. When the PEP's query the Policy server for authorization, the Policy server makes a Policy Decision to either accept or reject the request.

Quality of Service Enabled

Introduction

This section illustrates the messages exchanged to reserve bandwidth over the Internet, as well as the normal SIP messages used for call signaling.

Suggesting A Bandwidth Path

Figure 1-11 shows a request for bandwidth from User Agent B being processed through the networks. These signals are identified with letters, rather than numbers, because they are sent over the voice channel at roughly the same time that User Agent B sends a 180, Ringing, message, see Figure 1-14. Their sequence does not necessarily follow the sequence of the SIP messages shown in Figures 1-12 through 1-14.

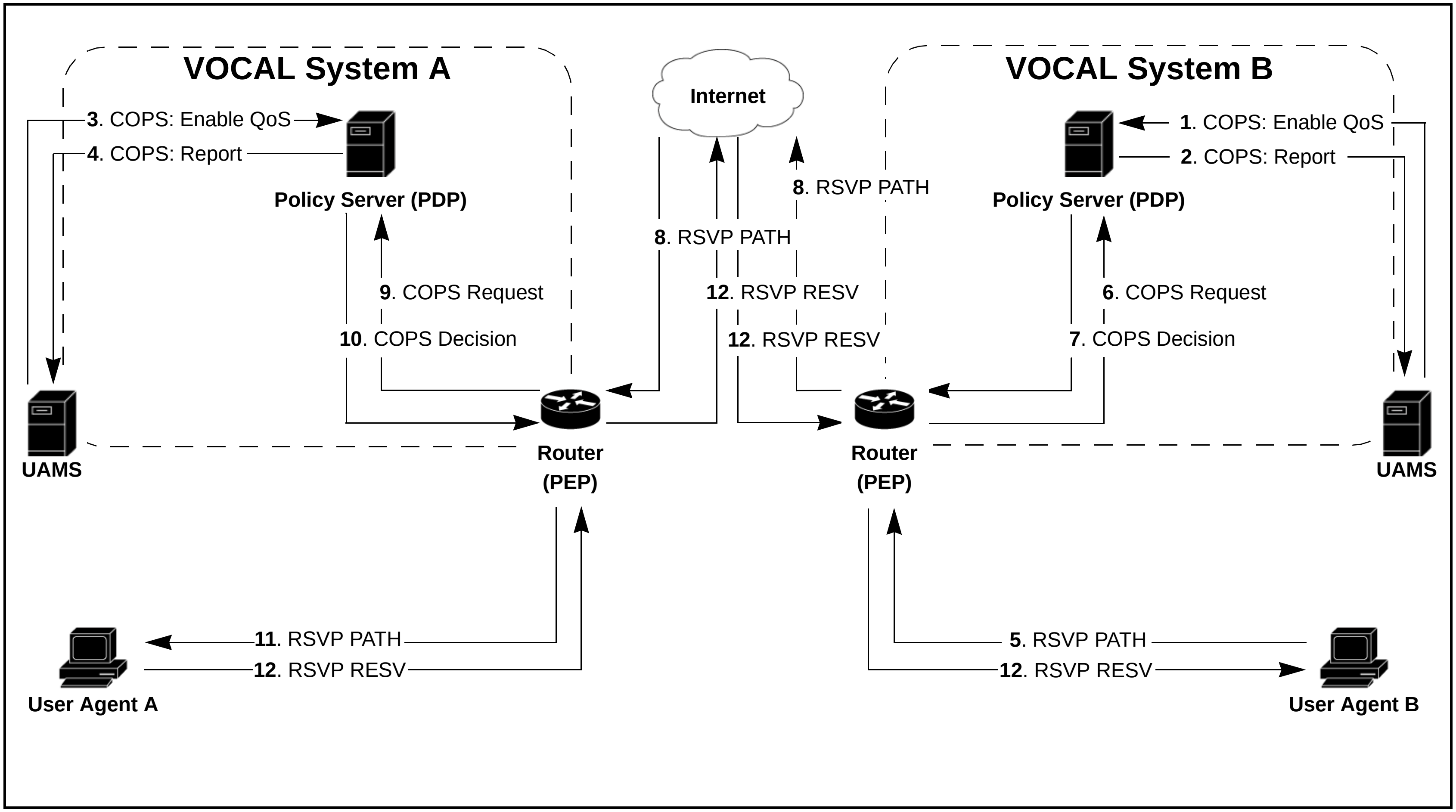


Figure 1-11. Interactions: Suggesting and Reserving a Bandwidth Path

Messages A - F Described

Table 1-16 describes the messages illustrated in Figure 1-11.

■ **Note**

These signals are identified with letters, rather than numbers, because they are sent over the voice channel at roughly the same time that User Agent B sends a 180, Ringing, message, see Figure 1-14. Their sequence does not necessarily follow the sequence of the SIP messages shown in Figures 1-12 through 1-14.

Table 1-16. Interactions Shown in Figure 1-11

Interaction	***Step***	***Description***
Enabling QoS	***1-4***	At the time that the UAMS receives either a 180 or 183 message from the called party, it sends a COPS message to the Policy Server (PoS) requesting it to establish QoS.
Requesting Bandwidth	***5***	User Agent B sends a RSVP PATH request to suggest a bandwidth path to the on-network router.
Requesting a Decision from the PoS	***6-7***	The router generates a COPS-RSVP request and sends it to the PoS, which responds with a COPS decision, authorizing the request.
Sending the Request to System A	***8***	The router sends the RSVP PATH request to the router in VOCAL System A.
Requesting a Decision from the PoS	***9-10***	The router generates a COPS-RSVP request and sends it to the PoS, which responds with a COPS decision, authorizing the request.
Sending the Request to User Agent A	***11***	The router sends the RSVP PATH request to User Agent A.
Sending an RSVP RESV	***12***	UA A sends a RSVP RESV message to UA B, reserving bandwidth.

Reverse Bandwidth Path

In order to establish an RTP path going the other way, the UAMS in VOCAL System A initiates the same process as illustrated in Figure 1-11 except in the opposite direction.

Open Settlement Protocol

Definition

Open Settlement Protocol (OSP) is a product of the Telecommunication and Internet Protocol Harmonization Over Networks (TIPHON) project at the European Telecommunications Standards Institute (ETSI: www.etsi.com), and is a specification for providing interdomain authentication, authorization, and accounting (AAA) standards for IP Telephony.

Diagram #1: Internetwork Calls From the calling party to the RS

Figure 1-12 shows User Agent (UA) A initiating a call to User Agent B. In this scenario, the UA's are used together with basic analog phone sets, and are attached to different VOCAL systems, and each VOCAL system is known to the other. The call signal routing is carried over the Internet.

The call may be routed through one or more feature servers before it reaches the Internetwork Marshal (INMS). For the sake of brevity, the feature servers have been omitted from this scenario.

Version 1.4.0 of VOCAL supports multiple INMS's. Each of these servers will accept off-network INVITE messages from one other known SIP-based server. If an INVITE is received from any other off-network entity, it will be rejected regardless if it includes a clearinghouse token or not.

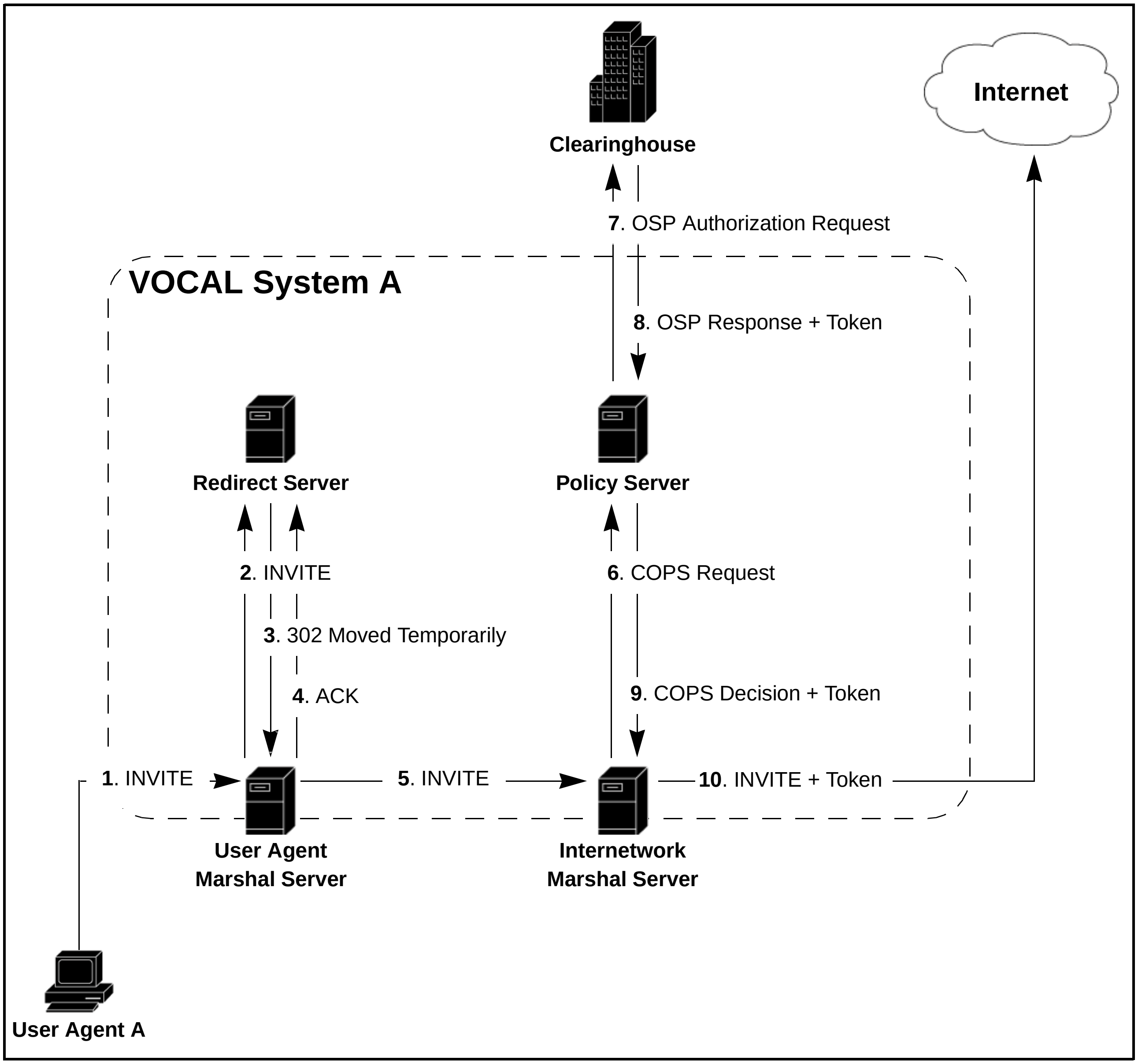


Figure 1-12. Transactions: Originating End

Messages 1 - 10 Described

Table 1-17 describes the messages illustrated in Figure 1-12.

Table 1-17. Interactions Shown in Figure 1-12

Interaction	***Step***	***Description***
SIP phone to INMS	***1- 3***	A call is initiated by an analog phone attached to User Agent A. The User Agent Marshal Server (UAMS) authenticates the message and forwards it to the Redirect Server (RS). The RS returns a 302 Moved Temporarily, message that provides routing information.
	4 - 5	The UAMS acknowledges receipt of the 302 message and forwards the INVITE to the Gateway Marshal Server (GWMS).
Requesting and Receiving an Internetwork Token from the Clearing House	***6***	The Internetwork Marshal Server (INMS) generates a COPS authorization request and sends it to the Policy Server.
	7	The Policy Server (PoS) composes an Open Settlement Protocol (OSP) authorization request and sends it to an internetwork clearinghouse and receives a response plus a token.
	8	The clearinghouse verifies the route, by confirming that the dialed digits are correct, and responds with an OSP Authorization plus a token.
	9	The PoS generates a COPS decision, which includes the clearing house's token, and sends it to the INMS.
INMS forwarding the INVITE Message Plus the Token	***10***	The INMS adds the token to the INVITE message and forwards it to the Internet via the router.

Diagram #3: The INVITE Message is Received

Figure 1-13 shows the receiving network processing the INVITE message and the token.

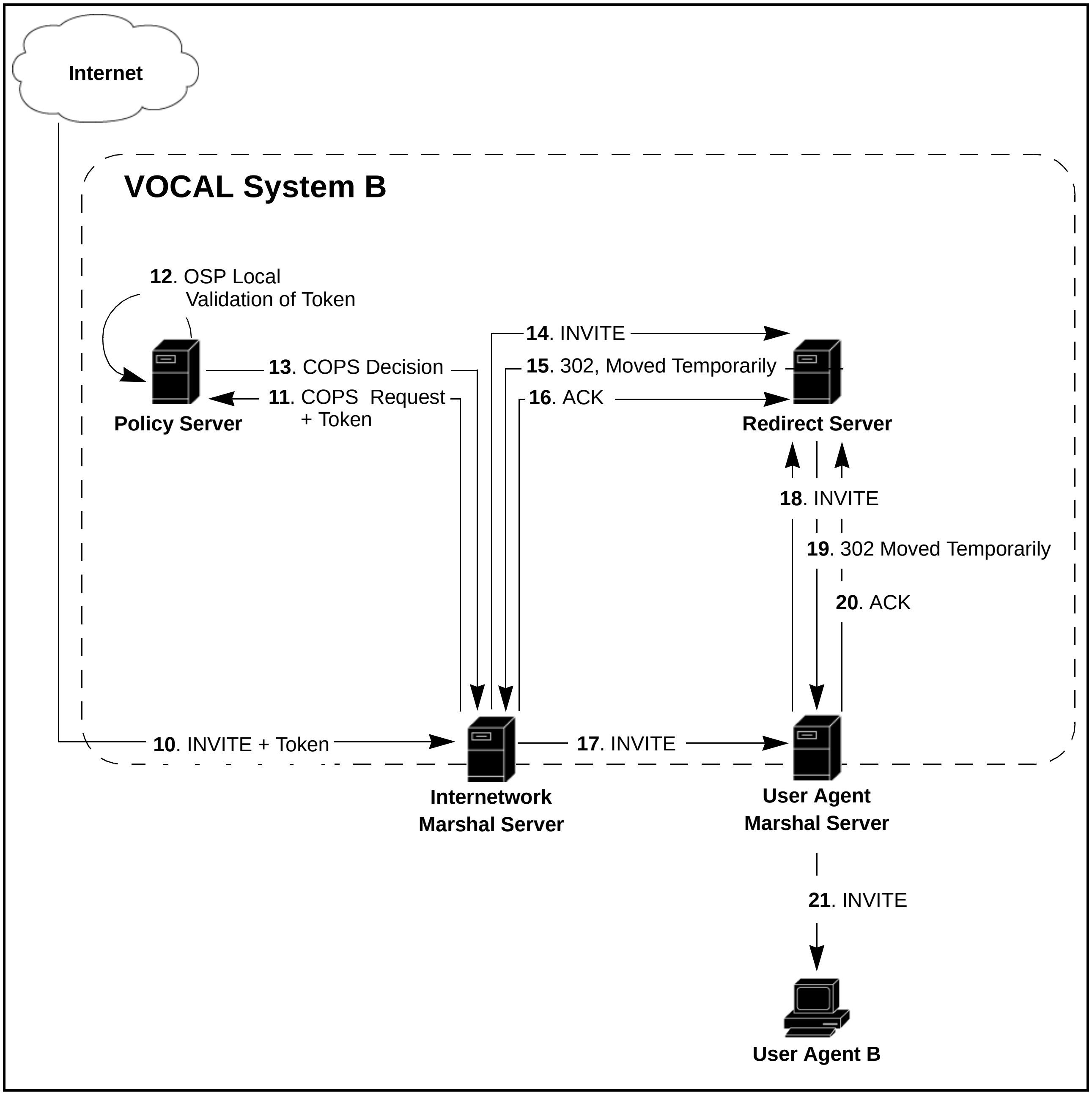


Figure 1-13. Transactions: Terminating End

Messages 10 - 21 Described

Table 1-18 describes the messages illustrated in Figure 1-13.

Table 1-18. Interactions Shown in Figure 1-13

Interaction	*Step*	*Description*
INVITE is received from known system	***10***	The INVITE message is received by the Internet Marshal Server (INMS).
The receiving INMS receives the INVITE and requests verification from the PoS	***11***	The INMS generates a COPS request and sends it, along with the token, to the Policy Server for verification.
	12	The Policy Server (PoS) verifies the token with its OSP client. The criteria for verification includes the source, the destination and the clearinghouse host name.
	13	The PoS strips the token from the message, generates a COPS Decision and sends it to the INMS.
The UAMS requests routing information from the RS.	***14 - 16***	The INMS strips the OSP token from the SIP INVITE header and forwards the INVITE message to the RS for routing. The RS returns a 302 Moved Temporarily and the INMS responds with an ACK message.
The INVITE message is sent to SIP Phone B.	***17 - 21***	The INMS forwards the INVITE to the UAMS, which forwards it, through the RS, to User Agent B.

Diagram #4: Establishing the Audio Path

Figure 1-14 shows the final series of SIP messages leading up the RTP audio path being established. Message 21, Ringing, is sent at roughly the same time that the RSVP PATH request is sent, see Figure 1-11.

Figure 1-14. Interactions: Establishing the Audio Path

Messages 22 - 25 Described

Table 1-19 describes the messages illustrated in Figure 1-14.

Table 1-19. Interactions Shown in Figure 1-14

Interaction	*Step*	*Description*
Ringing	***22***	SIP Phone B starts ringing and sends a 180 Ringing message to SIP Phone A.
OK	***23***	SIP Phone B sends a 200 OK message to SIP Phone A confirming that it is ready for establishing an audio path.
Acknowledge	***24***	SIP Phone A replies with an ACK message.
Audio Path	***25***	SIP Phone B is answered and an RTP audio path is established.

Operation System Support

Introduction

This section describes how the system is managed and supported.

What is OSS?

Operation System Support (OSS) includes methods that are used to monitor and maintain system performance. These methods include provisioning, authentication, billing and network management.

Provisioning

Introduction

Working with the Provisioning server is the subject of two chapters listed as hyperlinks below.

What is Provisioning?

Provisioning is a method for adding and maintaining network users. Users include servers, User Agents and subscribers.

Provisioning is divided into two interfaces, one for technicians and the other for system administrators. Each of these interfaces is a java based graphical user interface (GUI) that runs on a web browser.

Technician Interface

The technician interface works with maintaining the servers. This interface is described completely in Chapter 3, Provisioning.

System Administrator Interface

The system administrator interface works with maintaining the subscribers. This interface is described completely in the System Administration Guide.

Authentication

Introduction

The Marshal servers authenticate every message that they receive. This section explains how.

What is Authentication?

Authentication is the process that protects the system from unauthorized users. The marshal servers authenticate each call by checking the calling party's IP address against a master file. If the marshal server does not have the calling party's address on its list, it requests verification from the Provisioning server. If the Provisioning server does not verify the address, the marshal refuses to authenticate the call. The authentication method can be either Access List or Digest.

Access List

Overview

If the User Agent Marshal Server authenticates the user agent, it forwards the user agent's message through to the Redirect Server. If the message is REGISTER, the Redirect server registers the user agent, and returns a confirmation message back through the User Agent Marshal Server to the user agent.

Diagram: calling party Authentication

Figure 1-15 shows a SIP phone registering with the Redirect Server.

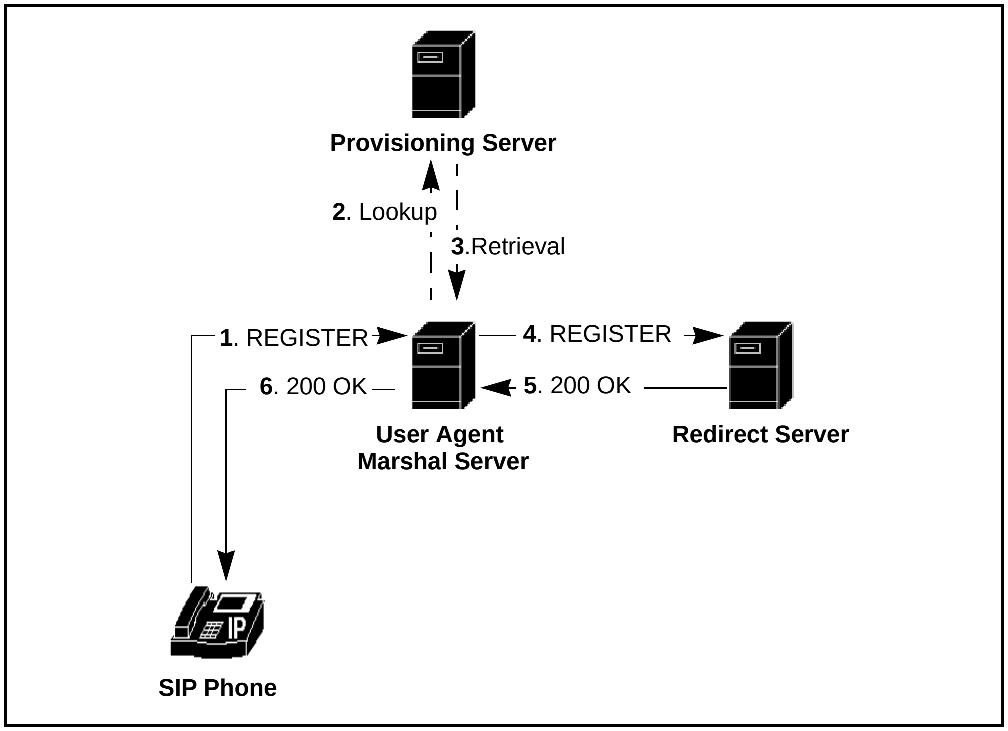


Figure 1-15. Calling Party Registration: Access List

Messages 1 - 6 Described

Table 1-20 describes the messages illustrated in Figure 1-15.

Table 1-20. Interactions Shown in Figure 1-15

Interaction	*Step*	*Description*
SIP Phone to UAMS	*1*	The SIP Phone is connected to the network and immediately sends a REGISTER message to the User Agent Marshal Server (UAMS).
UAMS to PS	*2 - 3*	The UAMS does not have a record of the SIP Phone's IP address in its database and it retrieves data from the Provisioning Server (PS) to validate the request. The UAMS adds the SIP phone to its list of authorized users.
UAMS to RS	*4*	The UAMS forwards the REGISTER message to the Redirect Server (RS).

Table 1-20. Interactions Shown in Figure 1-15 (Continued)

Interaction	*Step*	*Description*
OK Returned	***5 - 6***	The UAMS forwards the OK to the SIP Phone. The phone is registered in the system, and it will re-register every few minutes.

Digest

Overview

The Digest method for authentication and registration uses a password. This password is maintained in the Provisioning Server's database.

Diagram: calling party Authentication

Figure 1-16 shows a SIP phone registering with the Redirect Server.

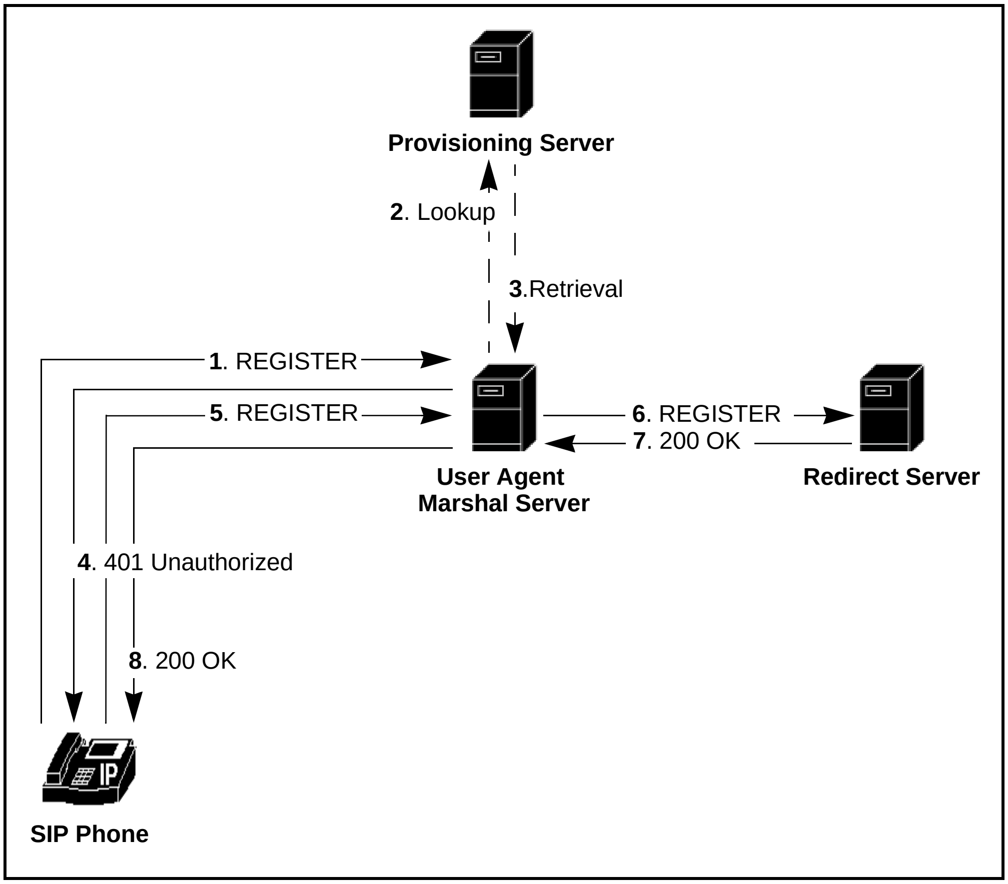


Figure 1-16. calling party Registration: Digest

Messages 1 - 8 Described

Table 1-21 describes the messages illustrated in Figure 1-16.

Table 1-21. Interactions Shown in Figure 1-16

Interaction	*Step*	*Description*
SIP Phone to UAMS	***1***	The SIP Phone is connected to the network and immediately sends a REGISTER message to the User Agent Marshal Server (UAMS).
UAMS to PS	***2 - 3***	The UAMS does not have a record of the SIP Phone's IP address in its database and it retrieves data from the Provisioning Server (PS) to validate the request. The UAMS adds the SIP phone to its list of authorized users.
Unauthorized	***4***	The UAMS returns a 401 Unauthorized message to the SIP Phone requesting a password.
New REGISTER message	***5***	The SIP sends a new REGISTER message that includes a password.
UAMS to RS	***6***	The UAMS authenticates the calling party and forwards the REGISTER to the RS.
OK Returned	***7 - 8***	The RS replies with a 200 OK message, which is forwarded to the SIP phone by the UAMS. The phone is registered in the system, and it will re-register every few minutes.

Call Detail Records and Billing

Introduction

The VOCAL system has a Call Detail Record (CDR) server that receives time-stamped information about every processed call. This information can be forwarded to a third-party billing system by using a Remote Authentication Dial In User Service (RADIUS) stack.

Call Detail Records

How Does the CDR Server Receive its Data?

The CDR server communicates with the marshal servers over TCP/IP. As it has been shown above under "Call Control" on page 1-16, every call involves both incoming and outgoing marshal servers. At the time when a call starts and again when it ends, both marshal servers notify the CDR server. From this notification, the CDR server creates a new billing file, called billing.dat, with two Start and two End records, one of each from both marshal servers.

What Defines the Start of a Call?

In a conventional setup, the start of a call happens when the voice channel is established. After the INVITE has been transmitted from the calling party to the called party, and the called party starts ringing, the called party picks up and thereby, transmits a 200, OK, message to the calling party. When the calling party replies with an ACK message, the marshal servers notify the CDR server to create a START record.

■ **Note**

You can provision the CDR server to bill for ring time. If you do, the marshals notify the CDR server to create a start record when they receive the 180, Ringing, message from the called party.

Notifying CDR Server for START Record

Figure 1-17 shows the SIP messages that lead up to the marshal servers notifying the CDR server to create a START record. In this scenario, the INVITE has already passed from SIP Phone A to SIP Phone B, as shown above in Figure 1-5.

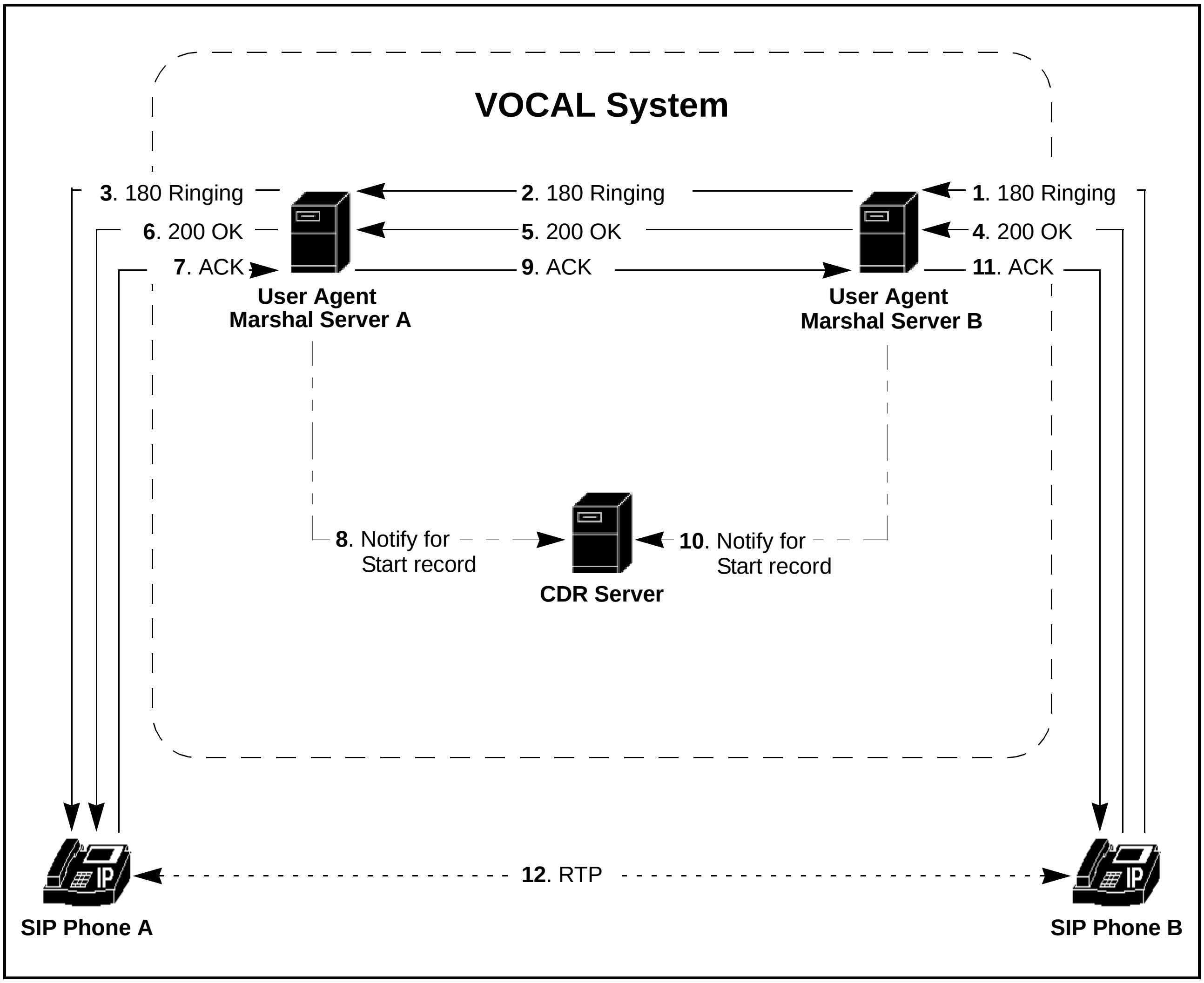


Figure 1-17. Notifying the CDR Server for the Start Record

Messages 1-12 Described

Table 1-22 describes the message illustrated in Figure 1-17.

■ **Note**

Message #1 in Figure 1-17 occurs after the INVITE has been passed from SIP Phone A, through the system to SIP Phone B. For more information, see Figure 1-5 and Table 1-9.

Table 1-22. Interactions Shown in Figure 1-17

Interaction	***Step***	***Description***
Ringing	***1 - 3***	SIP Phone B sends a 180 Ringing response to User Agent Marshal Server (UAMS) B, which is forwarded.
Pick-up	***4 - 6***	SIP Phone B sends a 200 OK response to the UAMS. This means that the phone has been activated and is ready to establish voice channel contact with SIP Phone A.
Pick-up acknowledged	***7***	SIP Phone A sends an ACK message to UAMS A confirming that it is ready to connect to a voice channel.
CDRS notified	***8***	UAMS A notifies the Call Detail Record Server (CDRS) that the call has started.
Pick-up acknowledged	***9***	UAMS A forwards the ACK message to UAMS B.
CDRS notified	***10***	UAMS B notifies the CDRS that the call has started.
Pick-up acknowledged	***11***	UAMS B forwards the ACK message to SIP Phone B.
A conversation takes place	***12***	The calling parties talk to each other using Real-time Transfer Protocol (RTP).

What Defines the End of a Call?

The end of a call happens when the first phone hangs up and, thereby, sends a BYE message to the other phone. Upon receiving the BYE message each marshal server notifies the CDR server to create an End record. This process is illustrated in Figure 1-18.

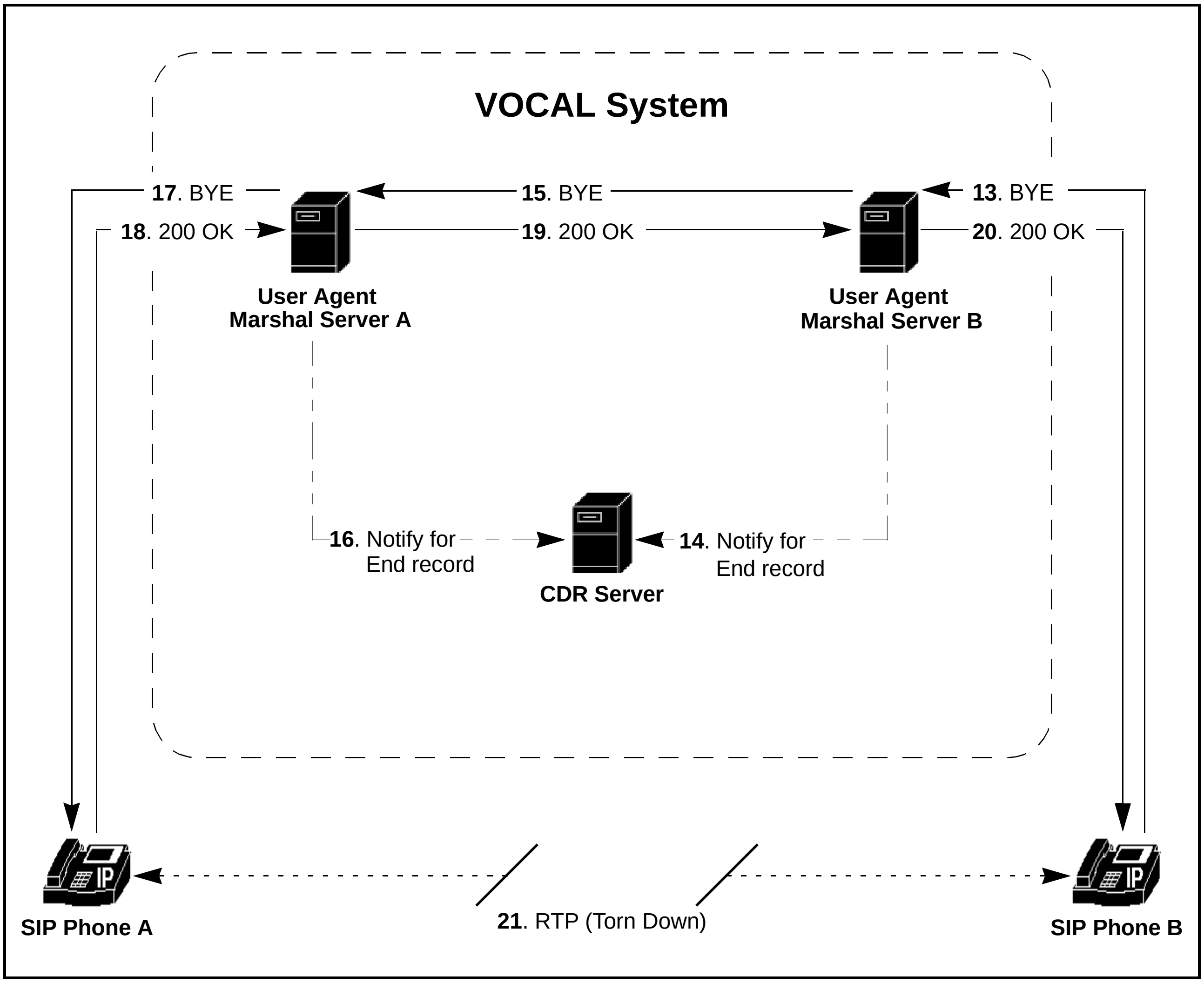


Figure 1-18. Notifying the CDR Server for the End Record

Messages 13-21 Described

Table 1-23 describes the message illustrated in Figure 1-18.

Table 1-23. Interactions Shown in Figure 1-18

Interaction	*Step*	*Description*
SIP Phone B Hangs Up	*13*	SIP Phone B sends a BYE request to User Agent Marshal Server (UAMS) B.
Notifying the CDRS	*14*	UAMS B notifies the Call Detail Record Server (CDRS) that the call has ended.

Table 1-23. Interactions Shown in Figure 1-18 (Continued)

Interaction	Step	Description
The BYE is forwarded from one UAMS to the other.	**15**	UAMS B forwards the BYE message to UAMS A.
Notifying the CDRS	**16**	UAMS A notifies the CDRS that the call has ended.
The BYE is received by SIP Phone A.	**17**	UAMS A forwards the BYE message to SIP Phone A.
SIP Phone A Hangs Up	**18 - 20**	SIP Phone A sends an ACK message, through the UAMSs, to SIP Phone B.
The Voice Channel is torn down	**21**	The BYE and ACK messages trigger the voice channel to shut down.

Billing

Introduction

The section above shows how the CDR Server collects data about each call. This data collection occurs regardless if there is a third-party billing system attached to the network or not. This section explains how the call detail records can be used to generate billing.

The Bill Record

After receiving notifications from the marshal servers about the start and end of each call, the CDR Server generates a Bill record that contains a duration field. This field is the calculated difference between the start and end times of the call. These Bill records can be sent from the CDR server to a third-party billing system on a regular schedule using RADIUS messaging over UDP.

New billing files, that have not been sent to the billing server, have a .unsent extension appended to their file name. Billing files are normally purged from the CDR Server after 72 hours.

Sending Records with RADIUS

The CDR server uses a RADIUS stack to communicate with the billing system. In order to send records to the billing system, you must know which Vendor Specific Attributes are being used in the billing system's code, and modify the CDR server to accept those attributes. For more information, go to the www.vovida.org web site.

The billing data is not sent during real time. You must set up a transmission schedule for hourly, daily or for the frequency that best suits your needs. For more information about setting up the CDR Server and the billing transmission frequency, see "CDR Servers, Data Entry Fields" on page 3-40.

Network Management

Introduction

Network management is the subject of a chapter in the System Administration guide. This section provides a brief description and a hyperlink to the chapter.

How Does VOCAL Provide Network Management

The VOCAL system uses Simple Network Management Protocol (SNMP) to control and monitor system processes. VOCAL provides a java-enabled GUI for the System Administrator to help him or her manage the system.

Network Management is described in more detail in the System Administration Guide.

Features

Introduction

Some features are provided by the VOCAL system regardless of the types of phones used by the calling parties. Other features are only available on certain IP phone sets. This section explains the differences between these types of features.

What are Features?

Features are the enhanced functions of the phone system that enable customers to do more than simply make and receive phone calls. Features are referred to as being either Core System or Set-based.

Core System features are those that a provided by the VOCAL system. Some of these features are built-in to the SIP messaging such as, Calling Line Identification.

Set-based features are those which are dependent on the design of the phone set such as, Transfer.

What is CPL?

The Call Processing Language (CPL) is used to describe and control Internet telephony services that are implemented on either network servers or user agent servers. CPL scripts are normally simple, extensible, and easy to edit. For more information about CPL see http://www.ietf.org/internet-drafts/draft-ietf-iptel-cpl-04.txt.

SIP Messages and Feature Servers

Introduction

This section illustrates how the VOCAL system routes calls to feature servers by using SIP messages. When the feature servers first come on-line, they download a register from the Provisioning Server but they do not download the file that controls the feature. It is not until the first time that the Feature Server runs a script, that it downloads the controlling file from the Provisioning Server.

Diagram #1: SIP Messages to the Feature Servers

Figure 1-19 shows a Feature server receiving a message from a Marshal and then requesting routing information from the Redirect server. It is possible that a call signal may be routed to several Feature servers before leaving the VOCAL system. Some calls may not be routed to any Feature servers before going to the outbound Marshal.

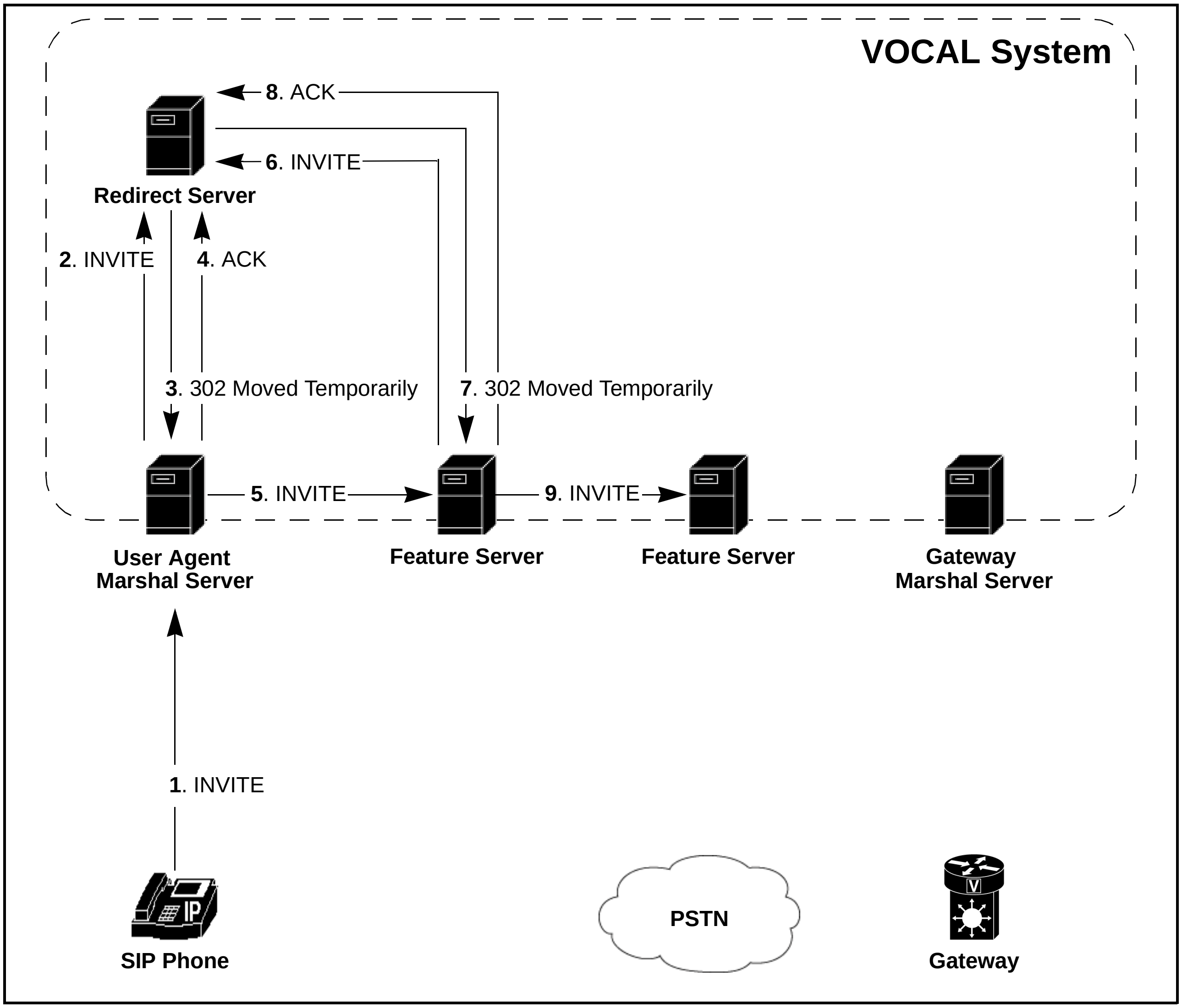


Figure 1-19. SIP Message Flow to the Feature Servers

Messages 1 - 7 Described

Table 1-24 describes the messages illustrated in Figure 1-19.

Table 1-24. Interaction shown in Figure 1-19

Interaction	***Step***	***Description***
SIP phone to UAMS	***1***	A call is initiated at one SIP phone to call a party attached to the PSTN. The SIP phone sends an INVITE message to the User Agent Marshal Server (UAMS).
UAMS to RS	***2 - 4***	The UAMS authenticates the user and forwards the INVITE message to the Redirect Server (RS). The RS looks up the contact information for the calling user, which includes the Call Blocking and Calling Party ID Blocking features. The called party destination is on the PSTN, therefore the RS has no contact list for the called party, but through its dial plan, the RS can provide routing information. The RS writes the calling party feature and called party routing information to a 302 message and sends it to the Marshal server, which completes the transaction with an ACK message. ■ **Note** The RS does not determine whether the called number is on the user's call blocking list. As long as the Call Blocking feature is assigned to this user, the RS will send every call from that user through the Call Blocking Feature server.
UAMS to a feature server via the RS.	***5***	The UAMS generates a new INVITE message and sends it to the Call Blocking Feature Server (FS).
The message is redirected to a second feature server	***6 - 8***	The Call Blocking FS generates a new INVITE message and sends it to the RS. As it did earlier, the RS looks up the calling party's contact list and the called party's routing information. From the information provided in INVITE message, the RS knows that the message has come from the Call Blocking Feature server and provides routing information that will direct it towards the Calling Party ID Blocking FS. The RS writes this information to the 302, and sends it to the Call Call Blocking FS, which responds with an ACK to complete the transaction.
Feature server to Feature server	***9***	The Call Blocking FS generates a new INVITE message and sends it to the Calling Party ID Blocking FS.

Diagram #2: Feature Servers to PSTN

Figure 1-20 shows the INVITE message being redirected from the feature servers to the PSTN.

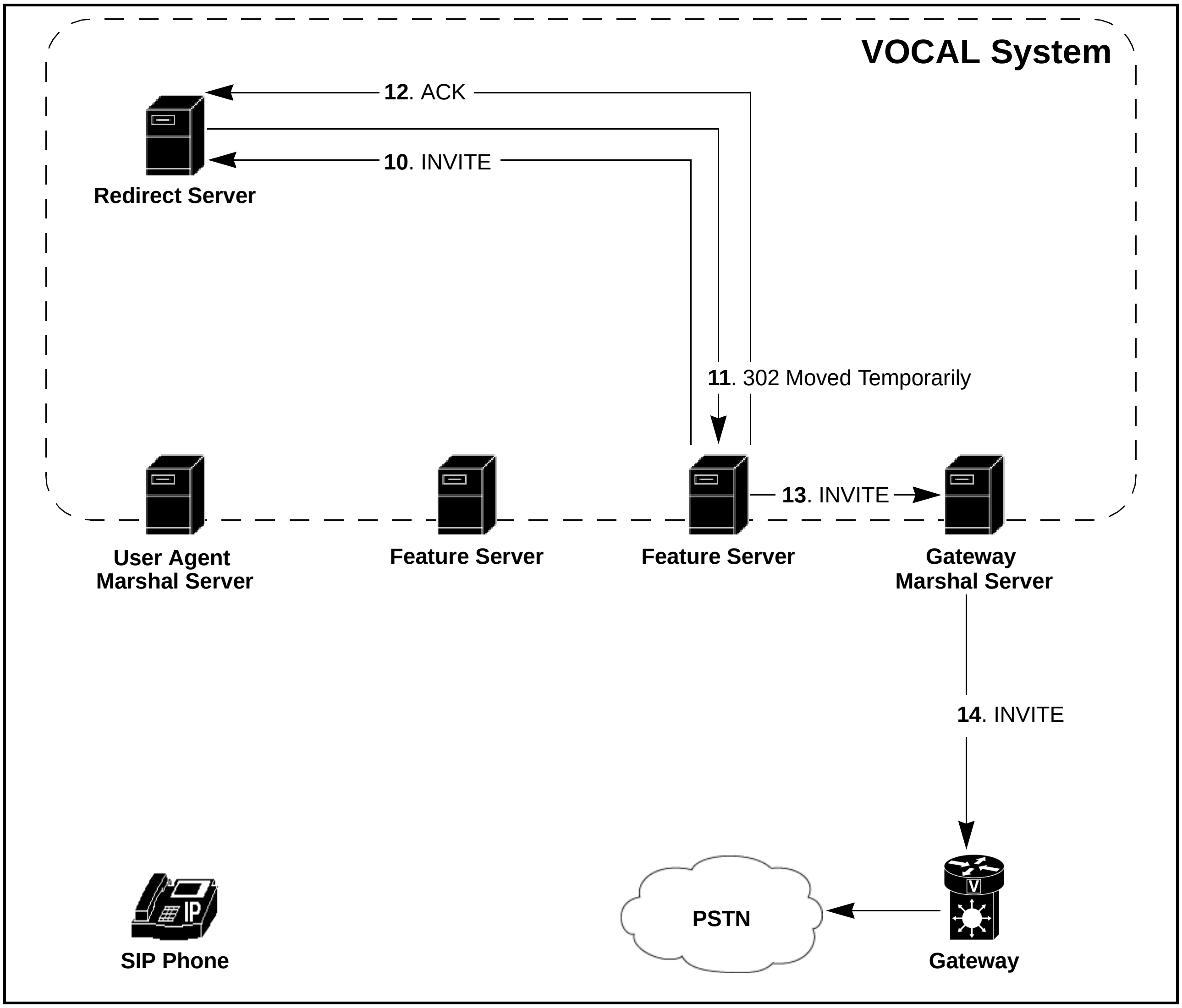


Figure 1-20. Feature Servers to PSTN

Messages 8 - 12 Described

Table 1-25 describes the messages illustrated in Figure 1-20.

Table 1-25. Interaction shown in Figure 1-20

Interaction	***Step***	***Description***
Feature server to GWMS	***10 - 12***	The Calling Party ID Blocking Feature Server (FS) generates a new INVITE message with calling party ID blocking instructions, and sends it to the Redirect Server (RS) for routing. The RS, once again, calls up the contact list for the calling party and the routing for the called party. The INVITE contains information telling the RS that the message has been routed through both Feature servers listed on the calling party's contact list. As the called party is on the PSTN, the RS does not have its contact list and therefore writes instructions to the 302 message to route the call to the Gateway Marshal server. This 302 is sent to the Calling Party ID Blocking Feature Server, which returns an ACK to complete the transaction. ■**Note** If the called party had been a subscriber to this system, the RS would have been able to call up its contact list and would have sent the message through the listed Feature servers before sending it to the appropriate Marshal server.
GWMS out to the PSTN	***13***	The GWMS forwards the INVITE to the Gateway Marshal server, which forwards it to the gateway. ■**Note** In version 1.4.0 of VOCAL, the GWMS does not forward INVITE messages back to the RS for final routing. This step was removed to speed up the call processing.
	14	The Gateway translates the message into a signaling format that is used on the PSTN and sends it out to the called party.

Core Features

What's a Core Feature?

Core features are network features that operate independently of the User Agent appliance used by the customer. These features include calling line information features, call forwarding, call blocking and call screening.

Calling and Called Features

Calling features are assigned to the call originator, and include Call Blocking, calling party ID Blocking and others. Called features are assigned to the calling destination, and they include Call Screening, Call Forward and others.

Calling Line Information Features

Calling Number Delivery (CND)

A calling feature: Calling Number Deliver (also known as Calling Line Identification (CLID)) provides information to the line about where the call is to be terminated, the Directory Number where the call was originated as well as the date and time of the call.

Calling Name Delivery (CNAM)

A calling feature: Calling Name Delivery (also known as Calling Party Name Delivery (CPND)) provides information to the line about where the call is to be terminated, the calling party's name as well as the date and time of the call.

Calling Party Identity Blocking (CIDB)

A calling feature: calling party ID Blocking allows a subscriber to control whether or not their number (CND) or name (CNAM) is delivered when they place an outgoing call.

Call Forwarding

Call Forward All Calls (CFA)

A called feature: Call Forward – All Calls allows a customer to re-route all calls to an alternative number. When CFA is activated, a call to the listed number is redirected to a user selected alternative number or a voice messaging system.

Call Forward – No Answer Mode (CFNA)

A called feature: Call Forward – No Answer Mode allows a customer to specify where an unanswered call should be routed. When CFNA is activated, a call to the listed number, that does not answer in a specified number of ringing cycles, will be forwarded to an alternative number selected by the user.

Call Forward – Busy Mode (CFB)

A called feature: Call Forward – Busy Mode allows a customer to specify where a call should be routed when the listed number is in use. When CFB is activated, a call to the listed number, while it is in use, will be redirected to another number.

Call Blocking

A calling feature: It prevents the customer from establishing connections to specified parties such as, 1-900 numbers.

■ **Note**

For version 1.4.0 of VOCAL, long-distance call blocking only works for calls originating from the North American Numbering Plan (NANP). Calls cannot be blocked if they originate from Europe, Asia or other locations that are not part of the NANP. For more information, see www.nanpa.com.

Call Screening

A called feature: It prevents incoming calls from specified parties to establish connections with the customer.

■ **Note**

For version 1.4.0 of VOCAL, phone numbers entered for call screening must include the area code, regardless if they are local or long-distance phone numbers. Call Processing Language does not provide a pattern matching method that differentiates seven digit (local) phone numbers from ten digit (long-distance) numbers.

Set Features

What are Set Features?

Set features are features that depend on the User Agent appliance. The VOCAL system supports transfer, call return and call waiting features.

Transfer

Call Transfer allows a user, on any existing two-party call, to place the existing call on hold and originate another call to a third party. The user may consult privately or connect the original call to the third party.

Call Return

Call Return allows the subscriber to place a call back to the last number that called him or her by dialing a special feature code.

■ **Note**

Call Return can be either a Core System or a Set-based feature.

Call Waiting (CW)

Call Waiting notifies a telephone user, who is on an established call, that an additional external call has been presented and is “waiting to be answered”. The waiting call receives normal ringing until it is answered, the incoming calling party abandons the call, or the ringing cycle timer expires, and the call is given Call Forward-No Answer treatment (if applicable). Only one Call Waiting call can be present at a time. Additional calls that may be presented are provided with Busy Mode treatment (CFB, if applicable).

■ **Note**

Implementation of Call Waiting requires support from the phone sets.

Cancel Call Waiting (CCW)

Otherwise known as “Do Not Disturb”, Cancel Call Waiting allows the subscriber to dial a feature activation code prior to making a call. For the duration of the subsequent call, the Call Waiting feature will be disabled for that line. The Cancel Call Waiting feature lasts only for the duration of one call and, when the subscriber goes on-hook again, their Call Waiting feature is re-enabled.

Scriptable Feature Development

What are Scriptable Features?

Scriptable features are features that can be expressed in a scripting language such as, Call Processing Language (CPL). The VOCAL system supports new feature development through CPL scripts.

For more information, see the System Administration Guide.

2

Software Installation

Chapter Content

This chapter explains how to acquire and load the VOCAL system software into a single server or a network of host machines.

Topic	See Page

Topic *(continued)*	**See Page**

Before You Begin

Introduction

This section describes:

- hardware requirements for each host machine
- software requirements for each host machine
- network requirements

■**Note**

Version 1.4.0 of VOCAL supports several different platforms including Linux and Solaris. This document provides information about installing onto Linux and Solaris only. While there are some differences in the requirements between these platforms, the deploy scripts run exactly the same way, with the same prompts and results in both platforms.

Install with or without Java

VOCAL version 1.4.0 provides the option to install and run the system without Java. Doing so disables the Java Provisioning GUI described later in this guide and in the System Administration Guide. Your choice is as follows:

1) Install VOCAL without using the JRE or JDK.

Considerations:

- The installation process is more simple than it was for previous versions of VOCAL.
- A simple HTTP user provisioning GUI is available
- There is no GUI for editing the the servers' provisioning .

2) Install VOCAL with the JRE or JDK

Considerations:

- The installation process is the same as it was for previous versions of VOCAL (version 1.3.0 and later).
- You can use the full Java Provisioning GUI as described in the user guides.

Hardware Requirements

Introduction

The VOCAL system is a distributed network that may be hosted on any number of machines. Systems may range in size from a demonstration system hosted on a single machine, to a large-scale network hosted on as many machines as required to meet the needs of the organization. For more information about the number of host machines required to support your subscriber base, see Appendix A, Engineering Guidelines.

Linux Hosts

Host Machine Requirements

The following is a list of recommended attributes for machines hosting a VOCAL system on the Linux operating system:

- 480 MHz, Intel Pentium II PC processor
- 128 MB RAM
- 1 GB of hard disk space
- The Feature servers require 10 kilobytes of RAM memory per provisioned user.

Sun Solaris Hosts

Solaris Host Machine Requirements

We have tested version 1.4.0 of VOCAL on the following rack mounted servers.

Single Processor

- NETRA T1
- 440 MHZ Ultra-Sparc-II
- 512 MB RAM
- 18 GB HARD DISK

Software Requirements

Introduction

Before the VOCAL system can be installed onto a host machine, the following software must be installed *and* running:

- Linux, Red Hat Versions 6.2, 7.1 or 7.2 or Sun Solaris 2.8
- Apache Server

For Java Provisioning Only

- Netscape web browser Version 4.6 or higher
- Java Run-time Environment (JRE) 1.3.1_01 Plug-in

Platforms, Kernels and Compilers

VOCAL has been compiled using the following operating systems.

Table 2-1.

Operating System	*Kernel*	*Compiler*
Debian 2.4.18 (Itanium 64)	2.4.18	gcc 2.96
FreeBSD		
Linux (Caldera Preview)	2.4	g++ 2.95.2
Linux (Mandrake 8.1)	2.4.8-26mdk	g++ 2.96
Linux (Redhat 6.2)	2.2.14	g++ 2.91.66
Linux (Redhat 7.1)	2.4.2-2	g++ 2.96
Linux (Redhat 7.2)	2.4.7-10	g++ 2.96
Linux (Suse 7.3)	2.4.10-4GB	g++ 2.95.3
Linux (TurboLinux Beta Version)	2.4-18-0.5	g++ 3.04
Solaris 2.8	2.8	Sun workshop 6 update 1
Solaris 2.8	2.8	g++ 2.96
Solaris 2.8 (i86pc)	2.8	gcc/g++ 2.95.3 (www.sunfreeware.com)
Windows 2000	-	MSVC++ 6.0, Service Pack 4 (Windows SIP UA only)

Linux Hosts

Linux Red Hat

Linux Red Hat Versions 6.2, 7.1 or 7.2 must be installed *and* running on each host machine.

We also recommend that you select the CUSTOM->INSTALL EVERYTHING option. This will ensure that all of the VOCAL components compile properly.

Networking Requirements

Introduction

Before downloading and compiling VOCAL, it would be wise to verify that your network settings and domain name service (DNS) are set properly.

INET and multicast addresses

Purpose

This test is intended to make sure that your machine has a valid IP address, as opposed to a loop-back address, and states that the multicast is up and running.

Steps

To verify your INET and multicast addresses

1) From root, type:

`ifconfig –a eth0`

The following appears:

```
eth0      Link encap:Ethernet  HWaddr 00:03:47:9C:2E:BA
          inet addr:192.168.0.3 Bcast:192.168.0.3 Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
          RX packets:91383 errors:0 dropped:0 overruns:0 frame:0
          TX packets:53 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          Interrupt:11 Base address:0xdf00
```

Results

If your inet addr: is 127.0.0.1, which is a loop back address, you will have trouble running VOCAL and you need to change it to something that resembles the address in the above example.

If the third line does not show UP BROADCAST RUNNING MULTICAST, then you need to change this as well. Consult your Linux manual for instructions about how to fix these problems.

Verifying DNS

Purpose

This test is intended to make sure that your DNS server is setup properly.

Steps

To verify DNS:

1) From root, type:

`cat /etc/resolv.conf`

The following appears:

```
search <domain name>
nameserver 171.69.2.133
```

Results If no nameserver appears, then you need to troubleshoot your DNS server.

Testing DNS

Purpose This test is just a simple ping request to a well-known domain on the Internet.

Steps **To test DNS:**

1) Type,

   `ping www.yahoo.com`

   The following appears:

```
PING www.yahoo.akadns.net (216.115.102.77) from 128.107.140.170 : 56(84)
bytes of data.
64 bytes from w5.snv.yahoo.com (216.115.102.77): icmp_seq=0 ttl=237
time=11.6 ms
64 bytes from w5.snv.yahoo.com (216.115.102.77): icmp_seq=1 ttl=237
time=11.5 ms
64 bytes from w5.snv.yahoo.com (216.115.102.77): icmp_seq=2 ttl=237
time=14.9 ms
64 bytes from w5.snv.yahoo.com (216.115.102.77): icmp_seq=3 ttl=237
time=11.7 ms
...
--- www.yahoo.akadns.net ping statistics ---
14 packets transmitted, 14 packets received, 0% packet loss
round-trip min/avg/max = 10.9/11.6/14.9 ms
```

Results This is not a big deal as far as running VOCAL is concerned, however, if you can't connect to the Internet, you can't download VOCAL and the JRE.

Verifying your host name

Purpose You will need to know this name after installing VOCAL, when you are ready to call up Provisioning.

Steps

To verify your host name:

1) From root, type:

hostname

The following appears:

<hostname>

Verifying your hosts file

Purpose

In order for VOCAL to function properly, your hosts file must contain both a loopback address for the localhost and a mapping between your IP address and hostname.

Steps

To check your /etc/hosts file,

1) Type:

cat /etc/hosts

Result

This file should contain a mapping for the loopback address and your IP address, as shown here:

```
127.0.0.1        localhost.localdomain localhost
<ip address>     <hostname>
```

If a hostname other than localhost appears in the first line, edit your file to match the above example.

Caution

Don't remove the loopback address (127.0.0.1), otherwise the programs that require network functionality will fail.

Further Clarification

If your /etc/hosts file has your hostname listed as an alias for the loopback address (127.0.0.1), VOCAL will not work. Make sure that the address associated with your hostname is your ethernet IP address.

This is incorrect:

```
127.0.0.1   localhost.localdomain localhost hostname
```

This is correct:

```
127.0.0.1   localhost.localdomain localhost
<ip-addr>   <hostname>
```

Java Run-time Environment

Optional

This material is revelant only to users who have elected to install VOCAL with Java.

Source

The Java Run-time Environment (JRE) is available from java.sun.com/j2se/1.3/jre/download-linux.html. Version 1.4.0 of VOCAL requires version 1.3.1_03 of the JRE. Download the RedHat RPM Shell Script version and install it before compiling VOCAL.

Opening the Shell

To open the shell,

1) Type:

```
sh j2re-1.3.1_03-linux-i386-rpm.bin
```

A license appears asking you to accept its conditions.

2) Type:

```
yes
```

The rpm file appears in the directory.

3) To decompress this file, type:

```
rpm –hivv jre-1.3.1_03-linux-i386.rpm
```

Result

The JRE software is now installed on your host.

Installing and Deploying VOCAL

Introduction

This sections describes how to acquire, compile and deploy VOCAL onto a single host.

Acquiring VOCAL Software

Source

VOCAL software is available on the Vovida.org website as a Linux source file, vocal-1.4.0.tar.gz. The tar ball is over 21 MB in size. Depending on your connection speed, downloading this file could require a few minutes, a few hours or, hopefully not, a few days.

Once you have finished downloading the tar ball, you need to untar it, recompile VOCAL, and run the deploy script before you can test it. All of these steps should be done as root. We used the Bash shell to enter all of the command examples shown below.

Untarring VOCAL

Purpose

This section tells you how to extract the tarball, vocal-1.4.0.tar.gz, into a directory on your machine.

Steps

To untar the tar ball, as root,

1) Type:

```
tar -xvzf vocal-1.4.0.tar.gz
```

Results

The files are extracted from the tarball.

Compiling and Deploying VOCAL without Java

Overview

This section explains how to do the following:

- Compile VOCAL without Java
- Deploy VOCAL without Java
- Access the HTTP Provisioning GUI

Installing VOCAL without Java

Options

There are two options for downloading and installing VOCAL:

- Installing from RPM/PKG
- Installing from Source

See the sections below that pertain to your preferred installation method.

Installing from RPM/PKG (Basic All-in-one System)

Linux

Download vocalbin-1.4.0alpha2-15.i386.rpm from www.vovida.org.

To install the RPM, as root, type:

```
rpm -U vocalbin-1.4.0alpha2-15.i386.rpm.
```

Solaris

Download VOCALb-sparc.pkg.tar.Z

To install the RPM, as root, type:

```
uncompress VOCALb-sparc.pkg.tar.Z
tar -xvf VOCALb-sparc.pkg.tar
pkgadd -d VOCALb-sparc.pkg
```

Configuring VOCAL

On Linux

To configure VOCAL, as root, type:

```
usr/local/vocal/bin/allinoneconfigure/allinoneconfigure
```

On Solaris

To configure VOCAL, as root, type:

```
opt/vocal/bin/allinoneconfigure/allinoneconfigure
```

On Both Linux and Solaris

The allinoneconfigure script starts and asks a number of questions. The text for these questions is shown in the section “Deploying VOCAL with Java” on page 2-22. Answer all questions with the default answers.

If everything goes well, the following message appears:

```
Congratulations: you have successfully installed VOCAL!
```

If you see an error message instead, your VOCAL system was not installed properly. See “Troubleshooting” on page 2-28 for help.

Apache Server

The Apache web server has now been reconfigured to provide basic web-based provisioning and must be restarted for this to take effect.

To restart Apache type:

```
/etc/rc.d/init.d/httpd restart
```

To use the web-based provisioning, point a web browser to:

`http://<your server name>/vocal/`

Verifying the Installation

On Linux

To verify your installation, as root, type:

`/usr/local/vocal/bin/allinoneconfigure/verifyinstall`

On Solaris

To verify your installation, as root, type:

`opt/vocal/bin/allinoneconfigure/verifyinstall`

On Both Linux and Solaris

If your installation is OK, you should see the following text:

```
Basic call succeeded.  Installation appears to be OK.
```

If you see an error message instead, your VOCAL system is not working properly. See “Troubleshooting” on page 2-28 for help.

Making a First Call

SIP UA

To make a first call, you can run the SIP UA, which is included in the VOCAL binary tree.

To run the SIP UA:

1) From /usr/local/vocal/bin, run
`./ua -r -f /usr/local/vocal/etc/ua1000.cfg`
in one xterm and run
`./ua -r -f /usr/local/vocal/etc/ua1001.cfg`
in another xterm.

■**Note**
Both of these commands need to be run on the machine on which you have installed your VOCAL system.

2) Press ‘a’ for offhook on each UA terminal.

To call one terminal from the other, type:

`a 1 0 0 0 #`

or

`a 1 0 0 1 #`

Press ‘z’ to hangup.

Installing From Source

Overview

Follow these instructions to compile and install the VOCAL source code on your machine:

On Linux

To install VOCAL from source, as root, type:

```
./configure
make
make install
```

On Solaris

To install VOCAL from source, as root, type:

```
./configure --with-toolchain=gnu --with-ar=/usr/local/
bin/ar
make
make install
```

On Both Linux and Solaris

The allinoneconfigure script starts and asks a number of questions. The text for these questions is shown in the section “Deploying VOCAL with Java” on page 2-22. Answer all questions with the default answers.

If everything goes well, the following message appears:

```
Congratulations: you have successfully installed VOCAL!
```

If you see an error message instead, your VOCAL system was not installed properly. See “Troubleshooting” on page 2-28 for help.

After installing your VOCAL system, follow the instructions above to configure, verify the installation and make a first call.

Default Provisioning

Configuration settings

A newly configured allinone system contains the following provisioning:

- Call Blocking Feature Server on port 5080
- Call Return Feature Server on port 5095
- Call Screening Feature Server on port 5100
- Caller ID Blocking Feature Server on port 5090
- Conference Bridge Marshal Server on port 5064
- Forward All Calls Feature Server on port 5085
- Forward No Answer or Busy Feature Server on port 5105
- Gateway Marshal Server on port 5065
- Redirect Server on port 5070
- UAVM Servers on ports 5170, 5171, 5172, 5173, 5174
- User Agent Marshal Server on port 5060
- Voicemail Feature Server on port 5110
- two users for testing: 1000 and 1001
- a dial plan that sends to the PSTN gateway
  - all numbers starting with 9
  - all numbers with 6 or more digits

Using the Web-based GUI

Use the web-based provisioning GUI to add, delete, or change users as as to enter the address of the default PSTN gateway.

For all other changes you must use the Java provisioning GUI (see ”Installing the Tools RPM”)

Installing the Tools RPM

Overview

The Tools RPM adds the Java GUI to the system to allow you to provision more features of the VOCAL system. The Java GUI runs in a web browser, using the Java Plug-in version 1.3 or higher. Use Internet Explorer version 5 or higher, or use Netscape version 4.7 or higher. Other web browsers, such as KDE Konquerer and Mozilla should also work.

Acquiring the Software

Download vocalbin-tools-1.4.0alpha2-15.i386.rpm from www.vovida.org.

Installation

To install the Tools RPM, as root, type:

```
rpm -U vocalbin-tools-1.4.0alpha2-15.i386.rpm.
```

Accessing Provisioning

Web access

To use the HTTP provisioning, point a web browser to:

```
http://<your server name>/vocal/
```

User Configuration

Reference

See the System Administration Guide for more information.

System Status

SNMP

The System Status page requires SNMP, which is not included in the Tools RPM but is included in the source code. In order to run SNMP, you must do an installation from source. See the FAQ-O-MATIC entry http://www.vovida.org/fom-serve/cache/685.html.

Uninstalling VOCAL

From an RPM

If you installed from an RPM, follow these instructions to uninstall VOCAL.

To uninstall VOCAL, as root, type:

```
rpm -e vocalbin-1.4.0alpha2-15
```

This will remove some, but not all of the files under /usr/local/vocal. The provisioning_data directory is left intact so that you will not lose the provisioning from this installation.

To remove everything, including the provisioning data, as root, type:

```
rm -Rf /usr/local/vocal
```

From Source

If you installed from source, follow these instructions to uninstall VOCAL.

To uninstall VOCAL, as root, type:

```
rm -Rf /usr/local/vocal
```

Compiling and Deploying VOCAL with Java

Overview

This section covers the following:

- Compiling VOCAL with Java
- Deploying VOCAL with Java
- Accessing the Java Provisioning

Compiling VOCAL with Java

Purpose

The source code comes uncompiled because, the file size for the binaries (267.8 Mb) is too large for many in our community to download over the public Internet.

■ **Note**

If you have both the JRE rpm file and the VOCAL tarball sitting in this directory, you might want to move them to /tmp or delete them to free up some disk space.

Steps

To compile VOCAL, as root:

1) Type:

```
cd vocal1.4.0
```

2) Press **Enter**, then type:

```
make CODE_OPTIMIZE=1 all
```

CODE_OPTIMIZE=1 Option

The CODE_OPTIMIZE=1 option increases the compile time but makes the software run much more quickly than it would otherwise. As this *is* an option, you could just run **make all**.

Caution

If you run **make all** with the CODE_OPTIMIZE=1 option, you must also run **make allinone** with the same option, otherwise VOCAL will not work. Similarly, if you run **make all** without any options, you must run **make allinone** without any options.

Results

You’re compiling over 260 Mb of code, which may take as long as 2 hours to complete. There is no interaction in this process; it returns the command line prompt when it is finished.

■ **Note**

It is possible that the script will come to a grinding halt if it runs out of disk space. If that happens try to free up at least a Gigabyte and try again. You can run this script over and over again without doing any damage to the software.

Deploying VOCAL with Java

Purpose

This next set of instructions will deploy VOCAL onto a single server and create two provisioned users for testing. This routine takes about 10 minutes to complete, and the first interactive prompt appears within the first 2 minutes.

■**Note**

If you are re-deploying, make sure that you are in the vocal1.4.0 directory.

Steps

To deploy VOCAL onto one machine, as root,

1) Type:

`make CODE_OPTIMIZE=1 allinone`

The script copies a large number of files to new directories, then runs ./allinonecofigure, which brings up the following warning:

```
WARNING WARNING WARNING WARNING WARNING WARNING WARNING

The following may destroy any configuration that you
currently have on your system. If you would like to exit,
press Contol-C now.

WARNING WARNING WARNING WARNING WARNING WARNING WARNING

Welcome to the VOCAL all-in-one configuration system.
This program is intended to configure a small example
system which has all of the servers running on one box,
known as the "all-in-one" system.

This all-in-one system is NOT intended as a production
system, but as a simple example to get users started
using VOCAL.

This configuration WILL destroy any currently configured
system on this machine. If this is not acceptable,
please quit by pressing Control-C now.
```

2) So, if you are not comfortable with loading VOCAL onto your machine, press Control-C. Otherwise, press Enter to accept the defaults for all of these:

```
Host IP Address [<your IP address>]:
Remote Contact hostname or address (this should NOT be
loopback or 127.0.0.1) [<your IP address>]:
Multicast Heartbeat IP Address [224.0.0.100]:
Multicast Heartbeat Port [9000]:
Log Level [LOG_ERR]:
User to run as [nobody]:
```

```
HTML directory to install .jar and .html files into [/
usr/local/vocal/html]:
```

3) The next prompt offers two options: Read both before continuing.

```
Provisioning your VOCAL system requires the ability to
view the contents of /usr/local/vocal/html from the web.
There are two ways to do this, review both options before
answering the next prompt.

Option 1:

  Step 1:  Answer y to the next prompt. This will let
           this script attempt to add the following to
           your Apache httpd.conf file:

           Alias /vocal/ "/usr/local/vocal/html/"
           <Directory "/usr/local/vocal/html">
               AllowOverride None
               Order allow,deny
               Allow from all
           </Directory>

           Adding this script creates an alias from the
           following URL:

           http://<hostname>/vocal/

           which points to /usr/local/vocal/html .


  Step 2:  After this script has completed running,
           restart your copy of Apache (httpd) for
           the change to take effect.


Option 2:  Answer n to the next prompt. Then, manually
           copy the directory /usr/local/vocal/html to
           your web server's HTML directory.  You
           should not need to restart your copy of
          Apache after the script has completed running.


Would you like this script to attempt Option 1, Step 1
(y), or would you like to perform Option 2 manually (n)?
```

```
(If y, you must restart Apache after this script has com-
pleted running.) [y]:
```

4) Press **Enter** to accept the default [y].

The following prompt appears:

```
Directory where Apache’s httpd.conf is located [/etc/
httpd/conf]:
```

5) Press **Enter**.

The following prompt appears requiring a different answer than the default:

```
Path to Java VM (if none, automated provisioning will
not work)(please include name of interpreter, e.g. /usr/
java/bin/java)
[none]:
```

6) Type:

`/usr/java/jre1.3.1_03/bin/java`

■ **Note**

Future versions of the JRE may use a different path. For more information refer to java.sun.com.

If you are an advanced Java user who is using the Java Development Kit, rather than the JRE, you will need to type in its location rather than the location of the JRE.

7) Press **Enter** and the following confirmation of your configuration appears:

```
                     ******************************

Configuration:

Host IP Address:                    172.19.174.207
Remote Contact Address:             172.19.174.207
Multicast Heartbeat IP Address:     224.0.0.100
Multicast Heartbeat Port:           9000
Log Level:                          LOG_ERR
User to run as:                     nobody
HTML directory:                     /usr/local/vocal/html
Add alias to:                       /etc/httpd/conf/
httpd.conf
Java Runtime:                     /usr/java/jre1.3.1/bin/
java

                     ******************************

Continue [n]:
```

8) Type:

y

9) Press **Enter** and the following appears:

```
********************************************************

Beginning the VOCAL configuration process.  This may
take a few seconds.

setting umask to 0022 -- users other than root must be
able to run VOCAL in its default configuration.

fixing permissions...
creating uavm config files...
creating UA config files for 1000 and 1001...
Stopping VOCAL...
Creating and filling provisioning directory...
Creating and filling HTML directory...
  Adding alias to httpd.conf directory...
Creating users...
Starting VOCAL...
Creating configuration files...
Restarting VOCAL...

********************************************************
Configuration complete!

  To configure your VOCAL system:

  * Go to

     http://<hostname>/vocal/

    and select Provisioning.

[root@<hostname> vocaln.n.n]#
The deployment is complete.
```

Apache File

If you are re-deploying, the alias may have already been added to your Apache configuration file, and the lines,

```
It appears that the alias has already been added to
httpd.conf. Skipping...
```

will appear in the script. This should not create any problems for you.

Accessing the Java Provisioning

Positioning the plug-ins

First, you must ensure that the Java plug-ins are in the correct location for use by Provisioning.

Steps

To export the plug-ins, as root:

1) Type:

```
export NPX_PLUGIN_PATH=/usr/java/jre1.3.1_03/plugin/
i386/ns4
```

Restarting Apache

If you chose the default option from the make allinone script for adding the alias script to httpd.conf, you need to restart the Apache server.

Steps

To restart the Apache server:

1) Type:

```
/etc/rc.d/init.d/httpd restart
```

2) Press **Enter** and the following appears:

```
Stopping httpd:                              [  OK  ]
Starting httpd:                              [  OK  ]
```

■ **Note**

If Stopping httpd returns [FAILURE], no problem, that means that it was not running. Starting httpd will start it, which is the result you want.

Launching the browser

Finally, you can launch Netscape (or your favorite browser, providing it supports the Java JRE).

Steps

To launch your browser and check for plug-ins:

1) Type (for example, if you are using Netscape):

```
netscape &
```

Your browser appears.

■ **Note**

The path for the plug-ins, specified by the EXPORT command above, is good only for this session of your browser.

2) (If you are using Netscape) type the following into the location field:

```
about:plugings
```

Results

You should see an obvious reference to jre-1.3.1_03-linux-i386 at the top of the listing.

Configuration menu

Purpose

You will need your browser to provision more users and servers to your system.

Steps

To test your access to provisioning

1) Type the following URL into the browser's location field:

`http://<hostname>/vocal/`

Results

A simple, text web page appears with the following title and menu:

```
VOCAL Configuration for <hostname>
Choose one:
• Provisioning
• System Status
• User Configuration
```

If this appears, then you can reduce or close your browser for now.

This completes the installation process.

Troubleshooting

Steps

If you are having trouble with a VOCAL system, here are a few things to try.

1) Make sure that you have read the latest VOCAL Errata in the VOCAL Errata section of Faq-o-matic, which can be found in README in the ERRATA section.

2) If you are still having trouble, please consult the VOCAL mailing list archives, at www.vovida.org/pipermail/vocal/.

3) Post a question to the VOCAL mailing list. Subscribe to the list first, by going to www.vovida.org/mailman/listinfo/vocal/ and following the directions there. Then, post your question to vocal@vovida.org.

Starting, Restarting, Stopping VOCAL

Introduction

This section describes how to start, re-start or stop VOCAL on a deployed system. These instructions apply regardless if you have chosen to compile and deploy with or without Java.

vocald and vocalctl

Vocald (VOCAL Daemon) and vocalctl (VOCAL Control) are replacements for vocalstart (used in versions 1.2.0 and 1.3.0 of VOCAL). Vocald allows automatic starting and stopping of the vocal servers based on provisioning. Vocalctl allows users to determine which servers are running.

Both scripts are written in perl and use named pipes to communicate, which, by default, are stored in /usr/local/vocal/var .

Files

The configuration file for vocald is:

```
/usr/local/vocal/etc/vocald.conf
```

Using the VOCAL Daemon

When to stop VOCAL

Stopping VOCAL when you need to stop all active calls and processes for:

- all VOCAL functions on the local server
- a specified VOCAL function on the local server
- a specified port number on the local server

When to restart or start VOCAL

After provisioning VOCAL, using the provisioning GUI or modifying the vocal.conf file, you may need to restart or start VOCAL to implement the changes made during provisioning. You can restart or start VOCAL:

- for all VOCAL functions on the local server
- for a specified VOCAL function on the local server
- for a specified port number on the local server

Caution

Restarting and starting VOCAL will terminate any active calls that utilize the function provided by the server. For example, if you have re-provisioned the Marshal Server and restarted the Marshal Server, any active calls using the Marshal Server will be terminated. However, if you restart a Feature Server that is not handling the active calls, the call will proceed normally.

Syntax for restarting, starting, and stopping

The syntax for using vocalctl is:

/usr/local/vocal/bin/vocalctl <command> <process>

where:

- <command> can be one of:
  - shutdown — stop vocald (no error if not running)
  - status — show status of the processes
  - disable <process> — disable process (or all if no argument)
  - restart — restart vocald (or start if not running)
  - config — print vocal.conf configuration values
  - start — start vocald
  - stop — stop vocald
  - reload — reread vocal.conf configuration file
  - enable <process> — enable process (or all if no argument)
- <process> can be rs, ms, fs, js, cdr, net, vm, pd, or hs

3

Provisioning

Chapter Content

This chapter provides information about using the GUI Technician's screens to edit provisioning data.

Topic *(continued)*	**See Page**

Working With The GUI Environment

Introduction

This is a general overview of the GUI environment. This section describes:

- Logging into the GUI
- The screen layouts
- The icons, buttons and fields in the GUI

Before you begin

The machine that was used to install the VOCAL system is known as the provisioning host. You can access the Provisioning Server from the provisioning host, or from any other PC that is connected to the network where the VOCAL system resides.

Before you can work with the Provisioning GUI, you must have the following:

- A web browser loaded on your machine. The browser can be any type that takes a Java Run-time Environment plug-in version 1.3.1_01.
- Access to the Internet to download the Java plug-in. All networking requirements are covered under "Java Run-time Environment" on page 2-9.

You must also know the name of your provisioning host and the system name that was entered during the software installation. This information is used to access the web page that contains the links to the GUI system utilities. This web address can be expressed as:

http://<provisioning host name>/vocal/index.html

■ **Note**

An example of this web address could be:
http://local_host/local_system/index.html

Accessing the GUI

To access the Provisioning GUI, go to:

http://<provisioning host name>/vocal/index.html

and select **Provision System**.

The first Provisioning screen calls a Java plug-in. The plug-in is not shipped with the software, it must be downloaded from the Internet. Normally, this plug-in would have been downloaded to the provisioning host during the software installation process.

However, if you are accessing the system from another machine besides the provisioning host, a download prompt appears when the screen is first loaded. Accept the download file and load the plug-in onto your machine. Once the plug-in has been loaded the Provisioning Log In screen appears as shown in Figure 3-1 on page 3-4.

Logging In

Introduction

The Provisioning Login screen provides access for Administrators to work with user data, and for Technicians to work with server data.

Definition

- The login screen is a java-enabled graphical user interface (GUI) that runs in a web browser. The browser can be any type that takes a Java Run-time Environment plug-in version 1.3.1_01.

Procedure

To Log in, follow these steps:

Table 3-1. Procedure for Logging In to the Configure Servers Screen

Step	*Description*
1	Select **Technician**. As shown in the figure below.
2	Type your user ID and password. The default user ID is vovida, password: vovida.
3	Click **Login**, or press **Enter**.

Screen Capture

Figure 3-1 shows the LogIn screen.

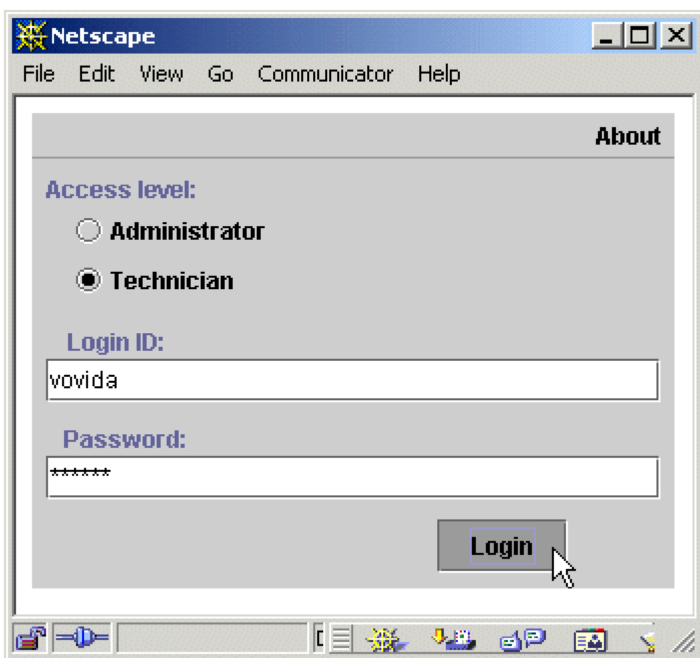


Figure 3-1. LogIn Screen

Login Screen: Items

These are the buttons and fields that appear on the Login screen.

Table 3-2. Login Screen: Items

Item	*Description*
Access Level	**Administrator** This brings up the User Configuration screen where you can add, modify or delete user entries. Refer to the System Administration Guide for more information. **Technician** This brings up the Configure Servers screen where you can provision the VOCAL system.
Login ID	For Administrators and Technicians, the default Login ID is "vovida."
Password	For Administrators and Technicians, the default password is "vovida."
Login	Submits your Login ID and Password for access to the system. The keyboard shortcut is **Enter**.

Password Administration

There is a separate user interface for changing passwords and adding or removing accounts for administrators and technicians.

It is not setup as an applet, but you can run it stand alone. It is included in the psClient.jar. The main class is vocal.pw.AdminAcctManager.

To maintain passwords, follow these instructions:

Table 3-3. Procedure for Maintaining Passwords

Step	*Description*

1	To run this user interface, type the following:
	`java -classpath /path/to/psClient.jar:path/to/ xerces.jar`
	`vocal.pw.AdminAcctManager pServer_host pServer_port`
	A screen appears with a list of all the administrative accounts.

2	Use the popup menu to maintain the accounts. ■**Note** It is possible to have an account that only accesses the administrator GUI, or only the technician GUI or both together.
3	To change the default password for a user, select the account in the GUI and click the **Change Password** button.

Configuring Servers

Introduction

This section describes the Configure Servers screen and its features.

Overview

This screen is divided into two frames, a directory tree frame and a data entry frame. The middle frame border is adjustable. Click the frame border and drag left or right to expand the view of either frame.

Screen Capture

Figure 3-2 shows the Configure Servers screen highlighting its frames.

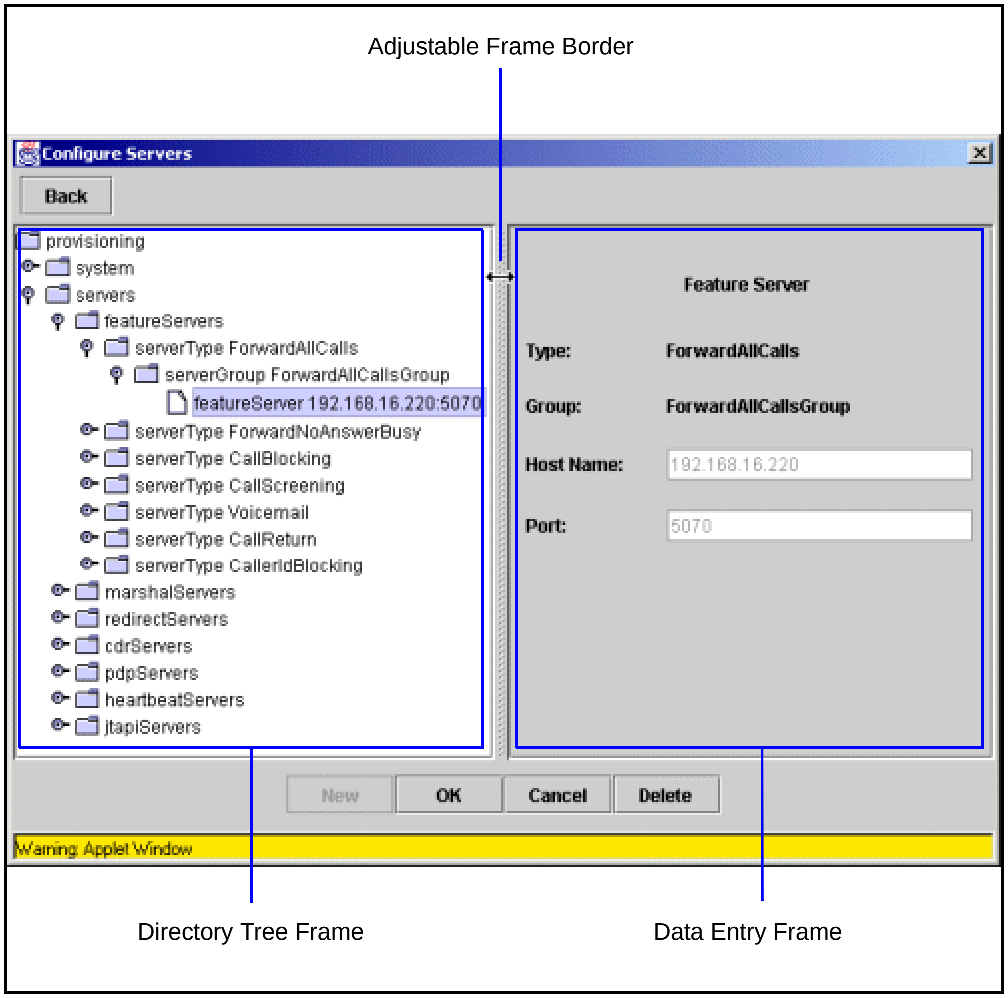


Figure 3-2. Configure Servers Screen

Configure Servers Screen Elements

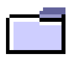
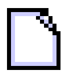
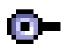
Overview

There are several types of elements on this screen including icons, buttons, and data entry fields.

Icons

Table 3-4 describes the icons:

Table 3-4. Configure Servers Screen: Icons

Icon	*Description*
	*Folder:* Appears in the directory tree when the item can be expanded into lower levels.
	*Document:* Appears in the directory tree when the item cannot be expanded.
	*Expand=Off:* Appears beside contracted folders. Click this icon to expand the folder.
	*Expand=On:* Appears beside expanded folders. Click this icon to contract the folder.

Buttons

Table 3-5 describes the buttons:

Table 3-5. Configure Servers Screen: Buttons

Button	*Description*
Back	Returns to the Log In Screen.
New	When activated, brings up a data entry screen for a new entity.
OK	Submits data entry.
Cancel	Exits data entry screen without submitting data.
Delete	When activated, deletes the displayed entity from the system.

Provisioning System Parameters

Introduction

This section provides information about editing the following:

- System Configuration Data
- OSP Server
- IP Plan
- Digital Plan

Overview

The deployment script, described in Chapter 2, Software Installation, provisions the system and servers for operation when the system is installed/ deployed. This section provides information about using the GUI to edit the system parameters.

Navigation

In the left frame of the Configuring Servers screen, there are two sub-folders under **provisioning**:

- The **system** folder provides access to the System Configuration data, the Open Settlement Protocol (OSP) Server, the Internet Protocol (IP) dial plan and the Digital Dial plan.
- The **servers** folder provides access to the servers.

This section provides information about the contents of the **system** folder.

System Configuration Data

Introduction

The System Configuration Data parameters control how the VOCAL system works with registration messages and heartbeat signals.

What is Registration?

Registration is the method used by SIP-based systems to keep the Redirect Server informed about the location of on-network user agents. Registrations are temporary: their duration is set by an expiry timer. User agents must re-register after each expiry interval to keep their registration up-to-date.

The System Configuration Data screen includes a field for setting the registration expiry timer. Refer to Figure 3-3 and Table 3-7 for more information.

What Are Heartbeats?

Heartbeats are a series of signals emitted at regular intervals, over the multicast channel, by every server on the network. Heartbeats are used exclusively by the Network Manager to provide SNMP reporting on the state of each server on the network.

The System Configuration Data screen includes fields for setting up the heartbeat broadcast, heartbeat intervals and the maximum number of missed heartbeats allowed by the system before a server is considered as being out of service. Refer to Figure 3-3 and Table 3-7 for more information.

Procedure

To edit the System Configuration Data parameters, follow these steps.

Table 3-6. Procedure for Editing System Configuration Data Parameters

Step	***Description***
1	From **system**, select **globalConfiguration.** The data entry fields appear in the right frame.
2	Edit the fields. ■**Note** These fields are discussed on page 3-10.
3	Click **OK**. Your changes are submitted to the system.

Applying the Procedure

Figure 3-3 shows the procedure for editing the parameters applied to a screen capture.

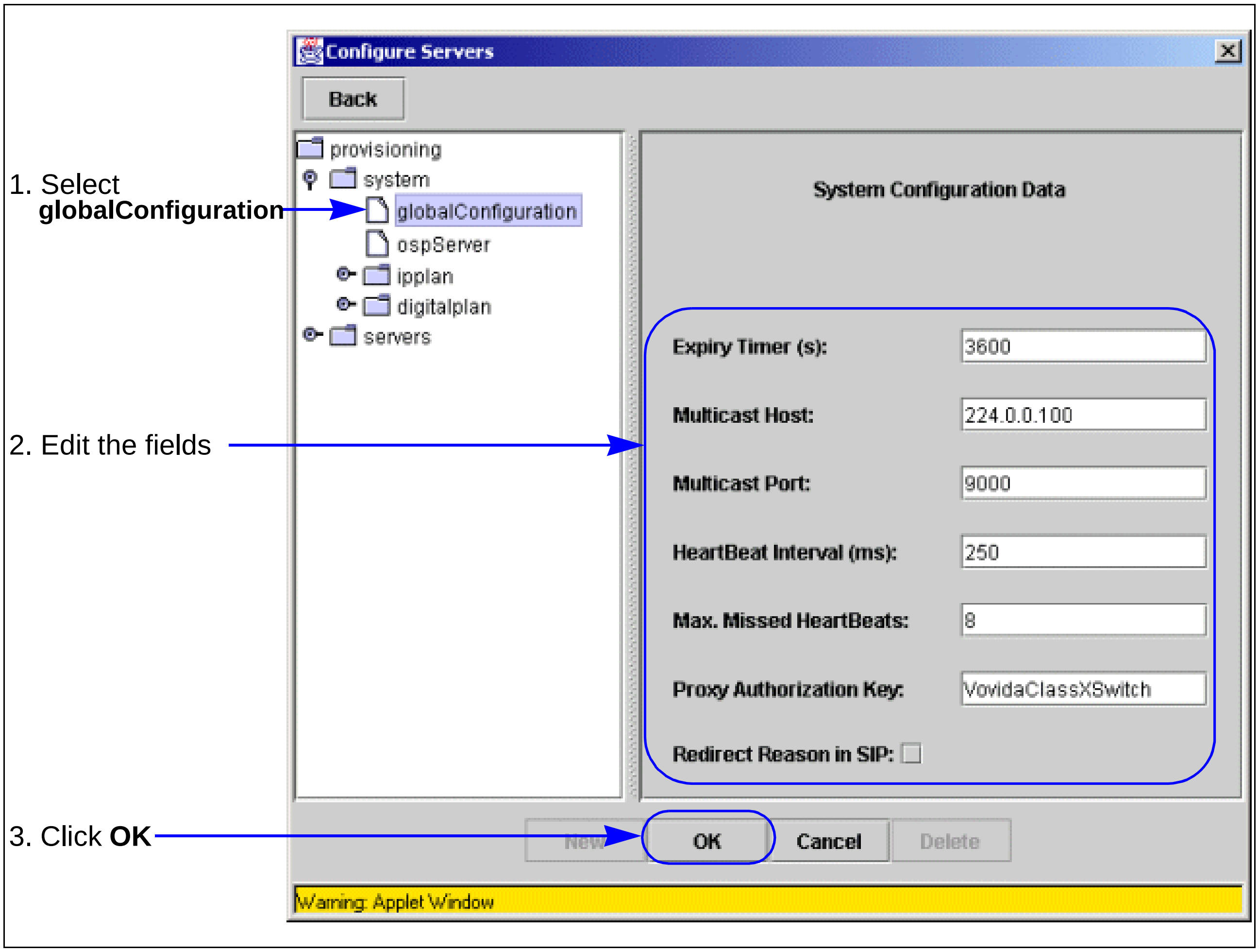


Figure 3-3. System Configuration Data: Data Entry Fields

Describing the Fields

Table 3-7 describes the data entry fields.

Table 3-7. System Configuration Data: Data Entry Fields

Field	*Description*
Expiry Timer	The time, measured in seconds, that user agents remain registered with the Redirect Server (RS) before they must send another SIP Register message. The RS compares this value against a requested expiry time sent within the SIP REGISTER message. The shorter expiry time is awarded to the User Agent. *Default value*: 3600

Table 3-7. System Configuration Data: Data Entry Fields (Continued)

Field	***Description***
Mulitcast Host	The Muliticast IP address used with the Multicast Port to send heartbeat broadcasts. *Default value:* 224.0.0.100
Multicast Port	The UDP Port used by applications to send heartbeat broadcasts. The Mulitcast Host & Port are concatenated to form a complete Multicast Address. *Default Value:* 9000 *Default Multicast Address:* 224.0.0.100:9000
Heartbeat Interval	The transmission rate for heartbeats on all applications. *Default Value:* 250 ms
Max. Missed Heartbeats	The maximum number of heartbeat an application can miss before its status becomes Inactive. *Default Value:* 8
Proxy Authorization Key	Any word or phrase used to identify the system. This phrase is added by the Marshal to all SIP messages as they enter the system, and removed by the Marshal as they exit the system. Spaces are not permitted in this field. *Example:* VOCALSystem.
Redirect Reason in SIP	If enabled, a cc-redirect header is included in SIP messages sent through the system. This header tells the callee where the call has been redirected to and why. It may also include a limit on the number of redirections permitted through the network. This field should be disabled if the host network contains devices that cannot process the cc-redirect header. *Default:* disabled

OSP Server

Introduction

These parameters enable the Open Settlement Protocol Server to communicate with a third-party settlement provider. Before you can set up this server, you will need to identify your settlement provider and the data required to connect to its server. Check the World Wide Web for lists of settlement providers. One source is:

http://dir.yahoo.com/Business_and_Economy/Business_to_Business/Communications_and_Networking/Telecommunications/Billing_and_Customer_Service/

What is OSP?

Open Settlement Protocol (OSP) is one of the protocols used to enable Quality of Service for internetwork calls. For more information, see “Quality of Service” on page 1-25.

What is a Third-Party Settlement Provider?

A third-party settlement provider is an organization that enables ISP’s to receive compensation for off-network VoIP calls routed to their network. For more information about settlement providers, see “Quality of Service” on page 1-25.

Procedure

To edit the OSP Server parameters, follow these steps:

Table 3-8. Procedure for Editing OSP Server Parameters

Step	*Action*
1	From **system**, select **ospServer**. The data entry fields appear in the right frame.
2	Edit the fields.
3	Click **OK**. Your changes are submitted to the system.

Screen Capture

Figure 3-4 shows the OSP Server data entry fields.

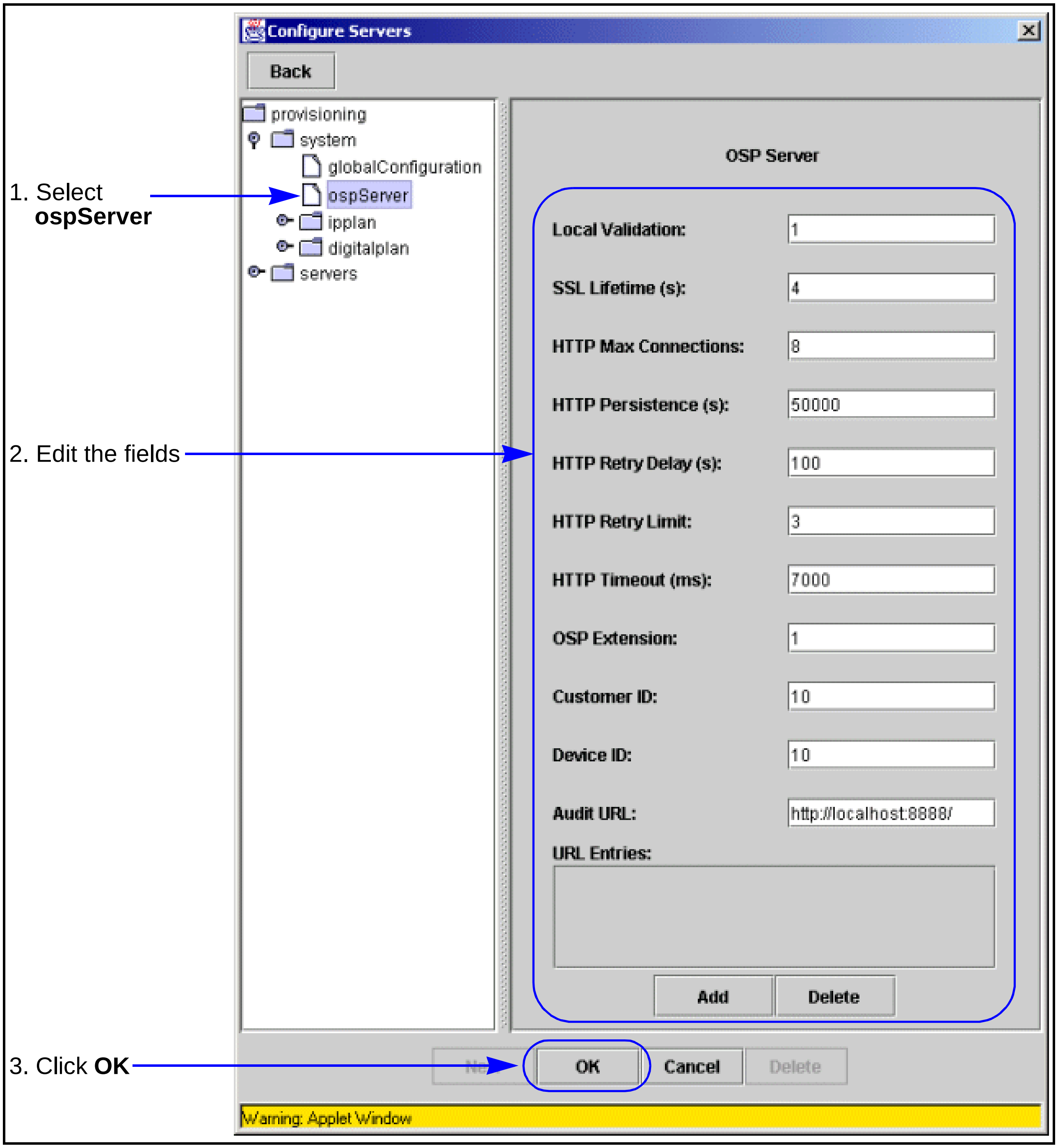


Figure 3-4. OSP Server: Data Entry Fields

Field Descriptions Table 3-9 describes these fields.

Table 3-9. OSP Server: Data Entry Fields

Field	***Description***
Local Validation	Determines how the OSP client validates tokens. *Range:* 0, 1 0 = the OSP Client authorizes token validations through a protocol exchange, where verification is done by the OSP Server. 1= the OSP Client authorizes token validations locally, by verifying digital signatures. *Default:* 0 ■ **Note** In Version 1.4.0 of VOCAL, the software ignores this field and the OSP Client validates tokens locally.
SSL Lifetime	The lifetime, measured in seconds, of a single Secured Socket Layer (SSL) session key. When this time limit expires, the OSP Client negotiates a new session key, without interrupting any communication exchanges in progress. *Recommended:* 40
HTTP Max Connections	The maximum number of simultaneous connections to be used for communication to the OSP Server. *Recommended connections:* 5 to 8
HTTP Persistence	The time, measured in seconds, that an HTTP connection is maintained after the completion of a communication exchange. Entering a longer duration will help avoid constant tear down and establishment of connections. *Recommended:* 50000
HTTP Retry Delay	The time, measured in seconds, between connection retry attempts to the OSP Server. After exhausting all service points for the OSP service provider, the OSP Client will apply the retry delay before resuming connection attempts. *Recommended:* 2
HTTP Retry Limit	The maximum number of retry attempts for connecting to the OSP Server. If no connection can be made, the OSP Client will cease connection attempts and return appropriate error conditions. This number excludes the initial connection attempt. *Recommended:* 2

Table 3-9. OSP Server: Data Entry Fields (Continued)

Field	***Description***
HTTP Timeout	The maximum time, measured in milliseconds, to wait for a response from the OSP Server. If no response is received before this time expires, the current connection is released and the OSP Client attempts to contact the next configured service point. *Recommended:* 3000
OSP Extension	Indicates whether the Customer ID and Device ID of the OSP service provider is known. *Range:* 0,1 0 = You do not know the Customer ID and Device ID of the OSP service provider. If you type 0, these fields are not required. 1 = You know the Customer ID and Device ID of the OSP service provider. If you type 1, the Customer ID and Device ID fields are required. *Default:* 0
Customer ID	A character string assigned by the OSP settlement provider as a unique customer identification code. Some providers may or may not require this field.
Device ID	A character string assigned by the OSP settlement provider as a unique device identification code. Some providers may or may not require this field.
Audit URL	The URL used for OSP audits. In Version 1.4.0 of VOCAL, the audit function is not implemented in the OSP Client. However, this field is requires an address such as, “http://localhost:8888/”. ■**Note** This field cannot remain “null”.

Table 3-9. OSP Server: Data Entry Fields (Continued)

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td>URL Entries</td><td>A list of character strings for the OSP Client to use for sending requests. Each service point takes the form of a URL. The service points can be one of the following formats:
• The domain name expressed as an octet, i.e., “http://255.255.255.255:443/osp-server”
• The domain name expressed as an alias, i.e., “httpd://www.hostname.com/service/osp”
• The domain name expressed as a local host, i.e., “httpd://local_host/osp-server/iis.dll”.
To Add a New URL Entry:
<table>
<tr><th>Step</th><th>Description</th></tr>
<tr><td>1</td><td>Click Add.
A blank space appears in the URL Entries table.</td></tr>
<tr><td>2</td><td>Type in a URL.</td></tr>
<tr><td>3</td><td>Click OK
The URL is submitted to the system.</td></tr>
</table>
To Delete a URL Entry:
<table>
<tr><th>Step</th><th>Description</th></tr>
<tr><td>1</td><td>Select an entry from the URL Entries table.
The entry’s background color changes to purple.</td></tr>
<tr><td>2</td><td>Click Delete.
The entry disappears.</td></tr>
<tr><td>3</td><td>Click OK
The altered table is submitted to the system.</td></tr>
</table>
</td></tr>
</table>

Dial Plans

Introduction

The dial plans are used by the Redirect Server (RS). These plans provide routing information for all types of calls. The VOCAL system uses two types of dialing plans, the IP and Digital dialing plans. The organization, syntax, maintenance procedures and GUI screens are identical for both plan types. The plan types are different in the types of calls that they anticipate.

What do Dial Plans Do?

Dial plans are used by the RS when the call destination does not match a provisioned user. Plans are made up of general keys and contacts associated with each key. The REQ URI field of the INVITE message, which contains the dialed number, is compared against the keys until a match is found. The Redirect Server then uses the associated contacts to determine the call routing.

Digital Plans vs. IP Plans

The Digital dialing plans are set up to handle phone numbers. These can either be Dual Tone Multi-Frequency (DTMF) tones originating from an analog phone set and translated into a SIP message by a residential gateway, or numbers entered from a SIP-based device, for example, 1-408-555-1212.

The IP dialing plans are set up to handle user addresses formatted as aliases or e-mail addresses, for example user@yourcompany.com.

Organization

Plans are organized into a table of indexes, keys and contacts.The index is used to set priorities within the table, the keys indicate anticipated dialing patterns, the contacts provide routing information. While the index determines the key's priority within the table, the contacts are also arranged by priority within each key. For more information about these fields, see Table 3-15 on page 3-24.

Syntax

The keys use Regular Expressions, also known as regex's, which are made up of ordinary and special characters. The special characters include `$', `^', `.', `*', `+', `?', `[', `]' and `\'. Any other character used in a Regular Expression is an ordinary character. Special characters become ordinary when they are preceded by a "\".

The syntax of Regular Expressions is explained more thoroughly in the following on-line reference:

http://www.math.utah.edu/docs/info/regex_1.html.

Symbols

Table 3-10 describes the most commonly used symbols in dialing plan keys. These symbols are used in the examples shown in Table 3-11 on page 3-19.

Table 3-10. Most Commonly Used Symbols in Diallng Plans

Symbol	*Description*
[ ]	Indicates a valid range. For example: [3-5]11 means 311, 411 and 511.
.	Matches any character except a new line.
{ }	Indicates a multiplier. For example: .{10} means 10 characters.
*	Indicates that the preceding regular expression can be repeated as many times as possible. For example: 011.* means 011 followed by any number of any characters.
+	Indicates that at least one match from the preceding regular expression is required. For example: 1[01]+2 does not match 12, but matches 102, 112, or any other expression that matches for 1[10]*2.
?	Indicates that zero or one match from the preceding regular expression is required. For example: 1[01]?2 matches 12, or 102, or 112 and nothing else.
\	Indicates a literal expression. For example: *69 means dialing "* 6 9".
@	This is not a special character, it is a device used in the VOCAL system to fully specify phone numbers. The @ character appears at the end of the user portion of the SIP URI. For example: 0@ means 0 is dialed by itself. The expression, 0@ does not refer to longer phone numbers that start with 0 such as, collect calls and international calls.
^	Beginning of a line.
$	End of a line.

Some Examples

Table 3-11 shows some dial plan examples.

■ **Note**

If $USER is used in the Contact field, it will be replaced with the user field in the Request URI.

Table 3-11. Dial Plan Examples

Key	*Contact*	*Description*
^sip:[3-8]11@	sip:$USER@92.168.116.110:5060;user=phone	When a user dials 311, 411, 511, 611, 711 or 811, their call is forwarded to the gateway.
^sip:.{10}@	sip:1$USER@192.168.116.110:5060;user=phone	When a user dials a 10 character phone number, for example area code + local number, the system adds a 1 to the beginning of the string, and then forwards the call to the gateway. This is used for call return.
^sip:[0-9]{10}@	sip:1$USER@192.168.116.110:5060;user=phone	This is the same as the previous example, except that this key requires all dialed characters to be numbers.
^sip:011.*	sip:$USER@192.168.116.110:5060;user=phone	When a user dials 011 followed by any number of any digits, the system forwards the call to the gateway. This is used for international calling.
^sip:1.{10}@	sip:$USER@192.168.116.110:5060;user=phone	When a user dials 1 plus a 10 digit phone number, the system forwards the call to the gateway.
^sip:.{7}@	sip:$USER@192.168.116.110:5060;user=phone	When a user dials a 7 digit phone number, the system forwards the call to the gateway.

Table 3-11. Dial Plan Examples (Continued)

Key	***Contact***	***Description***
^sip:7000@	sip:9999@192.168.116.220:5078;user=phone	When a user dials 7000, the system forwards the call to the Voice Mail User Agents. This is used by the call forwarding feature servers. ■ **Note** The actual phone number for the Voice Mail User Agents does not have to be “7000”. In this example, that number is “9999”.
^sip:*69	sip:$USER@192.168.116.220:5074;user=phone	When a user dials *69, the system forwards the call to the Call Return Feature Server.
^sip:0@	sip:$USER@192.168.116.110:5060;user=phone	When a user dials 0, the system forwards the call to the gateway. This is used to call an operator at an incumbent local exchange carrier.
^sip:00@	sip:$USER@192.168.116.110:5060;user=phone	When a user dials 00, the system forwards the call to the gateway. This is used to call an operator at a competitive local exchange carrier.
^sip:[0-9]{4}@	sip:$USER@192.168.10.10:5060;user=phone	Sends any internal numbers that cannot be directed to a specified location. For example, these calls could be sent to the receptionist.

Procedures

Introduction

The following procedures described in this section are identical for both the IP and Digital dialing plans.

Adding a New Key

To add a new key to the IP Plan table, follow these steps:

Table 3-12. Procedure for Adding a New IP Dial Plan Key

Step	***Action***
1	From **system**, select either **ipplan** or **digitalplan**. The data entry fields appear in the right frame.
2	From the Dial Plan Entries group, click **Add**. A dialog box appears.
3	Enter a key and a contact. • To add additional contacts, click **Add Contact**. • To remove a contact, click **Delete Contact**.
4	Select an Index position. The default is 0. If this is unchanged, your new key is given top priority. If you choose a position that is already being used, your new key will assume this position and push all other keys, with the same position or higher, down one position.
5	Click **OK**. The dialog box disappears.
6	Click **OK** on the Configure Servers screen to submit your new entry.

Editing Keys

To editing existing keys, follow these steps:

■ **Note**

This procedure is illustrated in Figure 3-6.

Table 3-13. Procedure for Editing an IP Dial Plan Key

Step	***Action***
1	From **system**, select either **ipplan** or **digitalplan**. The data entry fields appear in the right frame.
2	Select a table entry. You can select the Index number, the key or any of the contacts.
3	From the Dial Plan Entries group, click **Edit**. A dialog box appears.

Table 3-13. Procedure for Editing an IP Dial Plan Key (Continued)

Step	*Action*
4	Make your changes. • Edit the Key and Contacts by double clicking on their fields. • To add additional contacts, click **Add Contact**. • To remove a contact, click to select it then click **Delete Contact**. • To move a contact's position, cut the contact information and paste it into a new line. • To change the key's priority, select a different **Index** value.
5	Click **OK**. The dialog box disappears.
6	Click **OK** on the Configure Servers screen to submit your new entry.

Deleting Keys

To delete keys, follow these steps:

Table 3-14. Procedure for Deleting an IP Dial Plan Key

Step	*Action*
1	From **system**, select either **ipplan** or **digitalplan**. The data entry fields appear in the right frame.
2	Select a table entry. You can select the Index number, the key or any of the contacts.
3	From the Dial Plan Entries group, click **Delete**. A dialog box appears with the following message: Do you want to delete <key> at index <index_value>.
4	Click **Yes**. The key is removed from the table.
5	Click **OK** on the Configure Servers screen to submit your new entry.

Screen Capture

Figure 3-5 shows the procedure for editing a digital Dial Plan table entry.

■ **Note**

The procedure and the dialog box are the same for IP Dial Plans.

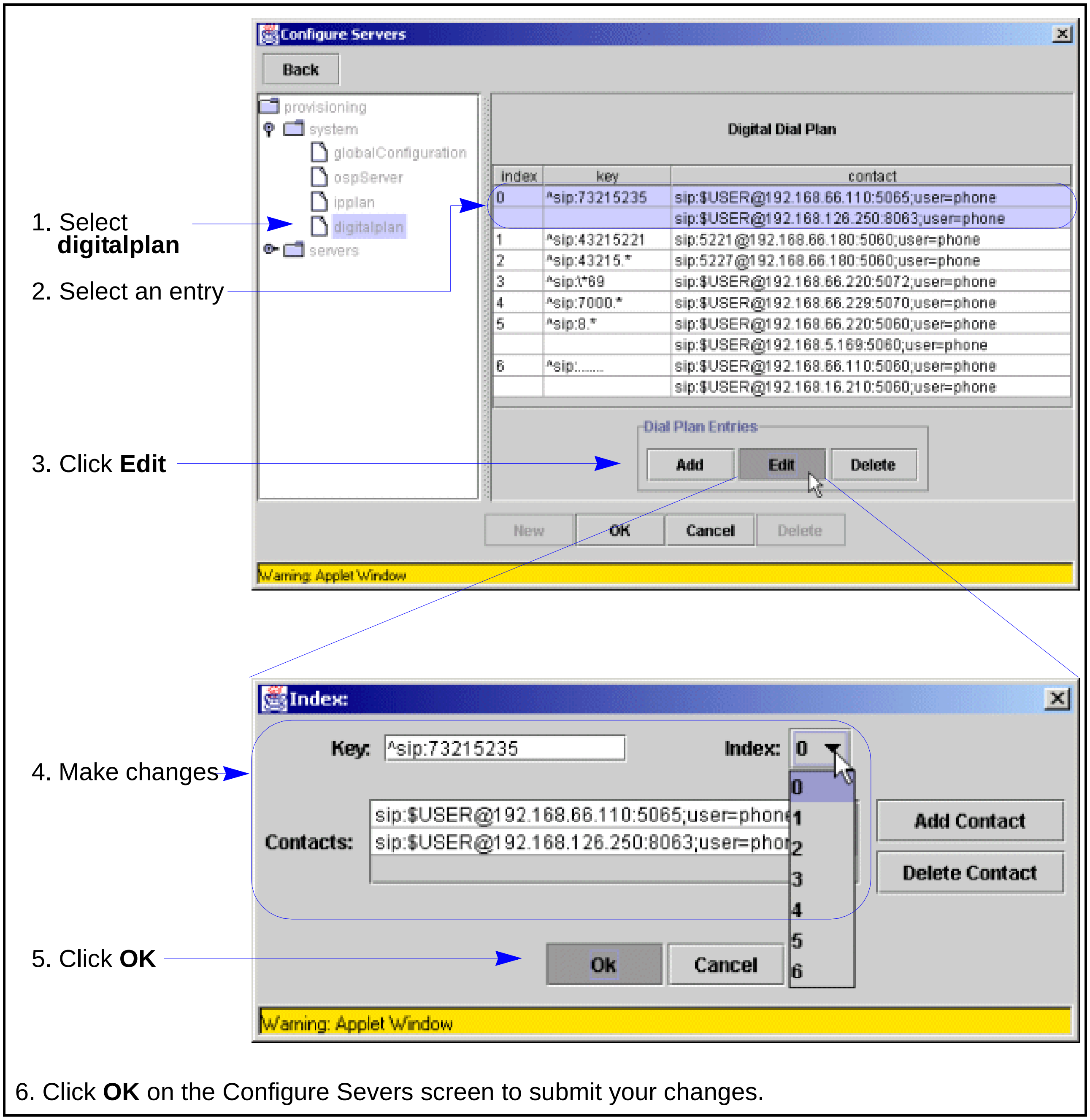


Figure 3-5. Editing Digital Dial Plans

Fields and Buttons

Introduction

This section describes the fields and buttons found on the dialing plan screens.

Table Fields

Table 3-15 describes the table fields.

Table 3-15. Dialing Plans: Table Fields

Field	*Description*
Index	The search order for the dial plan records. Given a request URI, the Redirect Server (RS) searches the dial plan for the first key that matches it. The search order determines priority within the plan.
Key	A series of characters representing a regular expression. When the user field of the request URI does not match a user configured in your system, this string is compared against the Request URI field of the SIP INVITE message.
Contact	The Contact List is the list of contacts in SIP format that will be traversed when an invite message comes in containing a Request URI field that matches the key. The format should look something like this: sip:$USER@198.176.54.32:5060;user=phone If $USER is used it will be replaced with the user field in the Request URI.

Buttons: Dial Plan Entries

Table 3-16 describes the Dial Plan Entries group buttons.

Table 3-16. Dial Plan Entries Group: Buttons

Button	*Description*
Add	Brings up a dialog box for adding new entries.
Edit	Brings up a dialog box for editing existing entries. ■**Note** Select a table entry before clicking **Edit**.
Delete	Brings up a confirm prompt with options to delete (**Yes**) or cancel (**No**) the request. ■**Note** Select a table entry before clicking **Delete**.

Dialog Box: Buttons

Table 3-17 describes the buttons found in the dialog box. The Index, Key and Contact are described above in Table 3-16 on page 3-24.

Table 3-17. Dial Plan Entries Dialog Box: Buttons

Button	*Description*
Add Contact	Adds a blank field to the contacts.
Delete Contact	Removes a contact field from the table entry. ■**Note** Select a contact before clicking **Delete**.
OK	Submits the entry to the table and closes the dialog box. ■**Note** You must click **OK** from the Configure Servers screen to submit the table to the system.
CANCEL	Closes the dialog box.

Digital Dial Plan

Introduction

The Digital plan contains prefixes and phone numbers for any special handling for phone numbers not related to a specific user. The Redirect Server (RS) checks the Digital plan to provide routing information to the other servers in the system.

Screen Capture

Figure 3-6 shows the Digital Dial Plan table.

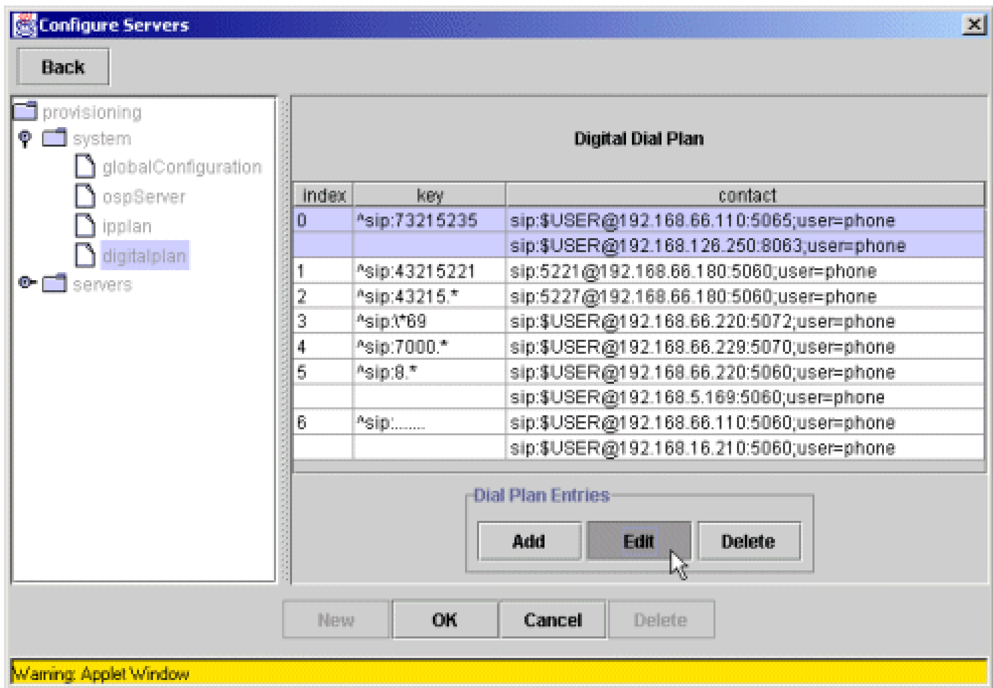


Figure 3-6. Digital Plan: Data Entry Screen

IP Dial Plan

Introduction

The IP plan contains the Universal Resource Indicators (URI's) of on-network subscribers. The Redirect Server (RS) checks the IP plan to provide routing information to the other servers in the system.

■ **Note**

The IP Plan is under development and is not in use with version 1.4.0 of the VOCAL system.

Screen Capture

Figure 3-7 shows the IP Dial Plan screen.

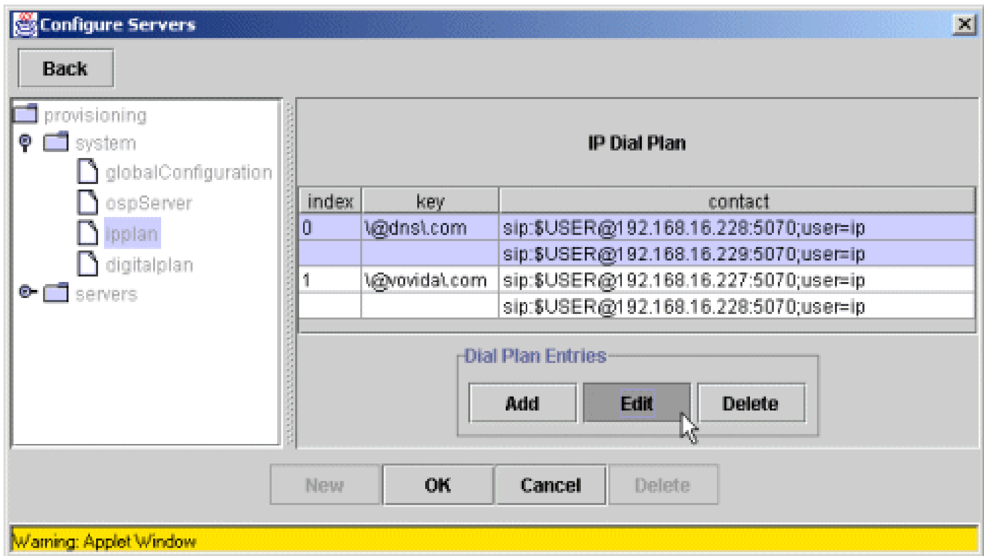


Figure 3-7. IP Dial Plan: Data Entry Screen

Provisioning Servers

Introduction

This section provides information about:

- Adding server groups
- Adding servers
- Editing servers
- Deleting servers

Overview

Provisioning is the series of tasks required for setting up the servers to communicate with each other, and with off-network entities. These tasks are performed by the deployment script during the system installation.

New entities can be added, edited and deleted through the GUI. The procedure is the same for each server type.

Navigation

In the left frame of the Configure Servers screen, there are two sub-folders under **provisioning**:

- The **system** folder provides access to the System Configuration data, the OSP server, the IP dial plan and the Digital Dial plan.
  For more information, see "Provisioning System Parameters" on page 3-8.
- The **servers** folder provides access to the servers.

This section provides information about the contents of the **servers** folder.

Server Organization

The servers are organized into the following levels of hierarchy:

- *Process:* a general description of the server's function such as, the feature servers or the CDR server.
- *Type:* a specific description of the server's function such as, Forward All Calls Feature Server or User Agent Marshal Server (UAMS). Only the feature and marshal servers are divided by type.
- *Group:* a method for assigning users to servers. Each group can contain multiple servers. Providing users with primary and backup servers within assigned groups, enhances system redundancy and reliability.

Diagram

Figure 3-8 is a color enhanced view of the directory tree. These colors do not appear on the GUI, they are used in Figure 3-8 to emphasize the servers and their layers of hierarchy.

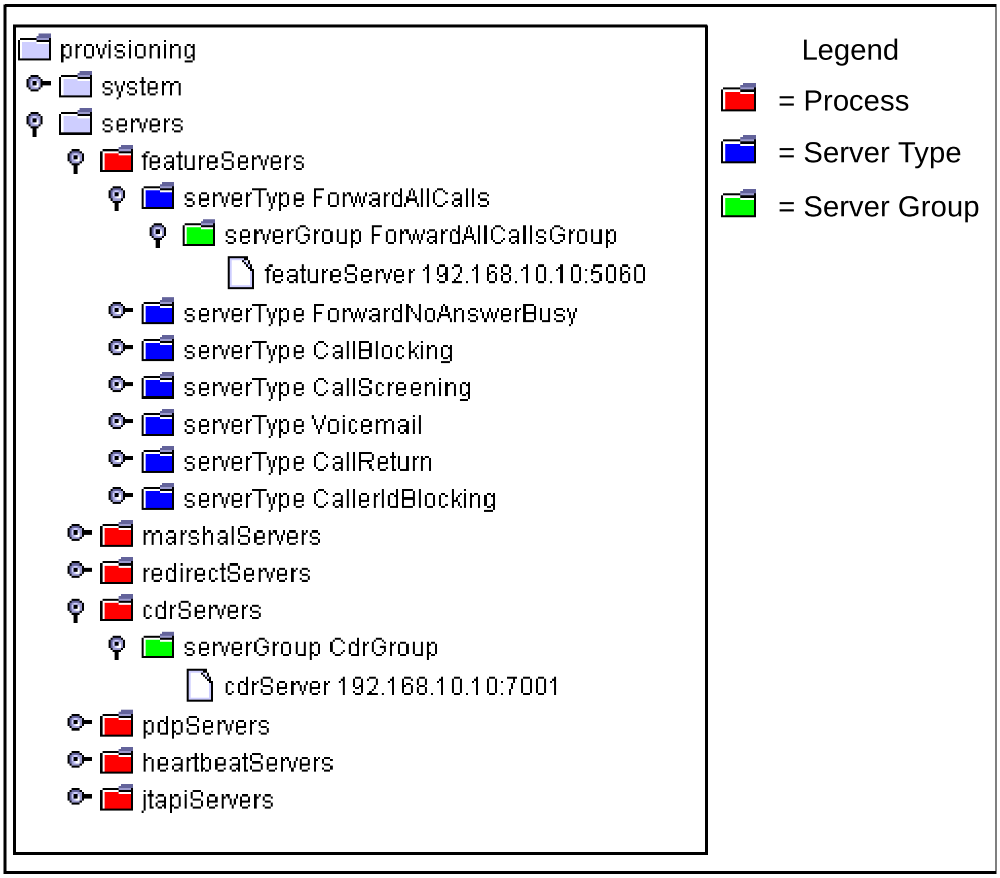


Figure 3-8. Colorized Directory Tree Showing the Server Hierarchy

Adding New Server Groups

Introduction

The VOCAL system provides one default server group per server type. New server groups can be added to the system.

■ **Note**

It is recommended that the administrator use these default groups without adding any more to the system.

Overview

In all cases, the servers are assigned to groups, and these groups are used to balance user traffic. The groups combine logical and physical servers together in redundancy schemes. The IP addresses of each server indicates its host. On a system loaded onto a single host, all logical servers have unique port numbers, concatenated to the same IP address.

For more information about balancing user traffic and redundancy schemes, see Appendix A, Engineering Guidelines.

■ **Note**

Group names cannot be changed after they have been defined. If you require different group names than those provided by the deployment script, you must delete the group, and add a new group with the desired name. It is better to delete groups before any users have been added to the system.

Procedure

To add a server group, follow these steps:

Table 3-18. Procedure for Adding a Server Group

Step	*Description*
1	From **servers**, select one of the following folders: • **redirectServers** • **cdrServers** • **pdpServers** • **heartbeatServers** • **jtapiServers** ■ **Note** To add a feature or marshal server group, see Table 3-19, “Procedure for Adding a Marshal or Feature Server Group,” on page 3-32
2	Click **New**. The Group Name data entry field appears in the right frame.
3	Type a group name.
4	Click **OK**. The group is added to the system.

Screen Capture Figure 3-9 shows the procedure for adding new groups.

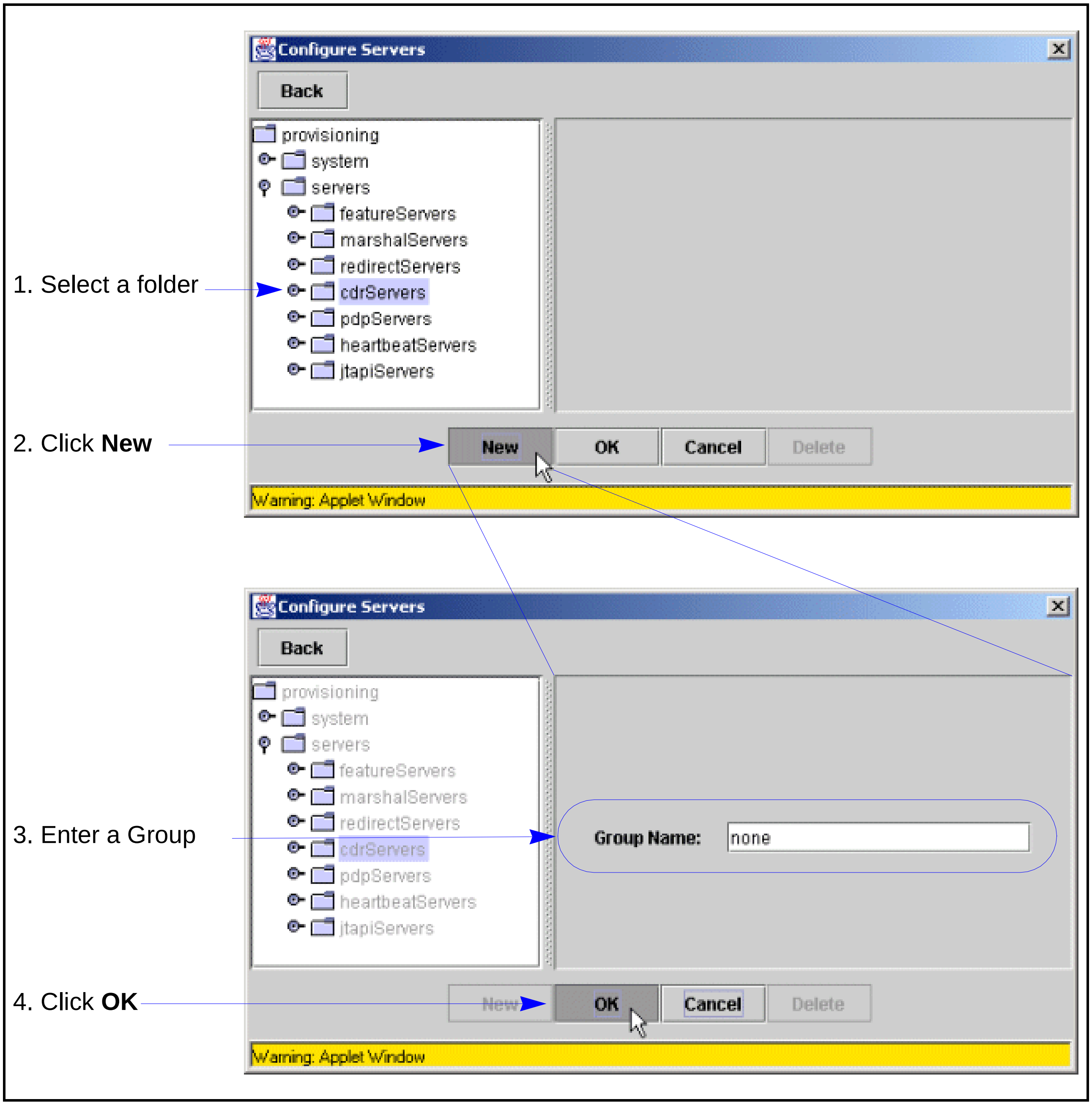


Figure 3-9. Adding a New CDR Server Group

Adding a Marshal or Feature Server

To add a Marshal or Feature Server group, follow these steps:

Table 3-19. Procedure for Adding a Marshal or Feature Server Group

Step	***Description***
1	From **servers**, select and expand one of the following folders: • **featureServers** • **marshalServers**
2	Select a server type, for example, **serverType ForwardAllCalls**.
3	Click **New**. The Group Name data entry field appears in the right frame.
4	Type a group name.
5	Click **OK**. The group is added to the system.

Screen Capture Figure 3-9 shows the procedure for adding new feature server groups.

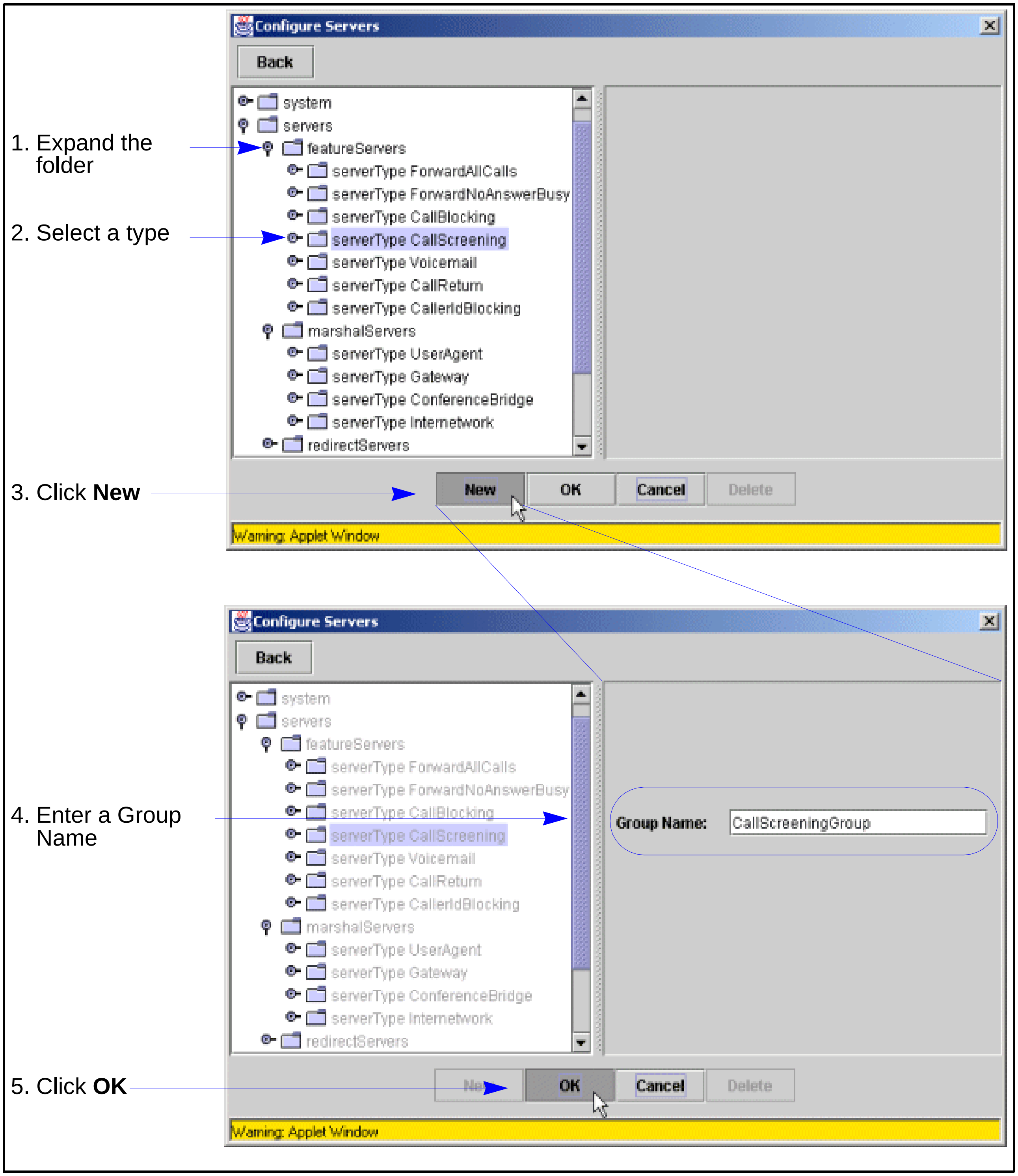


Figure 3-10. Feature Server Group: Data Entry Screen

Adding New Servers

Introduction

This section describes how to add new servers.

Definition

New servers can be added to server groups as the system scales up to process more users. Servers within the same groups are distinguished from each other by their port numbers.

Procedure

To add a new server, follow these steps:

■ **Note**
All new servers must be manually added to the vocal.conf file on the host. Otherwise the server will not come up when the system is rebooted.

Table 3-20. Procedure for Adding a Server

Step	*Description*
1	From any of the server folders, select a server group.
2	Click **New**. The data entry fields appear in the right frame. Figure 3-11 shows the CDR server's data entry fields as an example.
3	Fill in the fields. ■ **Note** For field descriptions, select one of the following: • Table 3-24. CDR Server: Data Entry Fields: CDR • Table 3-28. Redirect Server: Data Entry Fields • Table 3-30. User Agent Marshal Server: Data Entry Fields • Table 3-32. Gateway Marshal Server: Data Entry Fields • Table 3-34. Conference Bridge Marshal Server: Data Entry Fields • Table 3-37. Internetwork Marshal Server: Data Entry Fields • Table 3-39. Feature Server: Data Entry Fields • Table 3-41. Voice Mail Feature Server: Data Entry Fields • Table 3-43. JTAPI Server: Data Entry Fields • Table 3-45. Heartbeat Server: Data Entry Fields • Table 3-47. Policy Server: Data Entry Fields
4	Click **OK**. The server is added to the group.

Screen Capture Figure 3-11 shows the data entry procedure for adding a CDR Server.

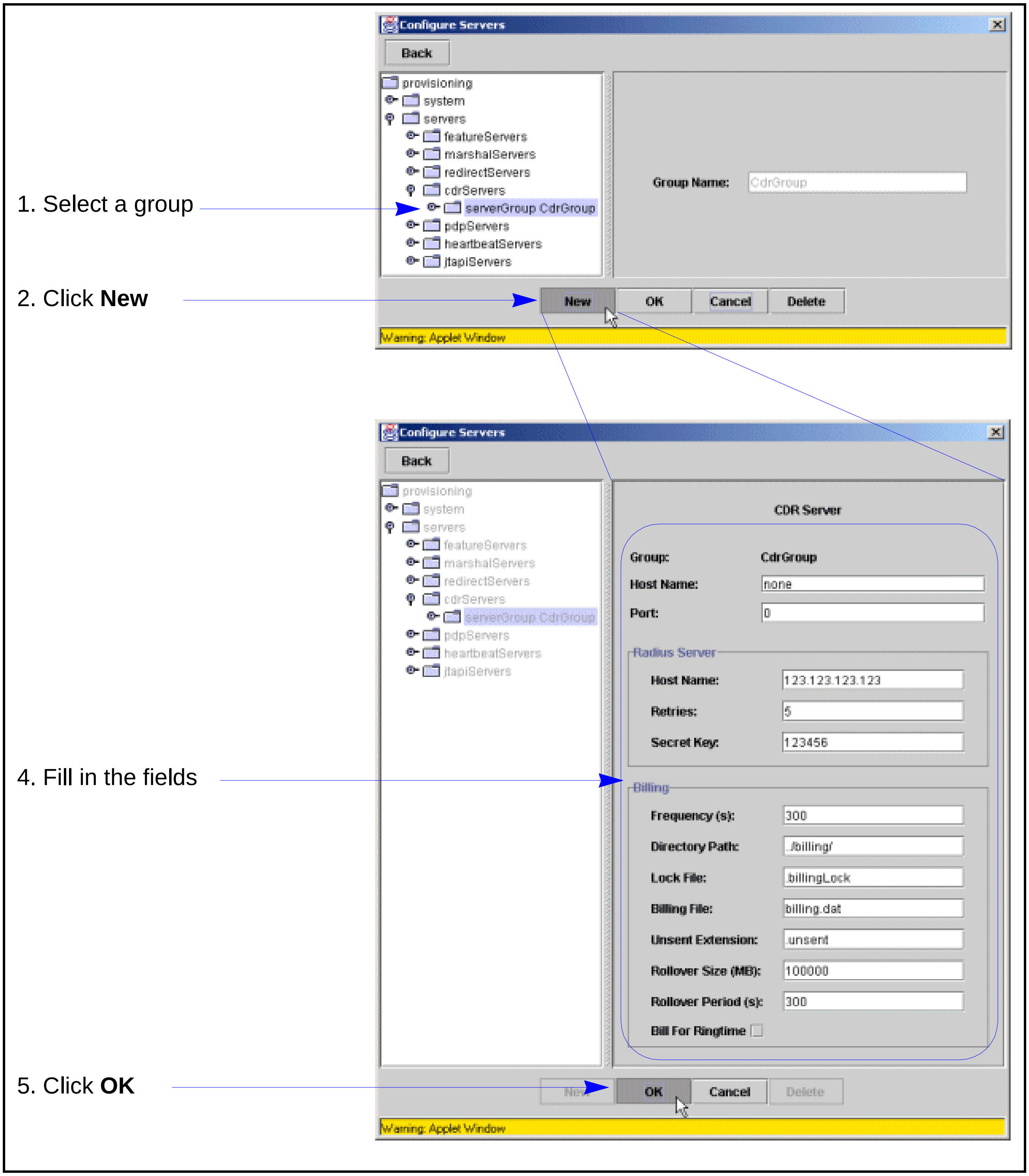


Figure 3-11. Adding a CDR Server

Editing Servers

Introduction

This section describes how to edit existing servers.

Definition

After a new server has been added, its provisioning data can be edited by selecting from the directory tree and changing its fields.

After changing the fields, from /usr/local/vocal/bin/vocalstart, run ./vocalstart restart [server type]. See Chapter 2 for more information about restarting servers.

Caution

Feature Servers should not be edited after users have been added to the system. Otherwise, all user agents that are assigned this feature must be regenerated to accept the changes.

Procedure

To edit a server, follow these steps:

Table 3-21. Procedure for Editing a Server

Step	***Action***
1	From a server group, select a server. The data entry fields appear in the right frame. Figure 3-12 shows the CDR Server's data entry fields as an example.
2	Make your changes. ■**Note** For field descriptions, select one of the following: • Table 3-24. CDR Server: Data Entry Fields: CDR • Table 3-28. Redirect Server: Data Entry Fields • Table 3-30. User Agent Marshal Server: Data Entry Fields • Table 3-32. Gateway Marshal Server: Data Entry Fields • Table 3-34. Conference Bridge Marshal Server: Data Entry Fields • Table 3-37. Internetwork Marshal Server: Data Entry Fields • Table 3-39. Feature Server: Data Entry Fields • Table 3-41. Voice Mail Feature Server: Data Entry Fields • Table 3-43. JTAPI Server: Data Entry Fields • Table 3-45. Heartbeat Server: Data Entry Fields • Table 3-47. Policy Server: Data Entry Fields
3	Click **OK**. The changes are submitted to the system.

Screen Capture Figure 3-11 shows the procedure for editing a CDR Server.

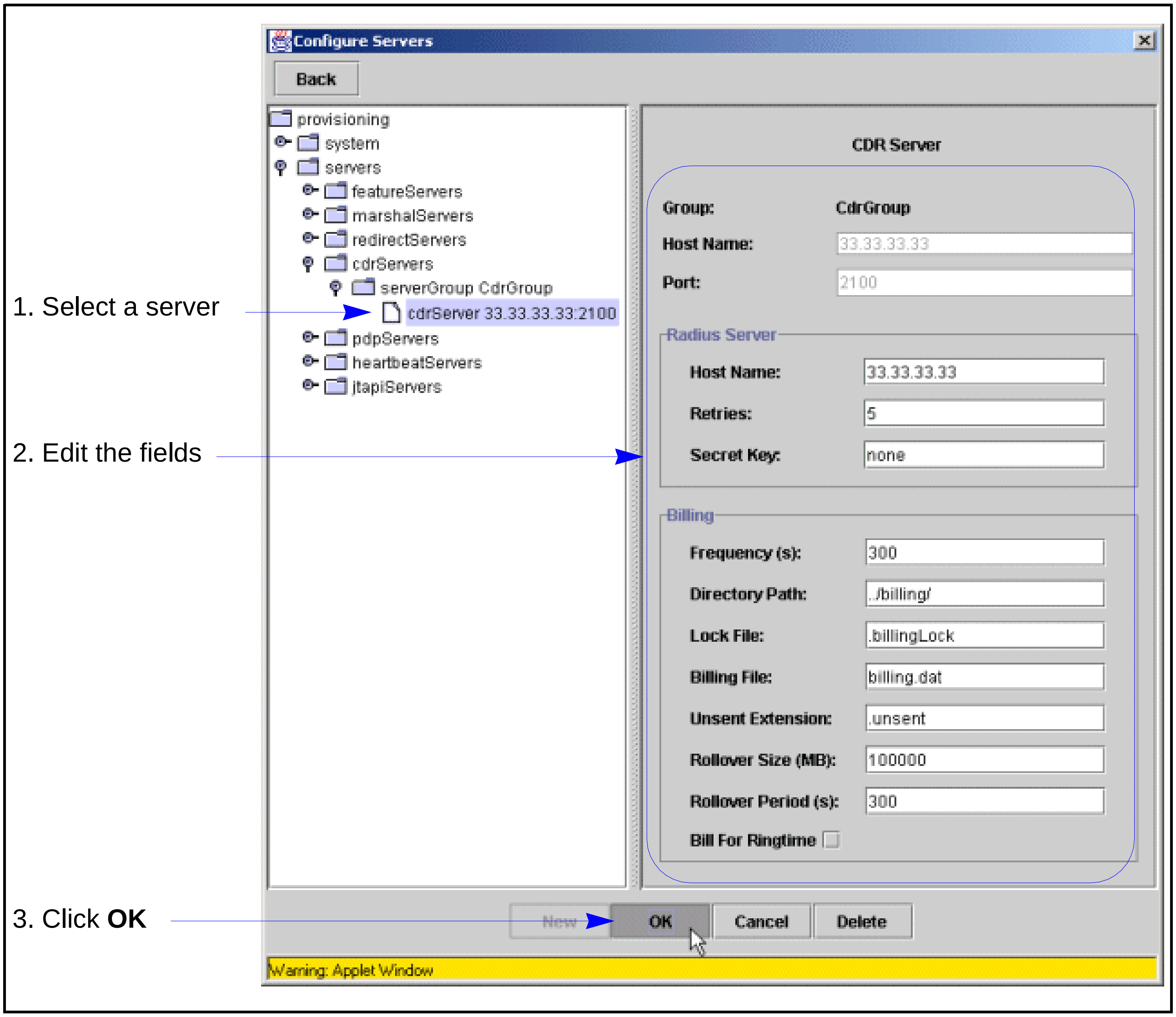


Figure 3-12. Editing a CDR Server

Deleting Servers

Introduction

This section describes how to delete servers.

Definition

Servers can be deleted from the system at any time, but once they are in service with user agents, deleting servers is not recommended.

Caution

Feature Servers should not be deleted after users have been added to the system. Otherwise, all user agents that are assigned this feature must be regenerated, by logging into Provisioning as an Administrator and manually changing all affected users, to accept the changes.

Steps

To delete a server, follow these steps:

Table 3-22. Procedure for Deleting A CDR Server

Step	*Action*
1	Select a server. The data entry fields appear in the right frame.
2	Click **Delete**. The following prompt appears: `Are you sure you want to delete this CDR Server?`
3	Click **Yes**. The server is deleted. ■**Note** If you click **No**, the dialog box disappears with no affect to the servers or any of the data.

Call Detail Record Servers

Introduction

This section describes how to provision the CDR Server.

Overview

The Call Detail Record (CDR) Server collects information from the Marshal Servers indicating when calls start and their duration. If a third-party billing system has been installed into the system, the CDR Server can send billing records to it in the Remote Authentication Dial-In User Service (RADIUS) format.

Provisioning Tasks

Table 3-23 shows the provisioning tasks that can be performed with the CDR Server.

Table 3-23. CDR Server, Tasks

Task	***Comments***
Adding a CDR Server Group	CDR Server groups are the first sub-level below **cdrServers** on the directory tree. In most systems, there is one CDR Server group containing 1 or 2 servers.
Adding a CDR Server	If a second CDR Server is required, add it to the existing group. Most systems have two CDR Servers mirrored for reliability.
Editing a CDR Server	Select a server, edit the fields and then click **OK**. Tables 3-24, 3-25 and 3-26 describe the fields.
Adding Additional CDR Servers	Systems that can process up to 50 calls per second, do not require more than 2 CDR Servers. Additional servers can be added as the system grows in size, or if greater reliability is desired. Follow the instructions in “Adding New Servers” on page 3-34. For more information about scaling the system, see Appendix A, Engineering Guidelines.
Deleting CDR Servers	Select a server and click **Delete**.

CDR Servers, Data Entry Fields

Introduction

This section shows and describes the data entry fields.

Screen Capture

Figure 3-13 shows the CDR Server’s data entry fields.

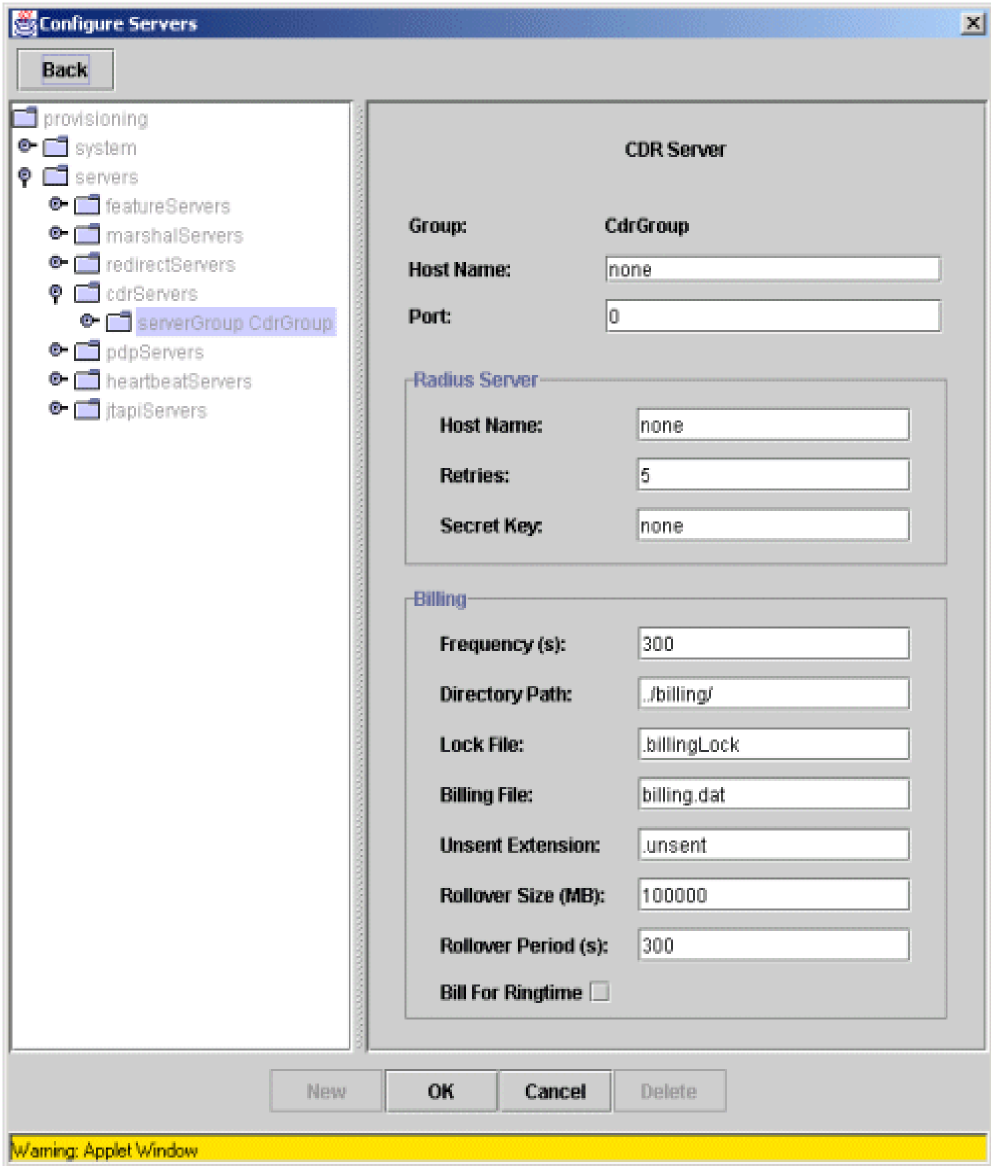


Figure 3-13. CDR Server: Data Entry Screen

Data Entry Fields Tables 3-24 through 3-26 describe the fields.

Table 3-24. CDR Server: Data Entry Fields: CDR

Field	***Description***
Host Name	The IP address of the CDR Server.
Port	The port number of the CDR Server.

Table 3-25. CDR Server: Data Entry Fields: Radius Server

Field	***Description***
Host Name	The IP address of Radius Server.
Retries	The number of times that the Radius Server will attempt to connect to the CDR Server before ceasing attempts and returning error messages. *Default:* 5
Secret Key	A text string used for MD5 Digest security.

Table 3-26. CDR Server: Data Entry Fields: Billing

Field	***Description***
Frequency	The frequency, measured in seconds, that the CDR Server sends records to the billing system. *Default*: 86400 seconds (1 day)
Directory Path	Location of stored billing files on CDR Server.
Lock File	If there are two CDR Servers, this file informs each server if the other is writing to the Directory Path. This prevents file allocation errors.
Billing File	Call detail records are written to this file.
Unsent Extension	A file extension appended to billing files that are to be sent to the billing system. *Default:* unsent
Rollover Size	Maximum permitted size, measured in megabytes, of a billing file before it is automatically rolled over. Default: 5
Rollover Period	Maximum permitted age, measured in seconds, of a billing file before it is automatically rolled over. *Default:* 86400 seconds (1 day)
Bill for Ringtime	If selected, the billing starts when the phone starts ringing.

Redirect Server

Introduction

This section describes how to provision the Redirect Server (RS).

Overview

The RS stores contact and feature data for all registered subscribers, and a dialing plan to enable routing for off-network calls.

Provisioning Tasks

Table 3-27 shows the provisioning tasks that can be performed with the RS.

Table 3-27. Redirect Server: Provisioning Tasks

Task	*Comments*
Adding a RS Group	RS groups are the first sub-level below **redirectServers** on the directory tree.
Editing a RS	Select a server, edit the fields and then click **OK**. Table 3-28 describes the fields.
Adding Additional RS's	Additional servers can be added as the system grows in size, or if greater reliability is desired. Follow the instructions in "Adding New Servers" on page 3-34. For more information about scaling the system, see Appendix A, Engineering Guidelines.
Deleting RS's	Select a server and click **Delete**.

Redirect Servers, Data Entry Fields

Introduction

This section shows and describes the data entry fields.

Screen Capture

Figure 3-14 shows the Redirect Server's (RS's) data entry fields.

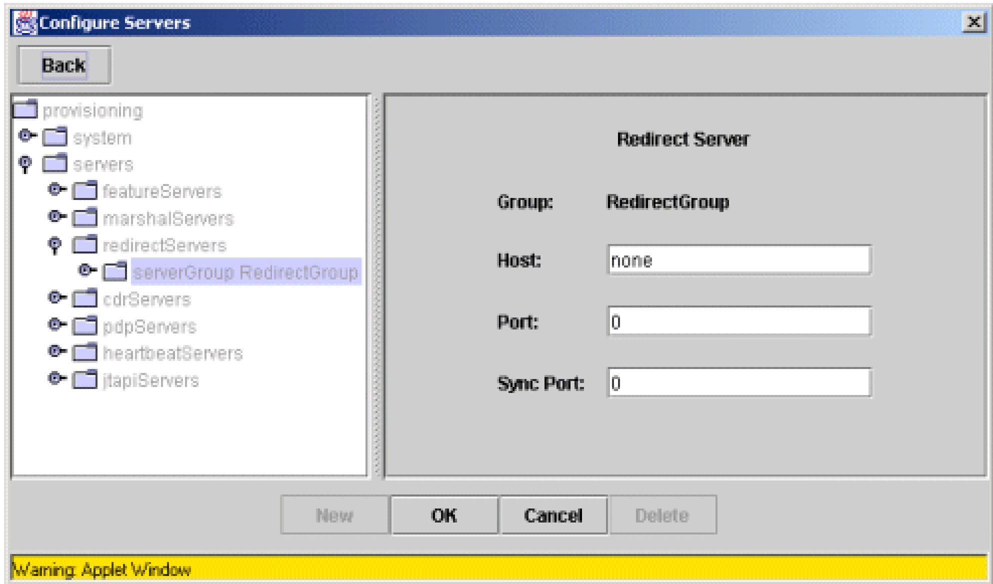


Figure 3-14. Redirect Server: Data Entry Screen

Data Entry Fields

Table 3-28 describes the fields.

Table 3-28. Redirect Server: Data Entry Fields

Field	*Description*
Host	The IP address of the RS.
Port	The port number used by the RS.
Sync Port	The Sync Port is a UDP port used by the RS to synchronize its data with the other RS on the system.

User Agent Marshal Servers

Introduction

This section describes how to provision the User Agent Marshal Server (UAMS).

Overview

The UAMS is the primary point of contact for all user agents (UA's), such as SIP phones. The UAMS authenticates UA's and forwards messages on their behalf.

Provisioning Tasks

Table 3-29 shows the provisioning tasks that can be performed with the UAMS.

Table 3-29. User Agent Marshal Server: Provisioning Tasks

Task	*Comments*
Adding a UAMS Group	UAMS groups are the second sub-level below **marshalServers**, and **serverType User Agent** on the directory tree.
Adding a UAMS	Depending on the subscriber base size, a system may include several MS's in the same server group or dispersed over several groups.
Editing a UAMS	Select a server, edit the fields and then click **OK**. Table 3-30 describes the fields.
Adding Additional UAMS's	Systems that can process up to 50 calls per second, do not require more than 2 UAMS's. Additional servers can be added as the system grows in size, or if greater reliability is desired. Follow the instructions in "Adding New Servers" on page 3-34. For more information about scaling the system, see Appendix A, Engineering Guidelines.
Deleting UAMS's	Select a server and click **Delete**.

User Agent Marshal Servers: Data Entry Fields

Introduction

This section shows and describes the data entry fields.

Screen Capture

Figure 3-15 shows the UAMS's data entry fields.

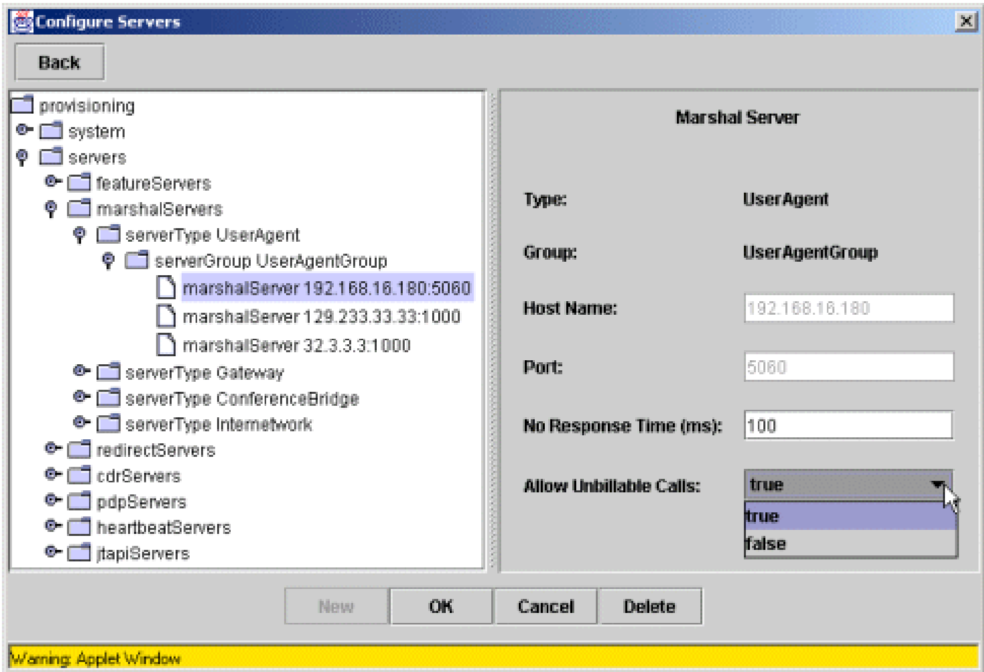


Figure 3-15. User Agent Marshal Server: Data Entry Screen

Data Entry Fields

Table 3-30 describes the fields.

Table 3-30. User Agent Marshal Server: Data Entry Fields

Field	Description
Type	A notice describing this server's MS type.
Group	A notice describing this server's MS group.
Host Name	The host name or IP address of this server.
Port	The SIP port number of this server.
No Response Time (ms)	This timer keeps track of the connection to the CDR Server. If the connection cannot be established before the time value, measured in milliseconds, for this cell expires, then the call is considered unbillable.
Allow Unbillable Calls	If the call is considered unbillable, and this field is set to TRUE, then the call proceeds. If the call is considered unbillable, and this field is set to FALSE, the MS rejects the call with a 402, Payment Required, message.

Gateway Marshal Servers

Introduction

This section describes how to provision the Gateway Marshal Servers (GWMS's).

Overview

The GWMS's connect the VOCAL system to the PSTN gateways.

Provisioning Tasks

Table 3-31 shows the provisioning tasks that can be performed with the CDR Server.

Table 3-31. Gateway Marshal Server: Provisioning Tasks

Task	Comments
Adding a GWMS Group	GWMS groups are the second sub-level below **marshalServers**, and **serverType Gateway** on the directory tree. GWMS's that are dedicated to specific PSTN area codes must be separated by area code into different groups.
Adding a GWMS	The number of GWMS's may depend on the number of dedicated PSTN area codes your system servers.
Editing a GWMS	Select a server, edit the fields and then click **OK**. Table 3-32 describes the fields.
Adding Additional GWMS's	Additional servers can be added as the system grows in size, or if greater reliability is desired. For more information about scaling the system, see Appendix A, Engineering Guidelines.
Deleting GWMS's	Select a server and click **Delete**.

Gateway Marshal Servers: Data Entry Fields

Introduction

This section shows and describes the data entry fields.

Screen Capture

Figure 3-16 shows the GWMS's data entry fields.

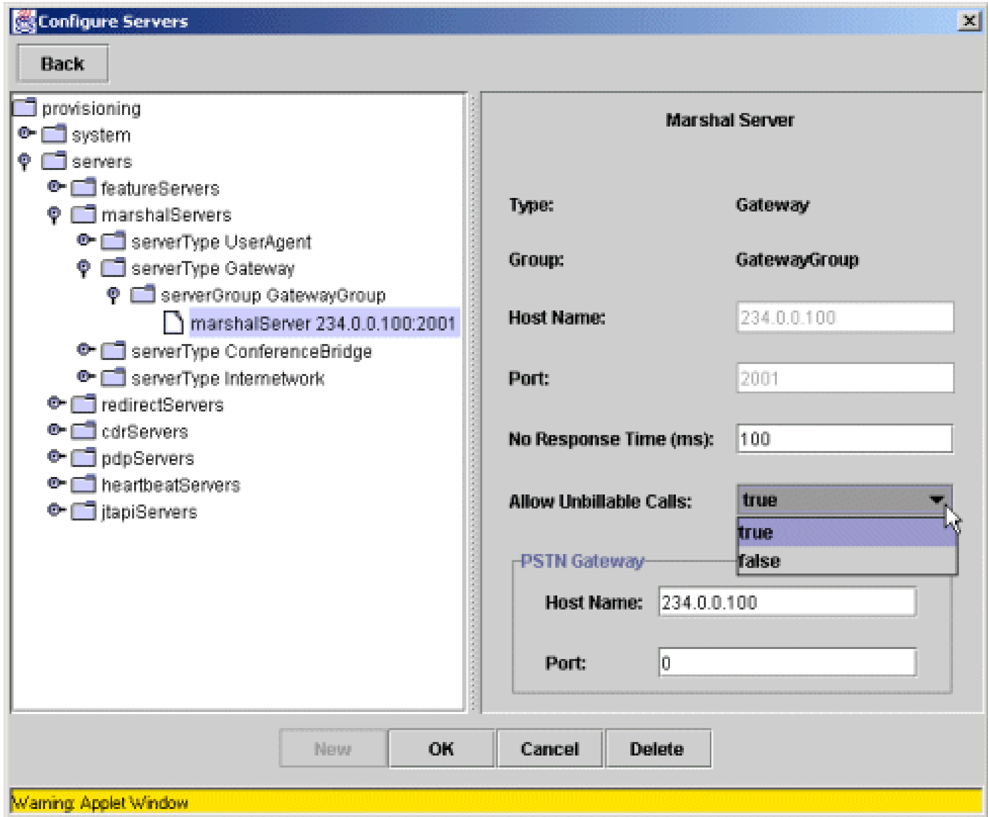


Figure 3-16. Gateway Marshal Server: Data Entry Screen

Data Entry Fields

Table 3-32 describes the fields.

Table 3-32. Gateway Marshal Server: Data Entry Fields

Field	*Description*
Type	A notice describing this server's MS type.
Group	A notice describing this server's MS group.
Host Name	The IP address of this server.
Port	The SIP port number of this server.
No Response Time (ms)	This timer keeps track of the connection to he CDR Server. If the connection cannot be established before the time value, measured in milliseconds, for this cell expires, then the call is considered unbillable.

Table 3-32. Gateway Marshal Server: Data Entry Fields (Continued)

Field	*Description*
Allow Unbillable Calls	If the call is considered unbillable, and this field is set to TRUE, then the call proceeds. If the call is considered unbillable, and this field is set to FALSE, the MS rejects the call with a 402, Payment Required, message.
PSTN Gateway	
Host Name	IP address of the PSTN-to-SIP gateway device that communicates with the marshal.
Port	The SIP port of the PSTN-to-SIP gateway device. *Default:* 5060

Conference Bridge Marshal Servers

Introduction

This section describes how to provision the Conference Bridge Marshal Server (CBMS).

Overview

The CBMS connects to gateways or routers that lead to third party conferencing systems. The VOCAL system supports both “Meet-Me” and “Ad-Hoc” conference calls.

■ **Note**

For Meet-Me conference calls, any marshal type can be used.

Provisioning Tasks

Table 3-33 shows the provisioning tasks that can be performed with the CBMS.

Table 3-33. Conference Bridge Marshal: Provisioning Tasks

Task	***Comments***
Adding a CBMS Group	CBMS groups are the second sub-level below **marshalServers**, and **serverType Conference Bridge** on the directory tree. MS’s are called randomly within the same group, therefore adding more servers to the same group will enhance performance as well as reliability.
Adding a CBMS	CBMS groups are the third sub-level below **marshalServers**, **serverType Conference Bridge** and **marshalServer conferenceBridge** <IP address:port> on the directory tree.
Editing a CBMS	Select a server, edit the fields and then click **OK**. Table 3-34 describes the fields.
Adding Additional CBMS	Additional servers can be added as the system grows in size, or if greater reliability is desired. Follow the instructions in “Adding New Servers” on page 3-34. For more information about scaling the system, see Appendix A, Engineering Guidelines.
Deleting CBMS	Select a server and click **Delete**.

Conference Bridge Marshal Servers: Data Entry Fields

Introduction

This section shows and describes the data entry fields.

Screen Capture

Figure 3-17 shows the CBMS's data entry fields.

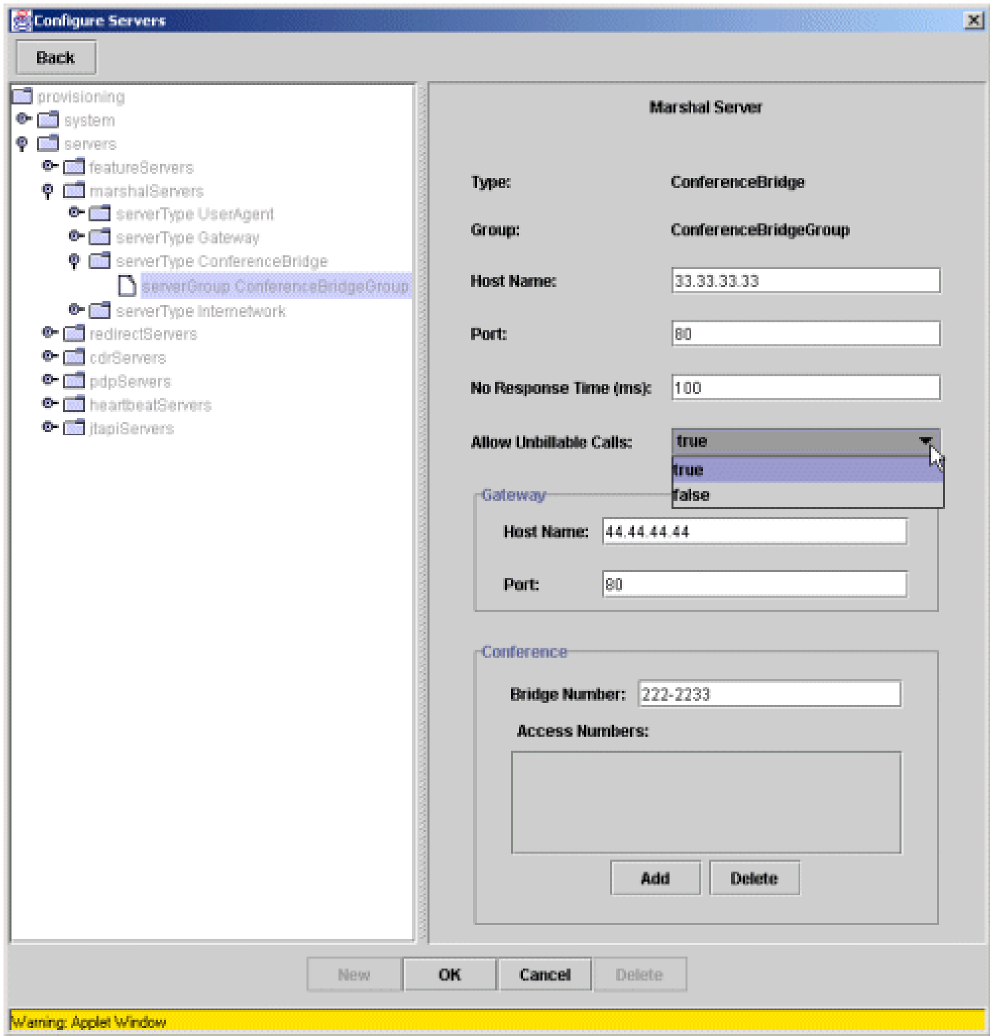


Figure 3-17. Conference Bridge Marshal Server: Data Entry Screen

Data Entry Fields

Table 3-34 describes the fields.

Table 3-34. Conference Bridge Marshal Server: Data Entry Fields

Field	Description
Type	A notice describing this server's MS type.
Group	A notice describing this server's MS group.
Host Name	The IP address of this server.

Table 3-34. Conference Bridge Marshal Server: Data Entry Fields

Field	***Description***
Port	The port number of this server.
No Response Time (ms)	This timer keeps track of the connection to he CDR Server. If the connection cannot be established before the time value, measured in milliseconds, for this cell expires, then the call is considered unbillable.
Allow Unbillable Calls	If the call is considered unbillable, and this field is set to TRUE, then the call proceeds. If the call is considered unbillable, and this field is set to FALSE, the MS rejects the call with a 402, Payment Required, message.
Gateway	
Host Name	IP address of the PSTN-to-SIP gateway device that communicates with the MS.
Port	The SIP port of the PSTN-to-SIP gateway device. *Default:* 5060
Conference	
Bridge Number	A well-known phone number used by user agents to make ad-hoc conference calls. This number can be any length. Dashes are not required.
Access Numbers	A list of numbers that match the access numbers for the conference bridge. The CBMS maps the access numbers to the bridge numbers.

Buttons

Table 3-35 describes the buttons.

Table 3-35. Conference Bridge Marshal Server: Buttons

Button	***Description***
Add	Adds Bridge Numbers to the Access List.
Delete	Deletes selected Bridge Numbers from the Access List.

Internetwork Marshal Servers

Introduction

This section describes how to provision the Internetwork Marshal Servers (INMS).

Overview

The INMS connect the VOCAL system to Internet gateways and routers.

Provisioning Tasks

Table 3-36 shows the provisioning tasks that can be performed with the CDR Server.

Table 3-36. Internetwork Marshal Server: Provisioning Tasks

Action	*Comments*
Adding a INMS Group	INMS groups are the second sub-level below **marshalServers**, and **serverType interNetwork** on the directory tree.
Adding a INMS	INMS groups are the third sub-level below **marshalServers**, **serverType Internetwork** and **marshalServer Internetwork** <IP address:port> on the directory tree.
Editing an INMS	Select a server, edit the fields and then click **OK**. Table 3-37 describes the fields.
Adding Additional INMS's	Additional servers can be added as the system grows in size, or if greater reliability is desired. Follow the instructions in "Adding New Servers" on page 3-34. For more information about scaling the system, see Appendix A, Engineering Guidelines.
Deleting INMS's	Select a server and click **Delete**.

Internetwork Marshal Servers: Data Entry Fields

Introduction

This section shows and describes the data entry fields.

Screen Capture

Figure 3-18 shows the INMS's data entry fields.

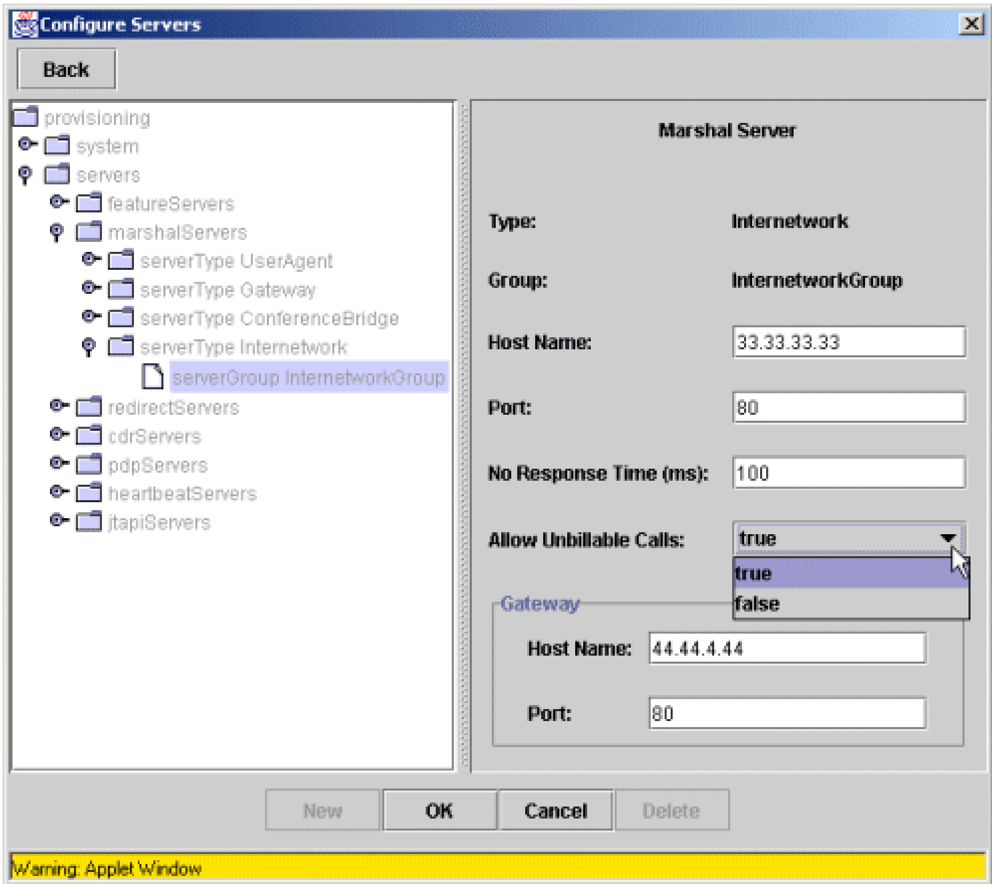


Figure 3-18. Internetwork Marshal Server: Data Entry Screen

Data Entry Fields

Table 3-37 describes the fields.

Table 3-37. Internetwork Marshal Server: Data Entry Fields

Field	Description
Type	A notice describing this server's MS type.
Group	A notice describing this server's MS group.
Host Name	The IP address of this server.
Port	The port number of this server.

Table 3-37. Internetwork Marshal Server: Data Entry Fields (Continued)

Field	*Description*
No Response Time (ms)	This timer keeps track of the connection to he CDR Server. If the connection cannot be established before the time value, measured in milliseconds, for this cell expires, then the call is considered unbillable.
Allow Unbillable Calls	If the call is considered unbillable, and this field is set to TRUE, then the call proceeds. If the call is considered unbillable, and this field is set to FALSE, the MS rejects the call with a 402, Payment Required, message.
Gateway	
Host Name	IP address of the PSTN-to-SIP gateway device that communicates with the MS.
Port	The SIP port of the PSTN-to-SIP gateway device. *Default:* 5060

Feature Servers

Introduction

This section describes how to provision the Feature Servers (FS's).

Overview

The VOCAL system supports the following features:

- Call Blocking
- Caller ID Blocking
- Call Forward All Calls
- Call Forward No Answer Busy
- Call Return
- Call Screening
- Voice Mail

There is a separate FS for all of these. On smaller systems, the features may all reside on the same host, but unlike the Provisioning (PS) and Redirect Servers (RS's), there is no master FS.

For more information about how to assign features to users, see the System Administration Guide, Chapter 1, Setting Up Users.

For more information about how features are used by subscribers, see the System Administration Guide, Appendix A, Using Features.

Provisioning Tasks

Table 3-38 shows the provisioning tasks that can be performed with the FS's.

Table 3-38. Feature Servers: Provisioning Tasks

Task	*Comments*
Adding a FS Group	FS groups are the second sub-level below **featureServers**, and **serverType** <feature name> on the directory tree.
Adding a FS	Every feature type must be separated into its own group.
Editing a FS	Select a server, edit the fields and then click **OK**. Table 3-39 describes the fields.
Adding Additional FS's	Additional servers can be added as the system grows in size, or if greater reliability is desired. Follow the instructions in "Adding New Servers" on page 3-34. For more information about scaling the system, see Appendix A, Engineering Guidelines.
Deleting FS's	Select a sever and click **Delete**. **Caution** Deleting FS's in live systems is not recommended. If you delete a FS that has users assigned to it, you will have to regenerate all of those users.

Feature Severs, Data Entry Fields

Introduction

This section shows and describes the data entry fields.

Screen Capture

Figure 3-18 shows a FS's data entry fields.

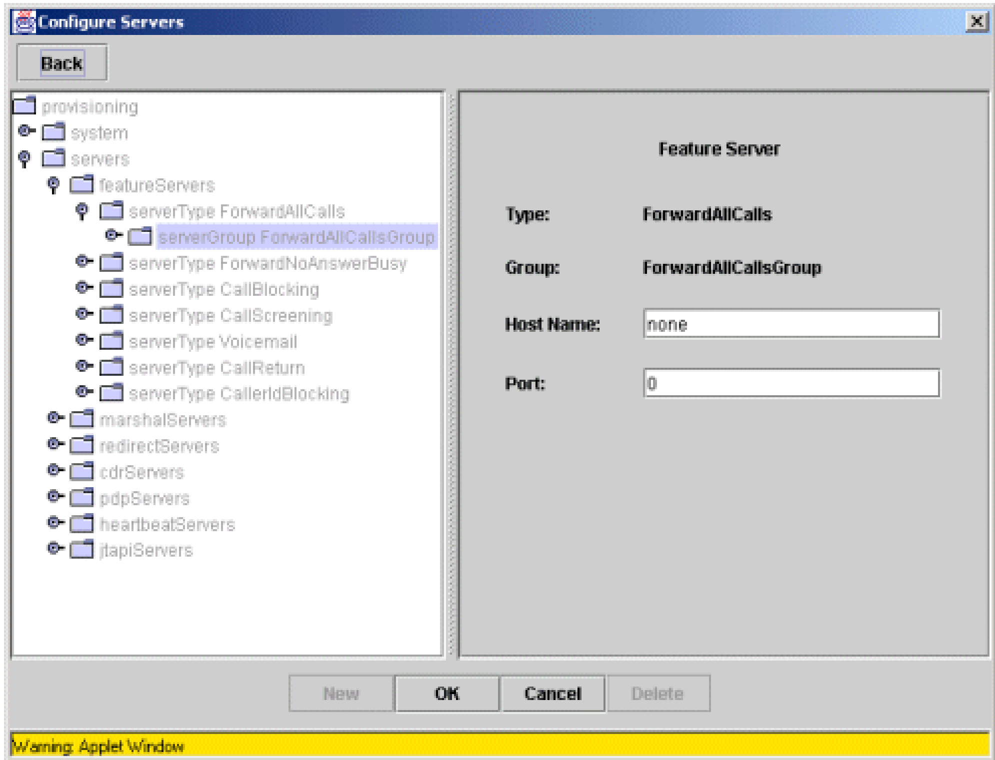


Figure 3-19. Feature Server: Data Entry Screen

Data Entry Fields

Table 3-39 describes the fields.

Table 3-39. Feature Server: Data Entry Fields

Field	Description
Type	A notice describing this server's FS type.
Group	A notice describing this server's FS group.
Host Name	The host name or IP address of this server.
Port	The SIP port number of this server.

Voice Mail Feature Servers

Introduction

This section describes how to provision the Voice Mail Feature Servers (VMFS's).

Overview

The VMFS provides the VOCAL system with the capability to forward calls to a voice mail system.

Provisioning Tasks

Table 3-40 shows the provisioning tasks that can be performed with the VMFS.

Table 3-40. Voice Mail Feature Server: Provisioning Tasks

Task	***Comments***
Adding a VMFS Group	VMFS groups are the second sub-level below **featureServers**, and **serverType voiceMail** on the directory tree.
Adding a VMFS	VMFS groups are the third sub-level below **featureServers**, and **serverType voiceMail** on the directory tree.
Editing a VMFS	Select a server, edit the fields and then click **OK**. Table 3-41 describes the fields.
Adding Additional VMFS's	Additional servers can be added as the system grows in size, or if greater reliability is desired. Follow the instructions in "Adding New Servers" on page 3-34. For more information about scaling the system, see Appendix A, Engineering Guidelines.
Deleting VMFS's	Select a server and click **Delete**. **Caution** Deleting servers in live systems is not recommended. If you delete a Feature Server that has users assigned to it, you must regenerate all of those users.

Voice Mail Feature Server, Data Entry Fields

Introduction

This section shows and describes the data entry fields.

Screen Capture

Figure 3-18 shows the VMFS's data entry fields.

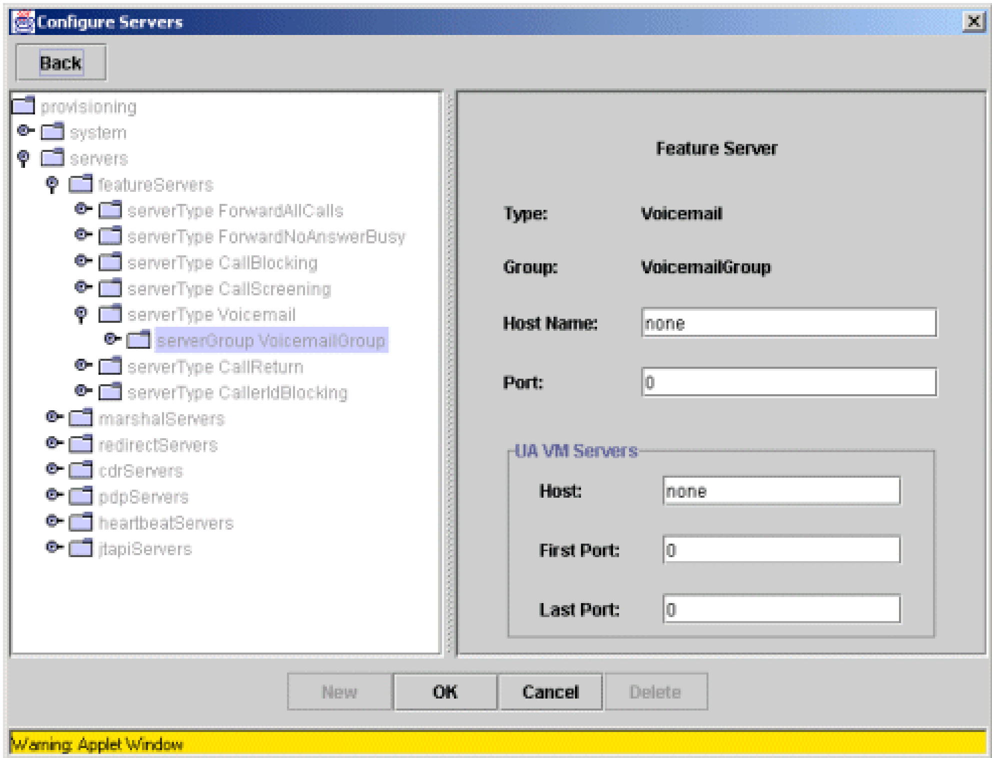


Figure 3-20. Voice Mail Feature Server: Data Entry Screen

Data Entry Fields

Table 3-41 describes the fields.

Table 3-41. Voice Mail Feature Server: Data Entry Fields

Field	Description
Type	A notice describing this server's FS type.
Group	A notice describing this server's FS group.
Host Name	The host name or IP address of this server.
Port	The SIP port number of this server.

Table 3-41. Voice Mail Feature Server: Data Entry Fields (Continued)

Field	*Description*
UAVM Servers	
Host	The host name or IP address of the User Agent Voice Mail (UAVM) Server.
First Port	The first available UDP port for receiving voice mail messages.
Last Port	The last available UDP port for receiving messages. ■**Note** The First Port and Last Port fields define the quantity of available ports for voice mail. If there are 10 ports, then the system will accept 10 voice mail users at any one time, and return a busy signal for all other callers.

JTAPI Servers

Introduction

This section describes how to provision the JTAPI Feature Server.

Overview

Java Telephony Application Programming Interface (JTAPI) is a Sun Microsystems specification for providing computer telephony intelligence (CTI). CTI applications are typically designed for call centers for functions such as controlled call redirection and automated dialing. While Sun provides the specification, there is no implementation library.

The JTAPI specification describes 5 packages:

- Core
- Call Control
- Phone
- Media
- Call Center

The VOCAL system includes an implementation of the Core package that supports basic third-party call control capability, and a sample application, proposed name, “VOCALpad”, that utilizes the implementation.This means that a user can control a user agent (UA) by running and a basic User Agent application (proposed name, “VOCALpad”) on their PC, and instructing the UA to call the called party.

Provisioning Tasks

Table 3-42 shows the provisioning tasks that can be performed with the JTAPI Feature Server.

Table 3-42. JTAPI Server: Provisioning Tasks

Action	*Comments*
Adding a JTAPI Server Group	JTAPI server groups are the first sub-level below the **jtapiServers** on the directory tree.
Adding a JTAPI Server	JTAPI servers are the second sub-level below the **jtapiServers** on the directory tree.
Editing a JTAPI Server	Select a server, edit the fields and then click **OK**. Table 3-43 describes the fields.
Adding Additional JTAPI Servers	Additional servers can be added as the system grows in size. Follow the instructions in “Adding New Servers” on page 3-34. For more information about scaling the system, see Appendix A, Engineering Guidelines.
Deleting JTAPI Servers	Select a server and click **Delete**.

JTAPI Servers, Data Entry Fields

Introduction

This section shows and describes the data entry fields.

Screen Capture

Figure 3-21 shows the JTAPI Server's data entry fields.

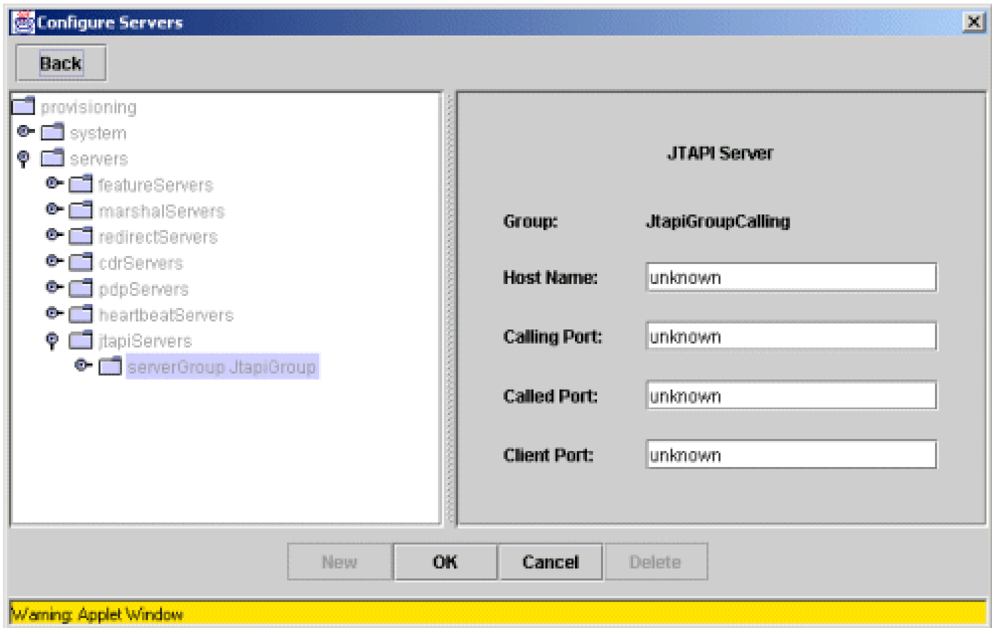


Figure 3-21. JTAPI Server: Data Entry Screen

Data Entry Fields

Table 3-43 describes the fields.

Table 3-43. JTAPI Server: Data Entry Fields

Field	*Description*
Host Name	The host name or IP Address of this server.
Calling Port	One of the SIP ports used by the JTAPI Server for sending and receiving SIP messages. *Example:* 5080
Called Port	A second SIP port used by the JTAPI Server for sending and receiving SIP messages. *Example:* 5081
Client Port	The UDP port used by the JTAPI Server for communication with the JTAPI clients. The JTAPI clients are the java applications used for controlling a User Agent. *Example:* 5082

Heartbeat Servers

Introduction

This section describes how to provision the Heartbeat Server (HS).

Overview

Every host transmits regular pulses called heartbeats. These heartbeats allow the other hosts on the network determine if any of the servers are down. The HS manages the heartbeat flow.

The HS is used by the Network Manager (NM) to update the GUI table of servers and server states. If your system does not include a NM, the HS is not required.

Provisioning Tasks

Table 3-44 shows the provisioning tasks that can be performed with the feature servers.

Table 3-44. Heartbeat Server: Provisioning Tasks

Action	***Comments***
Adding a HS Group	HS groups are the first sub-level below **heartbeatServers** on the directory tree. There is normally only one Heartbeat Server on a system.
Adding a HS	HS's are the second sub-level below **heartbeatServers** on the directory tree. There is normally only one Heartbeat Server on a system.
Editing a HS	Select a server, edit the fields and then click **OK**. Table 3-45 describes the fields.
Deleting HS	Select a server and click **Delete**.

Heatbeat Servers, Data Entry Fields

Introduction

This section shows and describes the data entry fields.

Screen Capture

Figure 3-22 shows the Heartbeat Server's data entry fields.

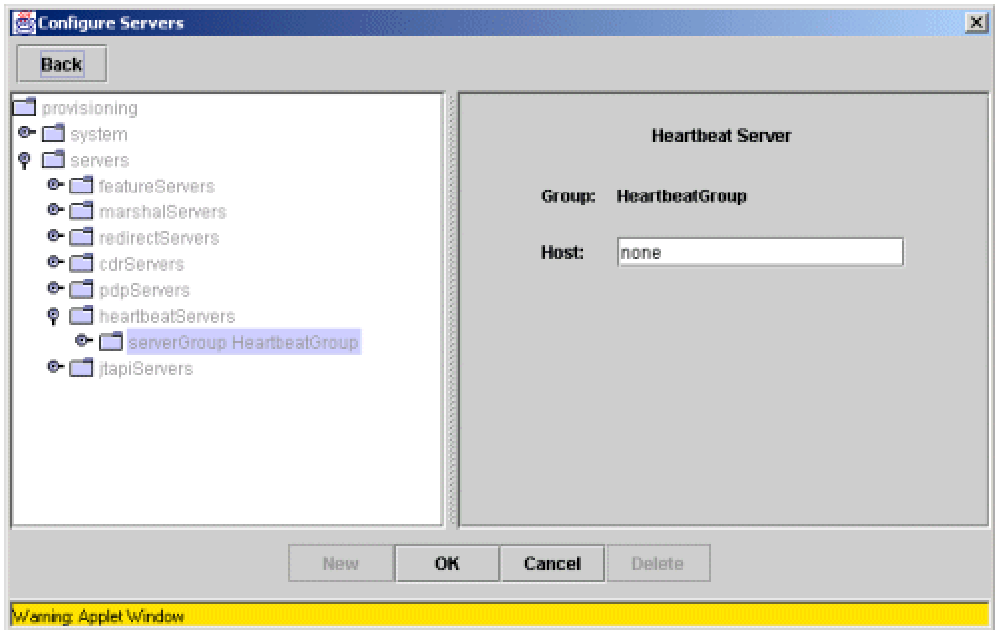


Figure 3-22. Heartbeat Server: Data Entry Screen

Data Entry Fields

Table 3-45 describes the fields.

Table 3-45. Heartbeat Server: Data Entry Fields

Field	Description
Host	The IP address of this server.
Port	The port number for this server.

Policy Servers

Introduction

This section describes how to provision the Policy Server (PoS).

Overview

The PoS is the key component used to achieve Quality of Service (QoS). Service providers typically will only ensure QoS if authorizations and payments are guaranteed by a third party. The PoS administers admission control for QoS requests and provides the Internetwork Marshal (policy client) with the information necessary to enforce the admitted QoS requests. The PoS out sources the Authorization, Authentication and Accounting (AAA) requests to a third-party clearing house, which then acts as a trusted broker between a large number of network providers.

Provisioning Tasks

Table 3-44 shows the provisioning tasks that can be performed with the PoS.

Table 3-46. Policy Server: Provisioning Tasks

Task	*Comments*
Adding a PoS Group	PoS groups are the first sub-level below **pdpServers** on the directory tree. There is normally only one PoS on a system.
Adding a PoS	PoS's are the second sub-level below **pdpServers** on the directory tree. There is normally only one PoS on a system.
Editing a PoS	Select a server, edit the fields and then click **OK**. Table 3-47 describes the fields.
Deleting PoS	Select a server and click **Delete**.

Policy Servers, Data Entry Fields

Introduction

This section shows and describes the data entry fields.

Screen Capture

Figure 3-22 shows the Policy Server's data entry fields.

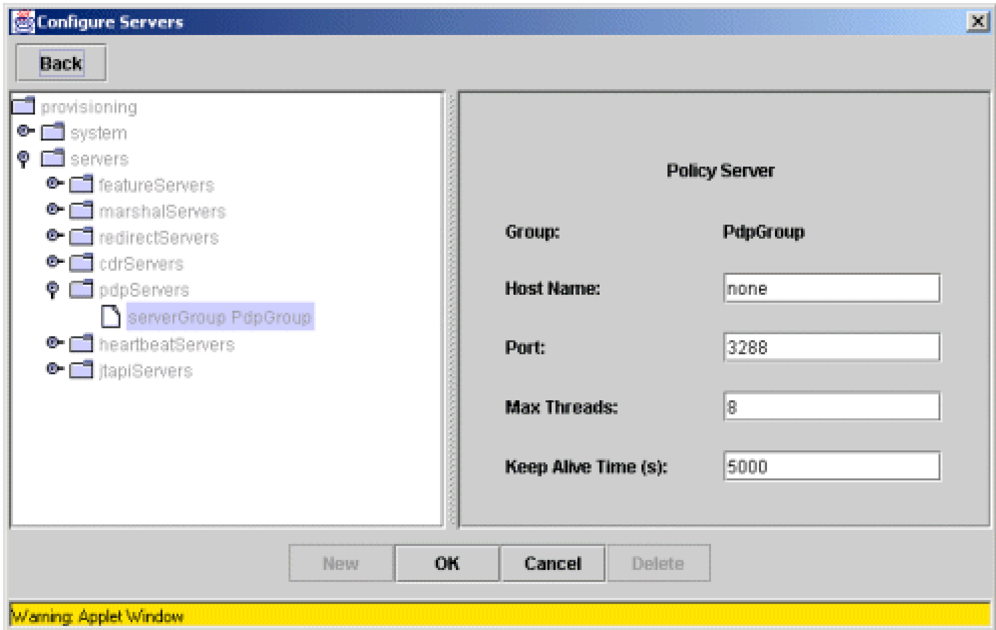


Figure 3-23. Policy Server: Data Entry Screen

Data Entry Fields

Table 3-47 describes the fields.

Table 3-47. Policy Server: Data Entry Fields

Field	***Description***
Host Name	The host name or IP address of this server.
Port	The port number of this server.
Max Threads	The maximum number of permitted connections to the PoS. *Recommended:* 5-8
Keep Alive Timer	The time, measured in seconds, to maintain the TCP/IP connection to the PoS.
Response Timer	The time, measured in milliseconds, for the Policy Enforcement Point (PEP) Client to wait for a response from the Policy Decision Point (PDP) on the Policy Server.

A

Engineering Guidelines

Chapter Content

This appendix describes how to scale the system properly to accommodate more customers.

Scaling Guidelines and Redundancy

Scaled Systems

Table A-1 lists three different VOCAL system setups and the supported capacity based on number of calls per second and number of busy hour call attempts.

■ **Note**

The scaled system identified in Table A-1 assume this hardware configuration for each server: a 700MHz Pentium III PC with 512MB RAM.

Table A-1. Scaled VOCAL Systems

Server Types	*6 Server System*	*14 Server System*	*26 Server System*
Redirect Servers	1	2	5
Feature Servers	1	2	5
Marshal Servers	2	4	10
Call Detail Record Servers	1/2	2	2
Provisioning Servers	1	2	2
Policy Servers	1/2	2	2
Total Number of Servers	6	14	26
Capacity in Call Per Second	35	70	175
Capacity in Busy Hour Call Attempts	125,000	250,000	630,000

Marshal Server

For scalability, multiple Marshal servers can exist in a VOCAL system.

Feature Server

For scalability, multiple Feature servers can exist in a VOCAL system.

Redirect Server

For scalability, multiple Redirect servers can exist in a VOCAL system. Each Redirect server contains the same information and registration information is shared between the Redirect servers.

Call Detail Record (CDR) Servers

For redundancy, a maximum of two CDR servers can exist in a VOCAL system. Marshal servers will send billing data to both primary and secondary CDR servers.

A Marshal Server sends billing data to the CDR server by this process:

Table A-2. Process: Marshal Server Sending Billing Data to the CDR Server

Step	*Action*
1	The Marshal server sends billing data to the primary CDR server in a UDP message and waits for a response.

Table A-2. Process: Marshal Server Sending Billing Data to the CDR Server (Continued)

Step	***Action***
2	If the primary CDR server does not respond in a defined time, the Marshal server re-transmits the UDP message.
3	If the primary CDR server does not respond after a defined number of messages from the Marshal server, the Marshal server sends the billing data to the backup CDR server.

Provisioning Server

For redundancy, a maximum of two Provisioning servers can exist in a VOCAL system. If two Provisioning server exist is a VOCAL system, information saved on provisioning server is shared with the other provisioning server. The Provisioning servers will synchronize information periodically.

B

Resources on the Web

Appendix Content This appendix lists the Uniform Resource Locators (URL's) for additional research material.

Topic	See Page

Web Resources

Vovida Website

Visit the following web site for more information about VOCAL: http://www.vovida.org

Reference Sites

Table B-1 lists a collection of useful reference sites on the Internet.

Table B-1. Useful Websites

Description	*URL*
5th SIP Bake-Off	http://www.pulver.com/sip/
Cable Labs	http://www.packetcable.com/
Call Processing Language (CPL) - IETF Internet Draft	http://www.cs.columbia.edu/~lennox/draft-ietf-iptel-cpl-02.txt
Center for Democracy and Technology	http://www.cdt.org/
Codec Central	http://www.terran-int.com/CodecCentral/Codecs/index.html
Cooperative Market for Open Source	http://www.cosource.com/
Delphion Intellectual Property Network	http://www.patents.ibm.com/ibm.html
Doc++	http://www.zib.de/Visual/software/doc++/index.html
Electronic Frontier Foundation	http://www.eff.org/
Fresh Meat.net	http://freshmeat.net/
H.323	http://www.openh323.org/
H.323	http://www.packetizer.com/
IETF Drafts Relating to SIP	http://www.ietf.org/ids.by.wg/sip.html
International Telecommunication Union	http://www.itu.int/
Internet drafts	ftp://ftp.isi.edu/internet-drafts/
LIBXML software	ftp://rpmfind.net/pub/veillard/libxml/
Linux Documentation Project	http://www.linuxdoc.org/
Linux HeadQuarters	http://www.linuxhq.com/
Linux Telephony	http://www.linuxtelephony.org/
Linux TV.org	http://linuxtv.org/
Microsoft Research	http://research.microsoft.com/
Multiparty Multimedia Session Control	http://www.ietf.org/html.charters/mmusic-charter.html
Opencode.org	http://eon.law.harvard.edu/opencode/

Table B-1. Useful Websites (Continued)

Description	***URL***
Pingtel	http://www.pingtel.com/homepage.php3
Programming in C++, Rules and Recommendations By Mats Henricson and Erik Nyquist.	http://kastanie.informatik.tu-cottbus.de/cpp/style_guides/ellemtel/index.htm
Red Hat Software	http://www.redhat.com/
RTP News	http://www.cs.columbia.edu/~hgs/rtp/
Session Initiation Protocol (SIP) Working Group Supplemental Home Page	http://www.softarmor.com/sipwg/
SIP	http://www.cs.columbia.edu/~hgs/sip/
SIP and Internet Telephony: Papers, Books and Talks	http://www.cs.columbia.edu/sip/papers.html
SIP Charters	http://www.ietf.org/html.charters/sip-charter.html
Softswitch Consortium	http://www.softswitch.org
Source Force	http://sourceforge.net/
Source Xchange	http://www.sourcexchange.com/info.html
The Apache Software Foundation	http://www.apache.org/
Voxilla.org	http://www.voxilla.org/
Vovida.org	http://www.vovida.org/

Index

Index (Continued)

Index (Continued)